

IT ワールド

Information
Technology
World



*IT*ワールド

Information Technology World

ITワールド

Introduction

1	コンピュータと情報社会	2
2	社会の中のコンピュータ	4
3	コンピュータをとりまく世界と本書の構成	7
4	情報処理技術者試験と本書の関係	9

第1部 ハードウェア

第1章	コンピュータの基本構成	12
1-1	コンピュータの種類	13
1-2	コンピュータの五大装置	15
第2章	コンピュータのデータ表現	18
2-1	データの表現	18
2-2	基数と基数変換	21
2-3	データの表現形式	25
第3章	中央処理装置と主記憶装置	40
3-1	中央処理装置の構成	40
3-2	主記憶装置の構成	43
3-3	命令とアドレッシング	46
3-4	ALUの構成回路	53
3-5	高速化技術	61
第4章	補助記憶装置	68
4-1	磁気ディスク	68
4-2	光ディスク	76
4-3	半導体メモリ	78
4-4	その他の補助記憶装置	79
第5章	入出力装置	80
5-1	入力装置	81
5-2	出力装置	86
5-3	その他の入出力装置	90
5-4	入出力制御方式	91
5-5	入出力インタフェース	92
	演習問題	98

第2部 情報処理システム

第1章	情報処理システムの処理形態	106
1-1	非対話型処理システムと対話型処理システム	106
1-2	一括処理システムと即時処理システム	107
1-3	集中処理システムと分散処理システム	108
第2章	高信頼化システムの構成	114
2-1	直列システム	114
2-2	並列システム	115
2-3	多重化システム	117
第3章	情報処理システムの評価	120
3-1	処理能力の評価	120
3-2	信頼性の評価	125
3-3	経済性の評価	132
第4章	ヒューマンインタフェース	133
4-1	ヒューマンインタフェース技術	133
4-2	インタフェース設計	136
第5章	マルチメディア	144
5-1	マルチメディア技術	144
5-2	マルチメディア応用	150
	演習問題	152

第3部 ソフトウェア

第1章	ソフトウェアの分類	156
1-1	ソフトウェアの体系による分類	156
1-2	ソフトウェアライセンスによる分類	163
第2章	OS（オペレーティングシステム）	166
2-1	OSの機能と構成	166
2-2	OSの管理機能	167
第3章	プログラム言語と言語プロセッサ	180
3-1	プログラム言語の分類	180
3-2	言語プロセッサ	186
3-3	プログラムの属性	193
第4章	ファイル	194
4-1	ファイルとレコード	194
4-2	ファイルのアクセス方式	197
4-3	ファイル編成方式	198
4-4	小型コンピュータのファイル管理	200
4-5	バックアップ	203
	演習問題	204

第4部 データベース

第1章	データベースの概要	210
1-1	データベースとファイルの違い	210
1-2	データベースの設計	211
1-3	データベース管理システム (DBMS)	218
第2章	SQL	224
2-1	データ定義	225
2-2	データ操作	229
第3章	いろいろなデータベース	241
3-1	分散データベース	241
3-2	データウェアハウス	242
3-3	ビッグデータ	242
3-4	その他の関連技術	244
	演習問題	245

第5部 ネットワーク

第1章	インターネット	250
1-1	インターネットの接続方法	251
1-2	インターネットの基本構成	254
1-3	インターネットサービス	255
1-4	インターネットの標準プロトコル	259
第2章	ネットワークアーキテクチャ	266
2-1	ネットワークアーキテクチャとは	266
2-2	OSI (開放型システム間相互接続)	266
2-3	TCP/IP	268
第3章	LAN	270
3-1	LAN の基礎技術	271
3-2	その他の LAN 技術	277
第4章	ネットワークの仕組み	278
4-1	ネットワークの構成要素	278
4-2	ネットワークの基礎技術	279
4-3	伝送制御手順	287
4-4	IoT 関連技術	291
第5章	ネットワーク管理	292
5-1	ネットワーク運用管理	292
5-2	ネットワーク管理手法	293
	演習問題	294

第6部 セキュリティ

第1章	情報セキュリティの概要	300
1-1	情報セキュリティの概念	300
1-2	情報セキュリティ技術	313
1-3	情報セキュリティ管理	324
1-4	情報セキュリティ機関・評価基準	330
第2章	情報セキュリティ対策	332
2-1	物理的セキュリティ対策	332
2-2	人的セキュリティ対策	333
2-3	技術的セキュリティ対策	334
2-4	セキュリティ実装技術	338
	演習問題	347

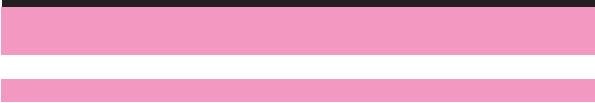
第7部 データ構造とアルゴリズム

第1章	データ構造	354
1-1	配列	354
1-2	リスト	356
1-3	スタックとキュー	358
1-4	木構造（ツリー構造）	360
第2章	基本アルゴリズム	365
2-1	フローチャート	365
2-2	データ探索処理	368
2-3	データ整列処理	374
2-4	その他のアルゴリズム	382
2-5	アルゴリズム設計	386
	演習問題	387

索引	391



Introduction



1 コンピュータと情報社会

現在、私たちが生活している社会は Society 4.0「情報社会（又は情報化社会）」から Society 5.0「超スマート社会」^{注1}へと向かっている。「情報社会」とは、「氾濫するさまざまなデータの中から、自分にとって有利な情報を見つけ出す（作り出す）必要がある社会」のことである。この説明を聞くと、「データと情報はどこが違うのだろう？」と疑問に思うかもしれない。実は、「データ」と「情報」は、まったく違うものなのである。それぞれの意味を辞書で調べると、次のように説明してある。

データ = 「議論や研究の基礎となる資料」

情報 = 「事情や状況の報告、ある人にとって有益なデータ」

この説明だけでは違いがわかりにくいため、例をあげて説明する。

今「20080607」という数字列があったとする。この数字列は、このままでは意味のない数字の羅列にすぎない（これが「データ」である）。しかし、「20080607」を「2008」年「06」月「07」日と解釈すると、例えば，“誕生日”という意味をもつことになる。このように、意味をもったデータを「情報」という。つまり、「データ」は意味をもって、初めて「情報」となるのである。

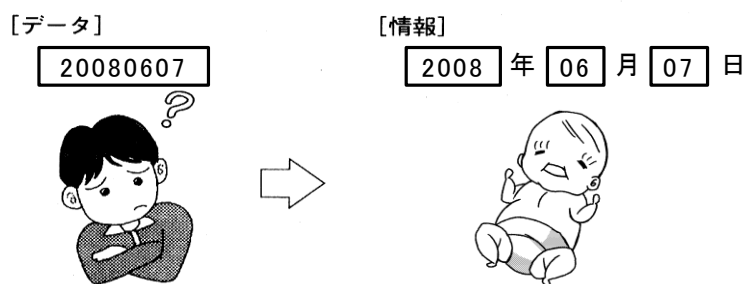


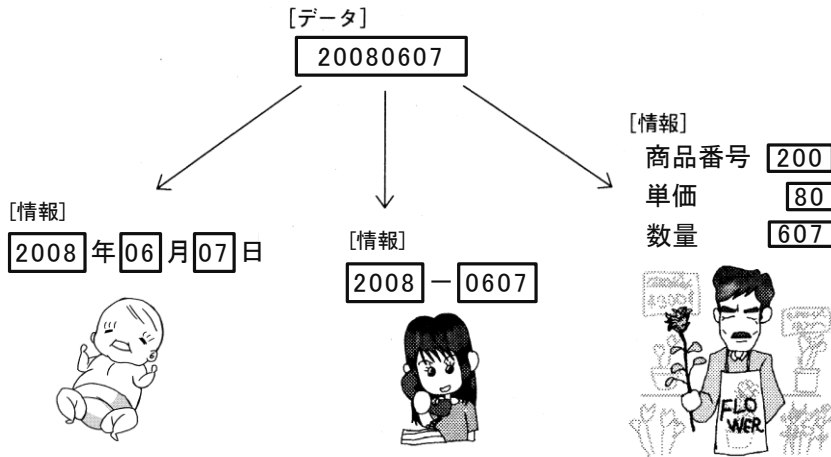
図 I データと情報の違い

注1：「ICT を最大限に活用し、サイバー空間とフィジカル空間（現実世界）とを融合させた取組により、人々に豊かさをもたらす『超スマート社会』を未来社会の姿として共有し、その実現に向けた一連の取組を更に深化させつつ『Society 5.0』として強力に推進し、世界に先駆けて超スマート社会を実現していく。」

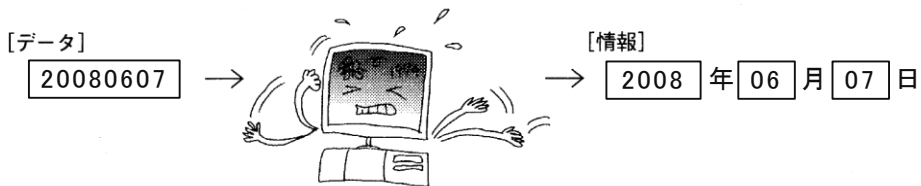
（『第5期科学技術基本計画』より引用）

ここで大事なことは、「データ」を「情報」として正しく解釈することである。

例えば、先ほど“誕生日”と解釈した数字列「20080607」は、実は「2008」－「0607」という“電話番号”かもしれないし、商品番号「200」、単価「80」円の商品が「607」個売れたという“商品売上記録”かもしれない。つまり、「データ」とは解釈の仕方により、まったく違う「情報」になってしまうのである。そこで、データを正しく、速く処理することが重要となる。このために必要とされた道具が「コンピュータ」^{注2}なのである。



図Ⅱ データはさまざまに解釈できる



図Ⅲ コンピュータは正しい情報を素早く手に入れるための道具である

注2：基本情報技術者試験（レベル2）シラバス Ver. 8.0 より、一部の用語で語尾の長音符（音引き）が記されるようになった（「プリンター」、「ユーザー」など）。ただし、「コンピュータ」、「プロセッサ」などの用語も混在するため、本書では語尾の長音符を省いた表記に統一している。

2 社会の中のコンピュータ

実際に、社会でどのようにコンピュータが利用されているのか、いろいろな例をあげながら説明していく。

① POS システム

これは「Point Of Sales システム」の略称で、日本語に直すと「販売時点情報管理システム」となる。複雑そうな名前ではあるが、現在ではなじみ深いものである。

スーパーストアやコンビニエンスストアで買い物をすると、レジで商品に機械を当てて「ピッ」「ピッ」「ピッ」と**バーコード**を読み取る。あれが、POS システムに商品データを入力する「**バーコードリーダ**」という装置である。バーコードとは、商品に付いている線の集まりで、線の太さや間隔の違いなどによりいろいろな情報を表している。この情報を読み込んで、どんな商品が売れているのかなどを分析し、商品の仕入数を決定したりするのが「POS システム」の役割である。



POS システム端末



4988 615 018428

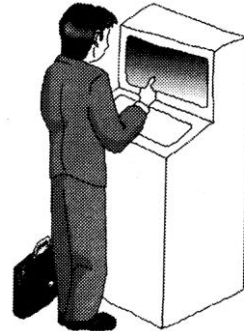
バーコード



バーコードリーダ

② ATM システム

銀行などには、カードや通帳で現金を引き出したり、残高を確認したりする機械が置いてある。利用者は、この機械にカードや通帳を入れ、入金・出金・残高照会などの処理の指示や金額などを入力する。すると、銀行で記録している個人情報（預金残高など）を基に、処理（現金が出てくるなど）が行われる。このとき、例えば現金を引き出した場合には、出金した分の金額を預金残高から引く処理も同時に行われる必要がある。これらの処理を矛盾なく行うのが、「ATM システム」である。このシステムのおかげで、混雑するカウンタ窓口に並ぶ必要がなくなったり、預金してある銀行以外からお金を引き出すことができたり、非常に便利になっている。



ATM システム

③ 座席予約システム

新幹線や飛行機の座席から、コンサートなどのチケットに至るまで、さまざまな予約のために使用されているシステムである。このようなシステムがなかった時代は、予約業務を人間が手作業で行っていた。そのため、特定の場所でしか予約ができないので朝早くから行列したり、同じ座席を複数の人が予約するなどのトラブルが発生したりした。現在の予約システムしか利用したことのない人達から見ると、信じられない話かもしれない。



座席予約システム

④ インターネット

今や知らない人はいないといっても過言ではないのが「インターネット」である。「インターネット」では、接続用端末としてコンピュータを使用する。そして、「インターネット」を管理／運営するのにコンピュータが利用されている。まさに、今日の情報化社会における代表的なシステムといえるのかもしれない。

“インターネットを利用してできること”には、大きく分けて「情報検索／発信」と「オンラインショッピング」がある。

〔情報検索／発信〕

自分が知りたい情報を調べたり（検索）、自分の知っている情報を多くの人にも知ってもらうために公開（発信）することである。現在、数多くの検索エンジン（情報を調べる手助けをするための道具）があり、また個人から企業に至るまで、数多くのホームページが公表されている。



ホームページの一例

〔オンラインショッピング〕

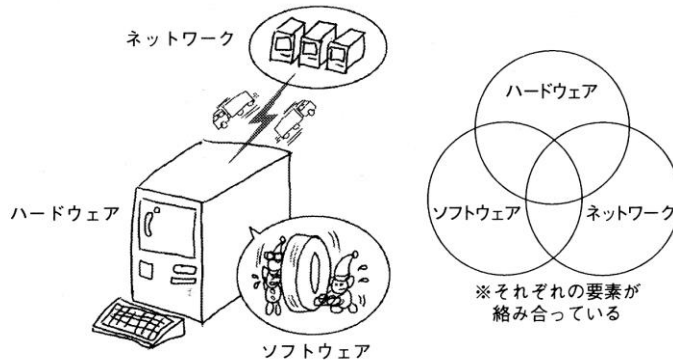
インターネットを利用して買い物をするすることである。現在では、多くの企業が自社のホームページの中で自社製品を販売している。また、実際の店舗をもたずに、インターネットだけで商品を販売する仮想商店やショッピングモールが数多く登場している。決済方法も、クレジットカードやオンラインバンキングからの振込が普及し、自宅での買い物に便利な環境が整っている。



オンラインショッピングの一例

3 コンピュータをとりまく世界と本書の構成

コンピュータがさまざまな仕事をしている現代では、世界がコンピュータを中心として結びつき、その輪は大きく広がっている。図Ⅳに示すように、現代のコンピュータをとりまく世界は、大きく分けて三つの構成要素が絡み合って成立している。



図Ⅳ コンピュータ世界の構成

前記の「2 社会の中のコンピュータ」で紹介した POS システムを例にあげて、具体的に説明していく。

① ハードウェア

コンピュータを構成する機械的な要素である。近年のハードウェア技術の進歩はめざましいものがあり、少し前までは実現不可能とされていたようなことも、次々と実現されている。また、価格的にも安価になったことが、現在のコンピュータ普及の大きな要因の一つとなっている。

POS システムで、商品のバーコードを読み取るために使用する“バーコードリーダ”がハードウェアの一つである。バーコードを読み込むために、レーザ光を当てて光学的に読み取っている。このようなハードウェアの仕組みについて、「**第1部 ハードウェア**」で学習する。

また、読み取られた情報は店頭にある POS システム端末で処理されるのではなく、本社にある大型コンピュータに送信されてから処理される。このような処理形態を、“オンライントランザクション処理”という。現代では、発生するデータも、それを処理する形態もさまざまである。また、コンピュータが故障すると、その及ぼす影響は広く、大きくなることも多い。したがって、最も適切な処理形態を、信頼性などを含めて考える必要がある。その際に必要となる知識について、「**第2部 情報処理システム**」で学習する。

② ソフトウェア

コンピュータ内部で、コンピュータに仕事を命令する要素である。人間が仕事を指示しない限り、コンピュータは自分で判断して仕事をすることはできない。コンピュータを使って、文書を作成したり、電子メールをやり取りすることができるのは、その処理をするためのソフトウェアがあるからである。

POS システムでは、“バーコードを読み込む”／“データを記録する”／“金額を計算する”などの処理を行う。これらの処理をするためには、ハードウェアに対して指示（例えば、バーコードリーダーに“商品のバーコードを読み込め”という命令）が出される。この指示を行うのが、ソフトウェアの役割である。実際にソフトウェアが指示を出す仕組みや、資源を効率良く使用するための工夫などについて、「第3部 ソフトウェア」で学習する。また、データの管理／利用方法の一つである“データベース”については、「第4部 データベース」で学習する。

また、実際の処理では、例えば入力されたデータに含まれている商品のコードをもとにして、その商品の名前や単価を調べたりする。このとき、たくさんの商品コードの中から目的の商品コードを見つけ出すための、効率の良い手順を考える必要がある。この手順は、データを記録するときの構造と密接な関係がある。データの構造や処理手順について、「第7部 データ構造とアルゴリズム」で学習する。

③ ネットワーク

コンピュータ同士で情報をやり取りするための技術である。近年、コンピュータがこれほど急成長したのは、ネットワークが急成長したからだともいえる。なぜなら、多くのコンピュータが情報を伝え合うことにより、従来では行えなかった仕事もコンピュータでできるようになったからである。

POS システムでは、店頭で読み込んだデータを本社の大型コンピュータで処理するために送信する。データを正しく、速く送信するためには、どのような方法／手順で送信すればよいのかという知識が必要となる。このような通信に関する基礎知識について、「第5部 ネットワーク」で学習する。

以上の①～③を組み合わせると、いろいろな仕事を正確に・速く・安全に行うために、さまざまな“システム”がつくられる（POS システムの“システム”もその意味である）。しかし、このようなシステムや、インターネットに代表されるネットワークで利用されるデータなどが安全に正しく管理されていなければ、私たちは安心して利用することができない。そこで、システムやネットワークを安心して利用できるようにするため、さまざまなセキュリティ技術が用いられている。これらのセキュリティ技術を利用した情報セキュリティ管理の考え方について、「第6部 セキュリティ」で学習する。

4 情報処理技術者試験と本書の関係

情報処理技術者試験は、情報処理業界の人材を育成するための指標の一つとして IPA（独立行政法人 情報処理推進機構）が実施している国家試験である。共通キャリア・スキルフレームワークのレベル 1～3 と対応付けられ、試験の合格をもってレベル判定が行われる。各試験の対象とする人材像は、次のように定義されている。

レベル	試験名称	対象とする人材像
1	IT パスポート試験	職業人及びこれから職業人となる者が備えておくべき、ITに関する共通的な基礎知識をもち、ITに携わる業務に就くか、担当業務に対してITを活用していこうとする者
2	基本情報技術者試験	ITを活用したサービス、製品、システム及びソフトウェアを作る人材に必要な基本的知識・技能をもち、実践的な活用能力を身に付けた者
3	応用情報技術者試験	ITを活用したサービス、製品、システム及びソフトウェアを作る人材に必要な応用的知識・技能をもち、高度IT人材としての方向性を確立した者

本書は、情報処理技術者試験の基本情報技術者試験における科目 A 出題範囲のテクノロジー系の分野について学習できる構成になっている。本書の各部と基本情報技術者試験の出題分野との対応関係は、次のとおりである。

IT ワールド		基本情報技術者試験の出題分野
第 1 部	ハードウェア	1 基礎理論（基礎理論〔離散数学など〕）
		2 コンピュータシステム（コンピュータ構成要素）
第 2 部	情報処理システム	2 コンピュータシステム（システム構成要素）
		3 技術要素（ヒューマンインタフェース）
		3 技術要素（マルチメディア）
第 3 部	ソフトウェア	2 コンピュータシステム（ソフトウェア）
第 4 部	データベース	3 技術要素（データベース）
第 5 部	ネットワーク	3 技術要素（ネットワーク）
第 6 部	セキュリティ	3 技術要素（セキュリティ）
第 7 部	データ構造とアルゴリズム	1 基礎理論（アルゴリズムとプログラミング）

なお、本書でカバーしていない科目A出題範囲の分野（テクノロジ系の一部、及びマネジメント系、ストラテジ系）については、当社発行の「IT戦略とマネジメント」で学習できるようになっている。「IT戦略とマネジメント」の各部と基本情報技術者試験の出題分野との対応関係は、次のとおりである。

IT戦略とマネジメント		基本情報技術者試験の出題分野
第1部	企業と法務	1 基礎理論（基礎理論〔応用数学など〕）
		9 企業と法務（企業活動）
		9 企業と法務（法務）
第2部	経営戦略	8 経営戦略（経営戦略マネジメント）
		8 経営戦略（技術戦略マネジメント）
		8 経営戦略（ビジネスインダストリ）
第3部	情報システム戦略	7 システム戦略（システム戦略）
		7 システム戦略（システム企画）
第4部	開発技術	4 開発技術（システム開発技術）
		4 開発技術（ソフトウェア開発管理技術）
第5部	プロジェクトマネジメント	5 プロジェクトマネジメント（プロジェクトマネジメント）
第6部	サービスマネジメント	6 サービスマネジメント（サービスマネジメント）
第7部	システム監査と内部統制	6 サービスマネジメント（システム監査）

以上のように、基本情報技術者試験の科目A問題については、本書と併せて「IT戦略とマネジメント」を活用することにより、出題分野をマスタできるように編成している。

また、本書は共通キャリア・スキルフレームワークのレベル2に対応しているので、学習内容にはレベル1のITパスポート試験の出題内容も含まれている。ITパスポート試験は、ITを利活用する社会人に求められる基礎知識を習得している全ての社会人のための試験である。本書と「IT戦略とマネジメント」を利用することで、ITパスポート試験、基本情報技術者試験と順番に受験していくことも可能となる。目標とする資格の取得を目指して、自身のスキルアップに本書を役立てていただきたい。

なお、本書では、情報処理技術者試験での出題傾向を分析して、試験合格に向けての学習の目安となるように、キーワードの重要度を次のフォントを用いてA～Cでランク付けしている。

ランク	フォント	意味
A	キーワード	重要度が高く、確実に習得することが求められる用語
B	キーワード	重要度はAに比べてやや低い、理解してもらいたい用語
C	キーワード	試験に限らず、情報処理を学ぶ上で役立つ用語

注記：“**フォント**”は、枠で囲んだ部分（欄）の〔 〕，【 】でくくった見出し、及び記述の要約や他の文章と区別する部分などに使用しており、キーワードを示すものではない。

第1部

ハードウェア

1 コンピュータの基本構成

私たちのまわりではさまざまな種類のコンピュータが活躍している。しかし、コンピュータ自体が登場してから、まだ100年も経っていない。コンピュータの歴史を、使用されている論理素子（演算などに利用されるコンピュータを構成する部品）で世代ごとにまとめると、次のようになる。

世代	代表的なコンピュータ・状況
第1世代 (1940年代～)	【論理素子：真空管】 ・世界最初のコンピュータ ENIAC (1946年) ・ストアプログラム方式^{注1}を採用した EDSAC (1949年)
第2世代 (1950年代～)	【論理素子：トランジスタ（半導体）】 ・商用コンピュータ UNIVAC I (1951年) ・コンピュータの小型化が進み、信頼性も向上
第3世代 (1960年代～)	【論理素子：IC（Integrated Circuit；集積回路）】 ・汎用コンピュータ^{注2} IBM/360 [System/360, S/360] (1964年) ・コンピュータの小型化・高速化が飛躍的に向上
第3.5世代 (1970年代～)	【論理素子：LSI（Large Scale Integration；大規模集積回路）】 ・工業用ロボットなどに取り付ける制御用コンピュータ ・家電製品などに取り付けるマイクロプロセッサ ・高速演算を可能とするスーパーコンピュータ（スパコン） ・個人での利用を目的としたマイクロコンピュータ（マイコン）
第4世代 (1980年代～)	【論理素子：VLSI（Very LSI；超大規模集積回路）】 ・「会社・施設などに1台」から「個人に1台」の時代へと変化 ・ネットワーク環境も整備され、ネットワークに接続する形態が普及
次世代 (？年代～)	【論理素子：？】 ・コンピュータの更なる小型化・高性能化 ・推論機能をもったコンピュータ ・省エネルギーに配慮したコンピュータ など

注1：ストアプログラム方式（プログラム内蔵方式，プログラム格納方式）

処理する内容をプログラムとしてコンピュータ内部に記憶させて実行する方式で、
提唱者の名前からノイマン型コンピュータともいう。

注2：汎用コンピュータ（汎用機）

どんな処理内容でも実行できるコンピュータで、基幹業務などに使用される大型
汎用コンピュータをメインフレームと呼ぶこともある。これに対し、使用目的が限
定されたコンピュータを、専用コンピュータ（専用機）と区別することもある。

しかし、コンピュータの世代は変わっても、コンピュータの基本的な構成は、登場した当初から現時点まで、ほとんど変わっていない（これから変わっていく可能性はある）。この章では、現在、利用されている代表的なコンピュータの種類と、コンピュータの基本構成である五大装置について学習する。

1-1 コンピュータの種類

コンピュータには、利用目的や形状によってさまざまな種類がある。ここでは、代表的なコンピュータの種類について学習する。

(1) パーソナルコンピュータ（パソコン）

パーソナルコンピュータは、個人での利用を目的としたコンピュータである。一般的には、“Personal Computer”の頭文字をとって**PC**と呼ばれることが多い。

PCは、その形状によって、机の上において使用する据置型の**デスクトップPC**と、持ち運びに便利なA4サイズ（又はB5サイズ）の**ノートPC**に分類される。なお、ノートPCよりも一回り大きい**ラップトップPC**や、見た目はPCであるが、より高性能・高機能な業務用・研究用の**ワークステーション**もある。



デスクトップPC



ノートPC

(2) サーバ

サーバは、複数の利用者に対し、さまざまなサービスを提供するコンピュータである。一般的にはPCよりも高機能なサーバ用のコンピュータのことを意味するが、PCをサーバとして使用することも可能である。

サーバも、その形状によって、PCよりもやや大型の**タワー型サーバ**や、専用のラック（サーバラック）に設置する**ラックマウント型サーバ**などに分類される。また、サーバの機能をもつブレードを、一つの筐体に差し込んで使用する**ブレード型サーバ**などもある。ブレード型サーバは電源装置や外部インターフェースなどを共有できるので、ラックマウント型サーバよりも高密度化、省スペース化、省電力化を実現できる。



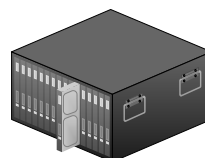
タワー型サーバ



ラックマウント型サーバ



ユニット
(サーバ)



ブレード型サーバ

(3) 携帯端末

携帯端末は、名前のとおり、携帯（持ち運び）できる端末（機器）のことであり、**モバイル端末**（モバイル機器）とも呼ばれる。

元々は、PDA（Personal Digital Assistant；**携帯情報端末**）という機器の略称としても使われていたが、現在では、**携帯電話**や**スマートフォン**、**タブレット端末**などを意味する言葉として用いられている（スマートフォンやタブレット端末など、PCと同程度の機能をもつ携帯端末を“**携帯情報端末**”と呼ぶこともある）。ほとんどの携帯端末には通信機能が内蔵されており、屋外でインターネットを利用することができる。



携帯電話



スマートフォン



タブレット端末

また、携帯端末の中で、身に付けるタイプのものを**ウェアラブルコンピュータ**（**ウェアラブル端末**）と呼ぶ。代表的なものとしては、腕時計タイプの**スマートウォッチ**や、眼鏡タイプの**スマートグラス**などがある（スマートフォン、スマートウォッチ、スマートグラス、更にはタブレット端末などを総称して、**スマートデバイス**と呼ぶこともある）。



スマートウォッチ



スマートグラス

(4) マイクロコンピュータ（マイコン）

マイクロコンピュータは、個人での利用を目的とした小さな（micro）コンピュータという意味であるが、PCが広く普及したことから、家電製品、工業用ロボットなどのさまざまな機器に組み込まれる小型コンピュータと捉えられている。

代表的なものとしては、一つのチップ（LSI）にコンピュータのすべての機能を集積した**ワンチップマイコン**（**シングルチップマイコン**）がある。スマートフォンなどに組み込まれている**SoC**（System on a Chip）も同様の機能をもつが、SoCは目的が特化されているのに対して、ワンチップマイコンにはある程度の汎用性がある（ただし、PCなどに比べると用途は限定されるため、専用コンピュータに近い）。

マイクロコンピュータには、**IC**（集積回路）よりも集積度の高い**LSI**（大規模集積回路）や**VLSI**（超大規模集積回路）が使用されている。ICは、数mm四方のチップでトランジスタ数百個分の能力をもち、LSIやVLSI（両者には厳密な基準や区別はない）も含め、ICと総称されることもある。

1-2 コンピュータの五大装置

コンピュータはさまざまな分野で利用され、いろいろな種類がある。しかし、基本的な構成要素は同じである。ここでは、コンピュータを構成する装置について学習する。

コンピュータの構成要素を説明するにあたり、まずコンピュータに求められる機能についてまとめることにする。そのためには、人間の行動と合わせて考えるのが、もっともわかりやすい。人間が「 $3+6=\square$ 」という問題を解く行動について考えると、次のようになる。

- 1) 問題を目で見て、記憶する。
↓
- 2) 「+」の意味を考える。
…二つの値を加算する記号
だとわかる。
↓
- 3) 「3」と「6」を加えた結果
「9」を記憶する。
↓
- 4) 「9」を答案用紙に書く。

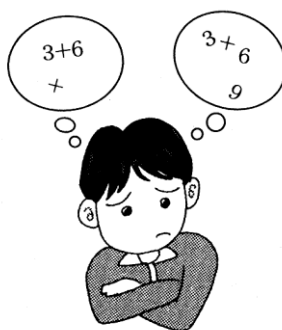


図 1-1 人間の行動

この人間の行動と同じことを、コンピュータでは五つの装置で行う。これらの装置をコンピュータの**五大装置**という。

- ① **入力装置**
コンピュータに処理させるデータを入力する装置
- ② **出力装置**
処理した結果を人間にわかる形で出力する装置
- ③ **記憶装置**
データを記録する装置（主記憶装置と補助記憶装置がある）
- ④ **演算装置**
記憶されているデータの計算や判断などを、制御装置の指示に従って
行う装置
- ⑤ **制御装置**
命令を解釈して、他の四つの装置に対して指示をする装置

以下に、具体的な例をあげて説明する。「 $3+6=$ □」の演算をコンピュータが行う場合の、各装置の働きをまとめると、次のようになる。

- 1) 入力装置から「 $3+6$ 」を入力し、記憶装置に記録する。
↓
- 2) 「 $+$ 」の意味を制御装置が解読する。解読した結果、加算の命令であることがわかるので、演算装置に指示を出す。
↓
- 3) 「 3 」と「 6 」を記憶装置から取り出し、演算装置が計算する。結果の「 9 」を記憶装置に記録する。
↓
- 4) 記憶装置の「 9 」を出力装置で出力する。

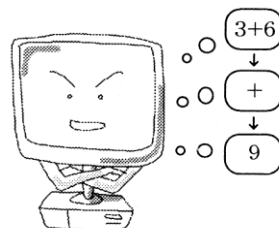


図 1-2 コンピュータの行動

また、ここでは触れていないが、入力や出力もすべて制御装置からの指示で動作する。したがって、各装置間のデータと制御（指示）の流れを図で表すと、次の図 1-3 のようになる。

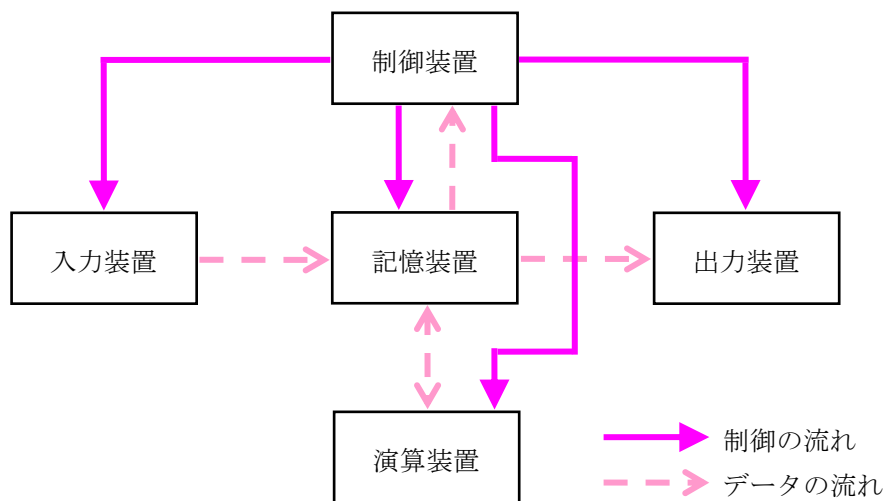


図 1-3 五大装置における制御とデータの流れ

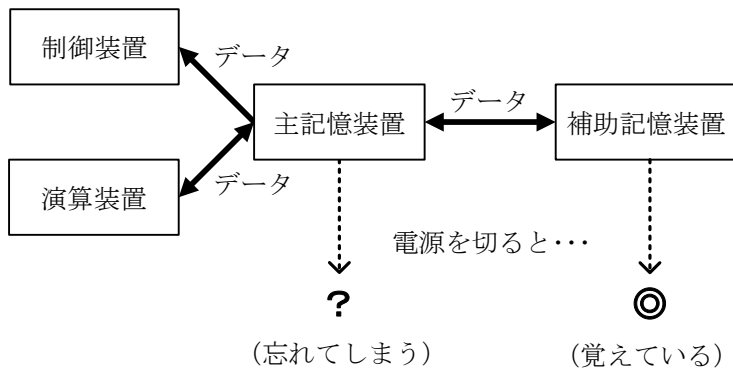
記憶装置には、主記憶装置と補助記憶装置がある（図 1-3 で記憶装置と記述してあるのは、主記憶装置である）。二つの記憶装置の違いをまとめると、次のようになる。

・ **主記憶装置**

制御装置、演算装置と直接データをやり取りできる記憶装置で、**揮発性**（電源を切ると記憶されている内容が失われる）である。

・ **補助記憶装置**

主記憶装置に入りきらないデータを記憶しておく記憶装置で、**不揮発性**（電源を切っても記憶している内容は失われない）である。



また、制御装置と演算装置をまとめて**中央処理装置**（CPU：Central Processing Unit）あるいは**処理装置**といい、中央処理装置の外にある入力装置や出力装置、補助記憶装置のことを**周辺装置**とも呼ぶ。

現在のPCや家電製品などでは、CPU（プロセッサ）の機能を、一つのLSIにまとめた**マイクロプロセッサ**（MPU：Micro Processing Unit）が利用されている。また、画像処理を高速に行う専用プロセッサ**GPU**（Graphics Processing Unit）が用いられることもある。

一方、CPUの機能だけでなくメモリなども含む、必要とされるすべての機能（システム）を集積した半導体チップ（LSI）を**SoC**（System on a Chip）という。SoCには、高速化や低消費電力というメリットがあるが、開発リスクが高いというデメリットもある。そのため、複数の半導体チップを一つのパッケージに集約する**SiP**（System in a Package）と組み合わせて用いられる場合もある。また、家電製品などに利用される**ワンチップマイコン**（**シングルチップマイコン**）では、CPUやメモリの機能だけでなく、入出力機能も一つの半導体チップに集積している。

このほかにも、プロセッサを補助する特定の処理だけを行う**コプロセッサ**や、PCなどのようにさまざまな処理を行う**汎用プロセッサ**とは逆に、一つの処理にだけ特化した**専用プロセッサ**などがある。

2 コンピュータのデータ表現

コンピュータに仕事をさせるには、あらかじめ、処理手順としてのプログラムや処理対象のデータをコンピュータ内部に記憶しておく必要がある。この章では、コンピュータ内部の情報の記録（表現）方法について学習する。

2-1 データの表現

(1) 表現単位

コンピュータ内部では、データを電気信号で記録する。電気信号は、基本的に二つの状態しか表現できない。

- ・電気が流れているか？ $\longleftrightarrow$ 流れていないか？
- ・電圧が高いか？ $\longleftrightarrow$ 低いかな？

この二つの状態に0と1を対応させ、コンピュータ内部でデータとして記録し、保存する。つまり、コンピュータが記録するデータは、0又は1で表すことになる。この0又は1を表す最小単位を**ビット**という。そして、このビットが表現する0と1を組み合わせでさまざまな意味をもたせ、データとして記録しておく。このとき、8個のビットを集めて一つの単位としたものを、**バイト**と呼ぶ。

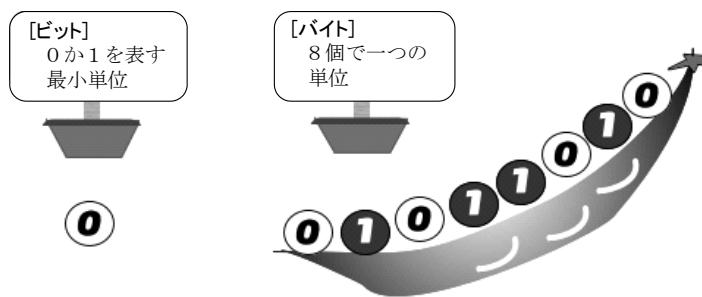


図 1-4 ビットとバイト

また、バイトよりも多くのビットを集めた**ワード**という単位がある。ワードは、コンピュータ内部での処理単位であり、使用するコンピュータの機種により16ビット、32ビット、64ビットなどがある。もちろん、多くのビットを一度に処理できるほうが一定時間で処理できる情報が多くなるため、1ワードのビット数が多いコンピュータほど処理速度が速い。なお、現在のPCでは、1ワード32ビット又は64ビットのものが多い。

(2) 情報量

1ビットで表現できる情報は2種類(0, 1)であり, これを**情報量**という。2ビットで表せる情報量は4種類(00, 01, 10, 11), 3ビットで表せる情報量は8種類(000, 001, 010, 011, 100, 101, 110, 111), …となる。

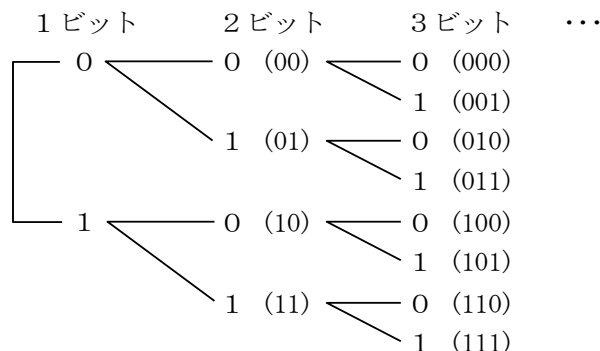


図 1-5 ビット数と情報量

図 1-5 のように, 使用するビットが一つ増えるごとに枝分かれが増えていき, 表現できる情報量は2倍ずつ増える。つまり, n ビットで表現できる情報量は 2^n 種類 (2^n は2のべき乗を表し, 2を n 回掛けるという意味である) となる。この考え方を利用して, バイトやワード(16ビット)で表現できる情報量をまとめると, 次のようになる。

☆ n ビットの情報量 = 2^n 種類

- ・ 1 バイト (= 8 ビット) で表現できる情報量
= 2^8 種類 = 256 種類 (00000000 ~ 11111111)
- ・ 1 ワード (= 16 ビット) で表現できる情報量
= 2^{16} 種類 = 65,536 種類 (0000000000000000 ~ 1111111111111111)

〔情報理論における情報量〕

情報理論の分野では, **生起確率** (ある事象が発生する確率) が $P(x)$ となる事象 x が発生したときに得られる情報量を $I(x)$ と定義する。このとき, $P(x)$ が小さいほど $I(x)$ は大きく, $P(x)$ が大きいほど $I(x)$ は小さくなる (情報量を, 事象が発生したことを知ったときの驚きと考えるとわかりやすい)。

ビットと情報量の関係で説明すると, 1 ビットは0又は1の情報のどちらかを表す。つまり, 0 又は 1 が表されるという事象 x の生起確率 $P(x)$ は $1/2$ となるので, 生起確率 $1/2$ の情報量が1ビットとなる。したがって, 情報量 n ビットとは, 生起確率 $1/2^n$ の事象すなわち 2^n 種類の情報となる。

(3) 接頭語

データを取り扱う場合、ある程度の大きさの数値はそのまま取り扱う。しかし、非常に大きな数値や小さな数値を、そのまま取り扱うのはとても大変なので、ある一定値を表す**接頭語**（補助単位）と組み合わせて表現する。

例えば、生まれたばかりの赤ん坊を抱いた看護師さんが「3,000gの元気な男の子です」と言うことはあっても、「この頃少し太って75,000gになってしまったよ」と言う人はいない。普通は「75kg」と言うはずである。この「k」が接頭語である。「k」は「 10^3 」という値を意味する接頭語で、「 $75\text{kg}=75\times 10^3\text{g}$ 」のように表すことができる。

代表的な接頭語の読み方と値を次表にまとめておく。

〔大きな数値を表すときに使われる接頭語〕

記号	読み方	10進数の値	2進数の値
k	キロ	10^3	2^{10}
M	メガ	10^6	2^{20}
G	ギガ	10^9	2^{30}
T	テラ	10^{12}	2^{40}
P	ペタ	10^{15}	2^{50}
E	エクサ	10^{18}	2^{60}

〔小さな数値を表すときに使われる接頭語〕

記号	読み方	10進数の値
m	ミリ	10^{-3}
μ	マイクロ	10^{-6}
n	ナノ	10^{-9}
p	ピコ	10^{-12}

ここで、大きな数値を表す接頭語では、2進数の値も付け加えておいた。コンピュータ内部では0, 1という二つの数字を使用した2進数で情報を表すため、コンピュータに関連する数値を表すときには接頭語も2進数に対応させるからである（2進数についての詳細は、この後で説明する）。

例えば、「1kカロリー」というデータはコンピュータに関係ないため、

$$1\text{ kカロリー}=1\times 10^3\text{ カロリー}=1,000\text{ カロリー}$$

という意味になる。一方、「1kバイト」はコンピュータに関連するデータであるから、

$$1\text{ kバイト}=1\times 2^{10}\text{ バイト}=1,024\text{ バイト}$$

という意味になる。ただし、「 $10^3\approx 2^{10}$ 、 $10^6\approx 2^{20}$ 、…」とほぼ等しい関係にあるので、最近では1kバイト=1,000バイトとして扱う場合も多い。なお、この関係を覚えておくと、計算問題を解くときなどに都合がよいこともある。

2-2 基数と基数変換

私たちが普段使用している **10 進数** は、0～9 の 10 個の数字を使用して、各桁に 10 のべき乗の重みを付けた数値である。

$$\begin{aligned}(362.9)_{10} &= 3 \times 10^2 + 6 \times 10^1 + 2 \times 10^0 + 9 \times 10^{-1} \\ &= (362.9)_{10} \quad \text{※} ()_{10} \text{ の } 10 \text{ は } 10 \text{ 進数であることを示す}\end{aligned}$$

10 進数の桁の重み (10) のことを **基数** といい、基数 n で表現される数値を **n 進数** と言う。 n 進数は 0～($n-1$) の n 個の数字を使用し、 n になると桁上がりする数値である。

(1) 2 進数

2 進数 は、0 と 1 の 2 個の数字を使用して、2 になると桁上がりする数値である。

コンピュータ内部で扱う情報の最小単位はビットで、0 と 1 の二つの数字しか使うことができないため、0 と 1 の二つの数字で数値を表現する 2 進数が利用されている。

$$\begin{aligned}(101.1)_2 &= 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} \\ &= (5.5)_{10} \quad \text{※} ()_2 \text{ の } 2 \text{ は } 2 \text{ 進数であることを示す}\end{aligned}$$

10 進数と 2 進数の対応は、次のとおりである。2 進数は一つの桁に 0 と 1 の二つの数字しか使えないため、10 進数に比べて桁数が早く大きくなる。

10 進数	0	1	2	3	4	5	6	7	8	9	10	...
2 進数	0	1	10	11	100	101	110	111	1000	1001	1010	...

図 1-6 10 進数と 2 進数の対応

(2) 8 進数

8 進数 は、0～7 の 8 個の数字を使用して、8 になると桁上がりする数値である。

$$\begin{aligned}(317.5)_8 &= 3 \times 8^2 + 1 \times 8^1 + 7 \times 8^0 + 5 \times 8^{-1} \\ &= (207.625)_{10} \quad \text{※} ()_8 \text{ の } 8 \text{ は } 8 \text{ 進数であることを示す}\end{aligned}$$

10 進数	0	1	2	3	4	5	6	7	8	9	10	...
8 進数	0	1	2	3	4	5	6	7	10	11	12	...

図 1-7 10 進数と 8 進数の対応

(5) 基数変換

基数変換とは、数値の基数を変えることである。2進数（基数2）の数値を8進数（基数8）や16進数（基数16）の数値に変換するのも基数変換である。

2進数、8進数、16進数のように基数に一定（べき乗）の関係がある場合は、複数の桁をまとめることで基数変換が行える。しかし、2進数と10進数のように基数に一定（べき乗）の関係がない場合は、計算により変換をする必要がある。

① 2進数から10進数への基数変換

2進数から10進数への基数変換は、2進数で1となっている桁の重みを加算していくことで行える（これは、各桁の重みに各桁の数字（0又は1）を乗算した結果を加算していくことと同じである）。

例. 2進数 $(1001101.101)_2$ を10進数に基数変換する。

$$\begin{array}{cccccccccccc}
 (1 & 0 & 0 & 1 & 1 & 0 & 1 & . & 1 & 0 & 1)_2 \\
 \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\
 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 & & 2^{-1} & 2^{-2} & 2^{-3} \\
 \downarrow & & & \downarrow & \downarrow & & \downarrow & & \downarrow & & \downarrow \\
 64 & & & +8 & +4 & & +1 & & +0.5 & +0.125 & = (77.625)_{10}
 \end{array}$$

② 10進数から2進数への基数変換

10進数から2進数への基数変換は、整数部と小数部に分けて行う。

整数部は、商が0になるまで2で除算を繰り返し、演算結果の余りを後ろから順に並べていくことで変換できる。

小数部は、演算結果の小数部分が0になるまで小数部分のみ次々と2倍していき、それぞれの演算結果の整数部分を前から順に並べていくことで変換できる。

例. 10進数 $(77.625)_{10}$ を2進数に基数変換する。

$$\begin{array}{lcl}
 \begin{array}{r}
 2 \overline{) 77} \\
 2 \overline{) 38} \quad \dots 1 \uparrow \\
 2 \overline{) 19} \quad \dots 0 \\
 2 \overline{) 9} \quad \dots 1 \\
 2 \overline{) 4} \quad \dots 1 \\
 2 \overline{) 2} \quad \dots 0 \\
 2 \overline{) 1} \quad \dots 0 \\
 \hline
 0 \quad \dots 1
 \end{array}
 &
 \begin{array}{c}
 \boxed{} \\
 \uparrow
 \end{array}
 &
 \begin{array}{r}
 0.625 \\
 \times 2 \\
 \hline
 1.250 \quad \dots 1 \\
 \times 2 \\
 \hline
 0.500 \quad \dots 0 \\
 \times 2 \\
 \hline
 1.000 \quad \dots 1
 \end{array}
 \end{array}$$

(1001101 . 101)₂

10進数とn進数の基数変換の手順をまとめると、次のようになる。

【n進数から10進数への基数変換の手順】

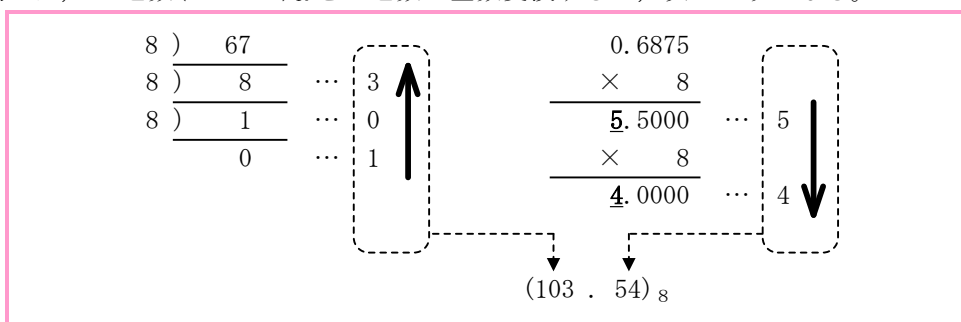
各桁の重み(nのべき乗)に各桁の数字{0～(n-1)}を乗算した結果を、すべて加算する。

【10進数からn進数への基数変換の手順】

整数部：商が0になるまでnで除算を繰り返し、演算結果の余りを後ろから順に並べる。

小数部：演算結果の小数部分が0になるまで小数部分のみを次々とn倍して、それぞれの演算結果の整数部分を前から順に並べる。

例えば、10進数 $(67.6875)_{10}$ を8進数に基数変換すると、次のようになる。

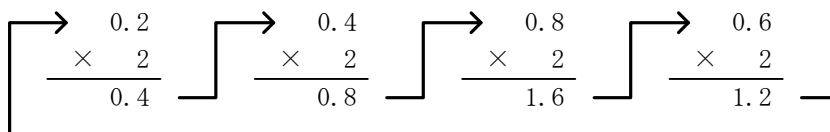


ただし、10進数を8進数や16進数に基数変換する場合には、10進数から2進数に基数変換した結果を8進数や16進数に変換するほうが簡単な場合もある。

$$\begin{aligned}
 10 \text{ 進数} : (67.6875)_{10} &= 2 \text{ 進数} : (1000011.1011)_2 \\
 &= (\underline{001} \ \underline{000} \ \underline{011} . \ \underline{101} \ \underline{100})_2 \\
 &= 8 \text{ 進数} : (1 \ 0 \ 3 . 5 \ 4)_8
 \end{aligned}$$

【有限桁数で表現できない小数】

10進数の小数部をn進数に変換するとき、演算結果の小数部分が0にならずに循環してしまい、**有限小数**に変換できないことがある。例えば、10進数 $(0.2)_{10}$ を2進数に変換すると、次のようになる。



$$(0.2)_{10} = (0.0011001100110011\cdots)_2$$

このように、基数変換した結果が**無限小数**又は**循環小数**になってしまう場合、コンピュータ内部では数値を近似値で扱う。

2-3 データの表現形式

コンピュータ内部では、すべてのデータを0又は1で表すが、その表現形式は次のように分類することができる。

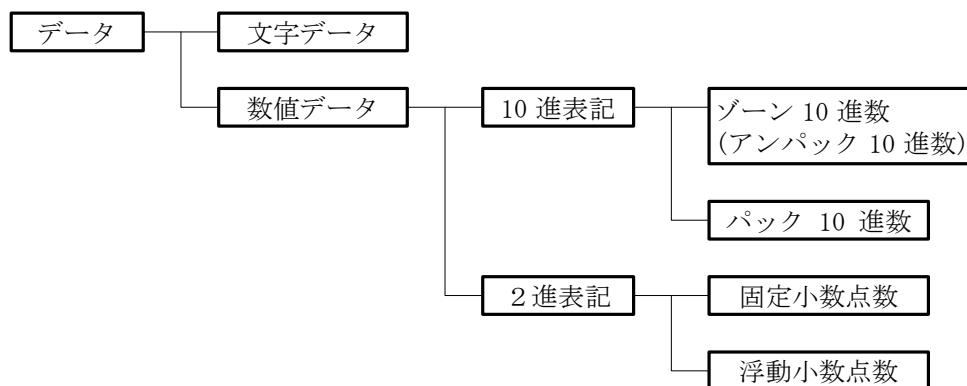


図 1-9 データの表現形式

2-3-1 文字データ

コンピュータ内部では、0と1の組合せで文字データを表現する。開発初期のコンピュータでは、8ビット（1バイト）のビットパターンに1文字を対応させていた。また、文字に対応させたビットパターンのことを**文字コード**と呼び、代表的なものとして次の六つがあげられる。なお、文字コードが異なるコンピュータ間でデータをやり取りすると、元のデータとは異なった文字が表示される**文字化け**が起きる可能性がある。

(1) ASCII コード

ASCII コードは、アメリカの規格団体 **ANSI** (American National Standards Institute ; 米国規格協会) が 1962 年に制定した文字コードである。アルファベットや数字などを表す符号ビット（7ビット）と、誤りを検査するパリティビット（1ビット）の8ビットで構成されている。PCなどで使用されるが、日本の文字（漢字や仮名など）に関する規定はない。

(2) ISO コード

ISO コードは、標準化機関 **ISO** (International Organization for Standardization ; 国際標準化機構) が 1967 年に ASCII コードをもとに制定した7ビットの文字コードである。世界各国で使われている文字コードの基本となっている文字コードである。

(3) JIS コード

JIS コードは、JISC (Japanese Industrial Standards Committee ; 日本産業標準調査会) が、ISO コードをもとに日本特有の文字を表せるように産業標準化法に基づいて審議し、JIS (Japanese Industrial Standards ; 日本産業規格) として制定した文字コードである。

① JIS 7 単位符号／JIS 8 単位符号 (JIS X 0201)

半角カタカナを表現できるようにした文字コードである。参考までに、JIS 8 単位符号の表を図 1-10 に示す。利用方法は、コードで表したい文字を表から探し出し、列 (上位 4 ビット) と行 (下位 4 ビット) を、それぞれ上から順、左から順に並べたものが、その文字の文字コードとなる。

例. “JIS” を文字コードに変換する。

文字	表の位置	文字コード	(16 進表記)
J	… 4 列 10 行	0100 1010	(4A)
I	… 4 列 9 行	0100 1001	(49)
S	… 5 列 3 行	0101 0011	(53)

										0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

図 1-10 JIS 8 単位符号 (JIS X 0201)

② JIS 漢字コード (JIS X 0208)

平仮名や漢字を表現するために、1 文字を 2 バイト (16 ビット) で表現する文字コードである。

③ シフト JIS コード (Shift_JIS)

JIS 漢字コードを拡張したコード体系で、特殊な切替えコードを使わずに 1 バイト文字と 2 バイト文字の混在を可能にする文字コードである。

(4) EBCDIC (Extended Binary Coded Decimal Interchange Code)

EBCDIC は、アメリカの IBM 社が開発した 8 ビットの文字コードである。元々は IBM 社が自社コンピュータのために開発したコードであるが、第 3 世代 (1960 年代～) は IBM 社のコンピュータが市場の大半を占めていたため、大型コンピュータの世界で業界標準 (**デファクトスタンダード**) となった。

(5) Unicode

Unicode は、アメリカのアップル社、IBM 社、マイクロソフト社などが PC のデータ交換を円滑化するために 2 バイト系の万国統一コードとして考案／提唱した文字コードである。アルファベット、漢字や平仮名／カタカナ、ハングル文字、アラビア文字など、多くの国の文字がサポートされている。これを ISO が国際標準として規格化し、現在では **UCS-2** (2 バイト) と **UCS-4** (4 バイト) が定義されている。

また、**UTF-8**、**UTF-16**、**UTF-32** という規格 (エンコーディング方法) もある。UTF-8 は 1 バイト (8 ビット) の ASCII コードと互換性をもたせるために、UCS-2 の中で ASCII コードに含まれる文字は 1 バイトで表し、それ以外の文字は 2 バイト、3 バイト、…と 1 バイト単位で表す可変長方式である。これに対して、UTF-16 は Unicode の 1 文字を 2 バイト (16 ビット) 又は 4 バイト (32 ビット) の可変長で、UTF-32 は Unicode の 1 文字を 4 バイトの固定長で表す方式である。なお、いくつかの候補から文字コードを選択するときなどに、Unicode と UTF-8 が併記されている場合、Unicode は UTF-16 を意味している場合が多い。

(6) EUC (Extended Unix Code ; 拡張 UNIX コード)

EUC (拡張 UNIX コード) は、**UNIX** (コンピュータを制御するためのソフトウェアであるオペレーティングシステム (OS) の一つ) の国際化対応のために AT&T ベル研究所で規定された文字コードである。半角英数字は 1 バイト、漢字や仮名は 2 バイトで表現する。このとき、漢字は JIS 漢字コードに 16 進数 “80” を加えたコードになっているため、漢字部分では最上位ビットの値が “1” となっており、1 バイト英数字と漢字を区別できる。

なお、文字コードではないが、10 進数データを表現する **ゾーン 10 進数** も文字データを表現するコードの一つとして分類されることがある (ゾーン 10 進数の詳細については、次ページからの「数値データ」で説明する)。

2-3-2 数値データ

コンピュータ内部での数値表現には、大別すると二つの方法がある。一つは、数値を2進数で表現する方法で、コンピュータ内部で計算などを行うのに適している。しかし、2進数は人間になじみにくいため、人間にとってわかりやすい10進数を取り入れた表現方法も使用されている。

(1) 10進表記

人間にとってわかりやすい10進数の考え方を取り入れた数値の表現方法である。具体的には、10進数の各桁をそれぞれ4ビットの2進数に変換した**2進化10進コード (BCDコード)**を利用する。

例. $(2741)_{10}$ を2進化10進コードで表現する。

10進数	:	2	7	4	1
		↓	↓	↓	↓
2進化10進コード	:	0010	0111	0100	0001

① ゾーン10進数

10進数の1桁を、1バイトで表現する形式である。各桁に対応する1バイトの下位4ビットには2進化10進コードを格納し、上位4ビットにはゾーンビットを格納する。ただし、最下位桁の上位4ビットには符号を表す符号ビットを格納する。

ゾーンビット及び符号ビットは、コンピュータで使用される文字コードなどにより異なる。EBCDICを使用している場合、ゾーンビットに“1111 (F)”，符号ビットとして正(+)ならば“1100 (C)”，負(-)ならば“1101 (D)”を格納する。一方、ASCIIコードやJISコードを使用している場合は、ゾーンビットに“0011 (3)”を格納する。なお、符号ビットはEBCDICの場合と同じものを用いることが多い(統一された規定がないため、メーカーにより異なる)。

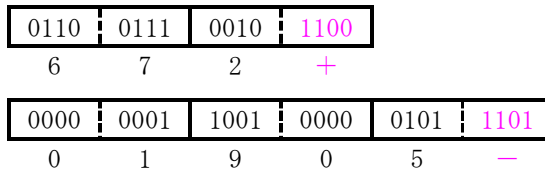
例. $(+672)_{10}$ と $(-1905)_{10}$ をゾーン10進数 (EBCDIC) で表現する。

1111	0110	1111	0111	1100	0010		
6	7			+	2		
1111	0001	1111	1001	1111	0000	1101	0101
1	9			0	-	5	

② パック 10 進数

10 進数の 2 桁を、1 バイトで表現する形式である。各桁を 2 進化 10 進コードで表すのはゾーン 10 進数と同じであるが、符号を最下位の 4 ビットで表し、バイト単位にならない場合は 0 を挿入してバイト単位にする点が違っている。

例. $(+672)_{10}$ と $(-1905)_{10}$ をパック 10 進数で表現する。



ゾーン 10 進数とパック 10 進数を比較すると、パック 10 進数はゾーン 10 進数のゾーンビット（上位 4 ビット）を省略して詰める（パックする）ことにより、少ないバイト数で多くの情報（桁）を表現できるようになっている。また、パック 10 進数をゾーン 10 進数で表現することをアンパックというため、ゾーン 10 進数のことを別名アンパック 10 進数とも呼ぶ。

ゾーン 10 進数は計算には使用できないが、パック 10 進数は使用できる。しかし、入力装置から 10 進数で入力する場合や、出力装置で 10 進数を出力する場合には、入出力単位（数値 1 桁）と情報単位（1 バイト）が対応するのでゾーン 10 進数のほうが好ましい。そこで、コンピュータ内部でゾーン 10 進数とパック 10 進数の変換が行われている。

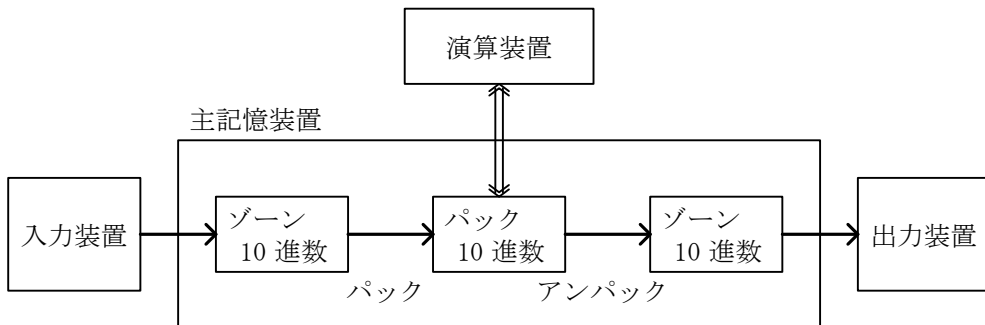


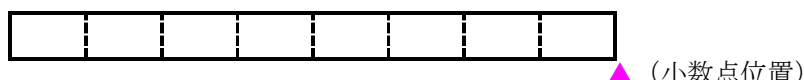
図 1-11 コンピュータ内部におけるゾーン 10 進数とパック 10 進数の変換

(2) 2進表記

コンピュータ内部で取り扱いやすい2進数として数値を表現する形式には、整数部と小数部の桁数をあらかじめ決めておく**固定小数点数**と、表現する数値に合わせて小数点の位置を変える**浮動小数点数**がある。

① 固定小数点数

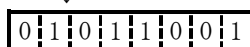
小数点を特定の位置に固定して数値を扱う表現形式である。一般的には、小数点の位置を最下位ビットに固定して、整数を扱うのに使用することが多い。



この表現形式では、数値を2進数に変換したものを、小数点の位置に合わせてそのまま使用する。

例. $(89)_{10}$ を8ビットの固定小数点数で表現する。

$$(89)_{10} = (1011001)_2$$



固定小数点数では、負数の表し方（**負の数の表現方法**）が重要となる。代表的な方法としては、次の二つがあげられる。

a : 符号付き絶対値表現

最上位の1ビットを符号ビットとして使用し、正ならば0，負ならば1を格納する方式である。残りのビットには、2進数のビット列をそのまま格納する。

例. $(+89)_{10}$ と $(-89)_{10}$ を8ビットの固定小数点数（符号付き絶対値）で表現する。

$$(89)_{10} = (1011001)_2$$

$$(+89)_{10} : \boxed{0} \boxed{1} \boxed{0} \boxed{1} \boxed{1} \boxed{0} \boxed{0} \boxed{1}$$

$$(-89)_{10} : \boxed{1} \boxed{1} \boxed{0} \boxed{1} \boxed{1} \boxed{0} \boxed{0} \boxed{1}$$

b : 補数表現

負の数を表す場合、正の数の補数をとる方式である。

〔補数〕

ある数を基準値として、その基準値に対していくつ足りないかを表す数である。
 n 進数には、 n の補数と $n-1$ の補数がある。

- ・ $n-1$ の補数 … 基準値は同一桁の最大値
- ・ n の補数 … 基準値は1桁上の最小値

補数は、基準値から補数を求めたい数を減算して求めることができる。

例1. 10進数 $(614)_{10}$ に対する補数

・ 9 の補数

$$\begin{array}{r} 999 \\ - 614 \\ \hline 385 \end{array}$$

・ 10 の補数

$$\begin{array}{r} 1000 \\ - 614 \\ \hline 386 \end{array}$$

例2. 2進数 $(01101101)_2$ に対する補数

・ 1 の補数

$$\begin{array}{r} 1111111 \\ - 01101101 \\ \hline 10010010 \end{array}$$

・ 2 の補数

$$\begin{array}{r} 100000000 \\ - 01101101 \\ \hline 10010011 \end{array}$$

例. $(+89)_{10}$ と $(-89)_{10}$ を8ビットの固定小数点数（2の補数）で表現する。

$$(89)_{10} = (1011001)_2 \quad \rightarrow \quad \begin{array}{r} 100000000 \\ - 01011001 \\ \hline 10100111 \end{array}$$

$$(+89)_{10} : \boxed{0} \boxed{1} \boxed{0} \boxed{1} \boxed{1} \boxed{0} \boxed{0} \boxed{1}$$

$$(-89)_{10} : \boxed{1} \boxed{0} \boxed{1} \boxed{0} \boxed{0} \boxed{1} \boxed{1} \boxed{1}$$

一般に固定小数点数では、以下の理由により2の補数表現を用いることが多い。

1. 減算を加算に置き換えることができる

符号付き絶対値表現を用いた場合は、演算対象の数値の先頭ビットを判断して加算や減算を使い分けなければならない。しかし、2の補数表現を用いた場合は、何も判断せずに計算できる。これを利用することで、減算も加算で表現することができる。

例. $(15)_{10} - (10)_{10}$ を $(15)_{10} + (-10)_{10}$ として計算する。

① $(-10)_{10}$ を 2 の補数で表現する。

$$\begin{array}{r} (10)_{10} = (1010)_2 \quad \rightarrow \quad 100000000 \\ - \quad 00001010 \\ \hline 11110110 \end{array}$$

$(-10)_{10}$: 1 1 1 1 0 1 1 0

② $(15)_{10}$ と 2 の補数表現にした $(-10)_{10}$ を加算する。

$$\begin{array}{r} (15)_{10} = (1111)_2 \\ (+15)_{10} \quad \text{0 0 0 0 1 1 1 1} \\ \hline \begin{array}{r} 00001111 \quad \cdots (+15)_{10} \\ + \quad 11110110 \quad \cdots (-10)_{10} \\ \hline 1 \quad 00000101 \quad \cdots (00000101)_2 = (+5)_{10} \end{array} \end{array}$$

↑
※あふれた 1 は切り捨てる

この考え方を使用すると、コンピュータには加算用の機構と補数を求める機構があれば、減算用の機構は必要なくなる。また、乗算は加算、除算は減算の繰返して実現すれば、四則演算（加減乗除）を加算だけで実現できる。

2. 表現できる数値の範囲が広い

符号付き絶対値表現を用いると、“+0”と“-0”という2種類の0が発生するが、2の補数表現を用いた場合は1種類である。同一ビット数の情報量は同じであるため、2の補数表現のほうが表現できる数値の範囲が広がる。

符号付き絶対値	
01111111	+127
01111110	+126
⋮	⋮
00000000	+0
10000000	-0
⋮	⋮
11111110	-126
11111111	-127

2 の補数	
01111111	+127
01111110	+126
⋮	⋮
00000000	0
11111111	-1
⋮	⋮
10000001	-127
10000000	-128

〔nビットで表現可能な数値の範囲（2の補数表現）〕

$$-2^{n-1} \sim +2^{n-1}-1$$

② 浮動小数点数

数値表現に使用できるビット数が決まっている場合、固定小数点数では表現できる範囲が限定される。したがって、非常に大きな数値を取り扱う場合には、ビット数もそれだけ多くしなければならない。しかし、実際にデータの記録に使うビット数を無制限に増やすわけにはいかない。そこで、非常に大きな値あるいは小数点以下の非常に小さな値を表現する方法として、浮動小数点数を使用する。

コンピュータ内部での表現を説明する前に、10進数を利用して浮動小数点数の仕組みを説明する。今、次のような数値があったとする。

+456,000,000,000

説明をわかりやすくするために、符号や一つの数字を記録するのに1単位を使用するとき、この数値をそのまま記録するには、13単位必要となる。

+	4	5	6	0	0	0	0	0	0	0	0	0

ここで、数値を次のように変換して、必要な情報だけ記録する方式に変更すると、全部で6単位で表現できることになる。

$$\begin{aligned}
 &+456,000,000,000 \\
 &= +0.456 \times 10^{12} \\
 &= (-1)^0 \times 0.456 \times 10^{12} \qquad \text{※}(-1)^0 = +1, (-1)^1 = -1
 \end{aligned}$$

符号：数値が正か負かを表現する

0

仮数：小数点以下に変換された数値

4	5	6

指数：仮数に10の何乗を掛けるか

1	2

これが浮動小数点数の考え方である。この考え方を使用することで、非常に大きな数値から非常に小さな数値までを、同じ桁数で表現することができる。

なお、指数で何乗かを表現する際に使用している“10”のことを**基数**という。今の説明では10進数であったため、小数点を移動するのにわかりやすい“10”を使用した。が、コンピュータ内部では2進数を使用するため、“2”又は“16”を使用することになる。

【浮動小数点数の基本形】

$$(-1)^{\text{符号}} \times \text{仮数} \times \text{基数}^{\text{指数}}$$

実際にコンピュータ内部で使用されている浮動小数点数の表現形式は、機種などによりさまざまである。ここでは、IEEE 754 形式として規格化されている**単精度浮動小数点数**（32 ビット形式）と**倍精度浮動小数点数**（64 ビット形式）を説明する。なお、倍精度浮動小数点数のほうが、扱える数値の範囲も広く精度も高くなる。

〔単精度浮動小数点数（32 ビット形式）〕

符号部 (1 ビット)	指数部 (8 ビット)	仮数部 (23 ビット)

〔倍精度浮動小数点数（64 ビット形式）〕

符号部 (1 ビット)	指数部 (11 ビット)	仮数部 (52 ビット)

浮動小数点数形式の各部における情報の表現方法にも、いくつかの種類がある。各部の代表的な表現方法は、次のとおりである。

a : **符号部** (S)

数値の符号を表すのに使用する。正の場合は0、負の場合は1である。

b : **指数部** (E)

基数に対する指数を表すのに使用する。代表的な表現方法として、次の二つの方法が利用される。

- ・ **2の補数方式**

：指数を2進数で記録し、負（－）の場合は2の補数を使用する。

- ・ **エクセス方式**

：指数に一定の値を加算（ゲタばき）した値を記録する。加算する値（**バイアス値**）は、指数部のビット数により決定される。単精度浮動小数点数で利用されるエクセス方式の例は、次のとおりである。

方式名称	対象形式 (代表例)	指数部の形式		バイアス値
		ビット数	情報量	
エクセス 127	IEEE 754 形式	8 ビット	256 種類	127
エクセス 64	IBM 形式	7 ビット	128 種類	64

c : **仮数部** (M)

小数点以下の数値を表すのに使用する。小数点以下の部分を表現するために、**正規化**しておくのが一般的である。このとき、整数部分を0にする場合と、1にする（整数部分の1を省略して記録する）場合がある。

- ・ 整数部分を0にする場合：0.101101 → 仮数部に“101101”を記録
- ・ 整数部分を1にする場合：1.011010 → 仮数部に“011010”を記録

〔正規化〕

仮数部に使用できる数字を増やして数値の精度を保つために行う操作である。
 多くの場合、小数点以下の余分な0を少なくすることにより行われる。

例えば、7桁の仮数部を使用して10進数 $(0.000123456789)_{10}$ を表現する場合、
 そのままの状態では仮数部に記録すると、次のようになる。

0	0	0	1	2	3	4

これに対して、正規化してから仮数部に記録すると、次のようになる。

正規化表現： $0.123456789 \times 10^{-3}$

1	2	3	4	5	6	7

つまり、正規化したほうがより多くの数字を表せる（有効桁数が多くなる）
 ので、数値の精度が高いということになる。

例. 10進数 $(1234.625)_{10}$ を、次に示すIEEE 754形式の単精度浮動小数点数
 数（32ビット）で表現する。

符号部（1ビット）：正を0，負を1で表す。

指数部（8ビット）：基数を2とするエクセス127方式で表す。

仮数部（23ビット）：整数部分を1とする正規化表現で表す。

- ① 10進数 $(1234.625)_{10}$ を2進数に変換する。

$$(1234.625)_{10} = (10011010010.101)_2$$

- ② ①で求めた2進数を正規化する。

$$\text{正規化表現： } +(1.0011010010101)_2 \times 2^{10}$$

- ③ 指数をエクセス127方式の2進数（8ビット）で表現する。

$$10 \text{ 乗} \rightarrow 10 + 127 = 137 \rightarrow (10001001)_2$$

- ④ 表現形式に合わせて記述する。

0	10001001	00110100101010000000000

符号部（1ビット）：0

指数部（8ビット）：10001001

仮数部（23ビット）：00110100101010000000000

2-3-3 誤差

誤差とは、実際の値とコンピュータ内部で表現する値との差のことである。コンピュータ内部で数値を扱う場合には、誤差に注意しなければならない。

例えば、 $(0.1)_{10}$ を2進数で表現してみると次のようになる。

$$(0.1)_{10} = (0.00011001100110011\cdots)_2$$

変換した結果は“0011”が無限に繰り返される**循環小数**となる。しかし、コンピュータ内部では数値表現に使えるビット数が決まっているため、循環を途中で打ち切って記録するしかない。仮に8ビットとすると、 $(0.00011001)_2$ までしか記録できない。この2進数を10進数に変換すると、 $(0.09765625)_{10}$ となる。つまり、元の数値である $(0.1)_{10}$ とは、 $(0.00234375)_{10}$ だけ違っている。これが誤差である。

(1) 丸め誤差

丸め誤差とは、コンピュータ内部で、実数を有限桁数で表すために、最小の桁より小さい部分について四捨五入、切上げ又は切捨てを行うことによって生じる誤差である。10進小数を2進小数に変換するとき、有限桁数で表現するために発生している誤差などが該当する。丸め誤差の対策としては、**単精度**（32ビット）を**倍精度**（64ビット）にすることなどにより、誤差の値をできるだけ小さくする方法などがある。

(2) 情報落ち

情報落ちとは、絶対値の非常に大きな値と非常に小さな値の演算をするときに、非常に小さな値が無視されてしまうことにより発生する誤差である。

例. $(0.10110011)_2 \times 2^{10} + (0.11010001)_2 \times 2^{-10}$ を計算する。

$$\begin{array}{r} 1011001100 \\ + \quad \quad \quad 0.000000000011010001 \\ \hline 1011001100.0000000000\underline{11010001} \\ = (0.10110011)_2 \times 2^{10} \quad \quad \quad \downarrow \\ \quad \quad \quad \quad \quad \quad \quad \text{この値が無視される} \end{array}$$

情報落ちをなるべく少なくするためには、浮動小数点数で複数の数値の加算を行うときに、すべてのデータを絶対値の昇順に並べ替え、先頭（小さいほう）から順に加算をしていくなどの工夫が必要である。

(3) 桁落ち

桁落ちとは、ほぼ等しい数値同士の演算を行うと、数値として信頼できる**有効桁数**（有効数字の桁数）が減少することにより発生する誤差である。

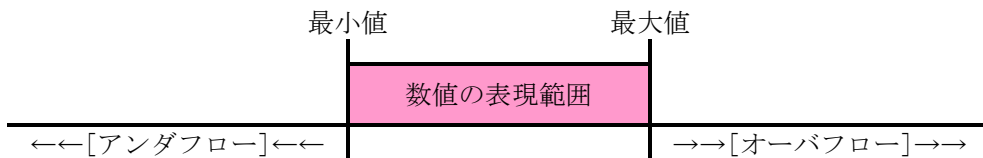
例. $(0.10110011)_2 \times 2^0 - (0.10110010)_2 \times 2^0$ を計算する。

$$\begin{array}{r} 0.10110011 \\ - 0.10110010 \\ \hline 0.00000001 \\ = (0.10000000)_2 \times 2^{-7} \end{array}$$

↓
この0は数値として信用できない

(4) オーバフロー, アンダフロー

コンピュータ内部では、限られたビット数で数値を表現するため、表現できる数値の範囲は決まっている。フロー（あふれ）とは、演算の結果がこの表現範囲を超えてしまうことにより発生する誤差である。表現範囲の最大値を超えることを**オーバフロー**といい、最小値を超えることを**アンダフロー**という。なお、単に“あふれ”といった場合は、オーバフローを意味する場合が多い。



また、これらの誤差について評価する（精度が高いか、低いか）考え方として、次の二つの指標が使用される。

- ・ **絶対誤差**：誤差そのものの大きさを評価する指標

$$\text{絶対誤差} = | \text{真値} - \text{計算値 (誤差のある値)} |$$

- ・ **相対誤差**：真値に対する誤差の割合（比）で評価する指標

$$\text{相対誤差} = \frac{| \text{真値} - \text{計算値 (誤差のある値)} |}{| \text{真値} |}$$

2-3-4 シフト演算

シフト演算とは、ビットの位置を左又は右にずらす演算である。シフト演算は、数値の演算に使用する場合と、ビットの位置を変えるために使用する場合がある。

(1) 算術シフト

算術シフトは、数値の演算をするときに使用するシフト演算である。主に、負数を2の補数で表現する固定小数点数で利用される。

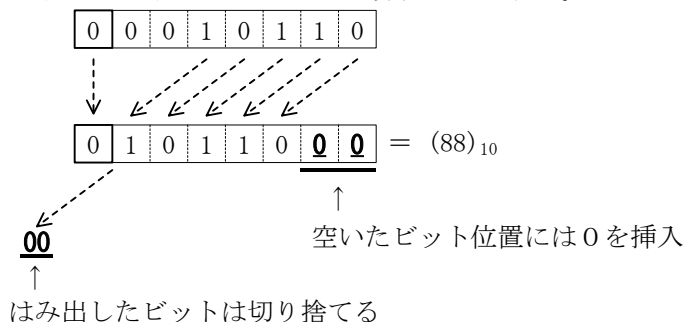
【算術シフトの法則】

- ・符号ビットはシフトしない。
- ・シフトした結果、はみ出したビットは切り捨てる。
- ・シフトした結果、空いたビット位置には次のビットを格納する。

左シフトの場合：0

右シフトの場合：符号ビットと同じもの

例. $(22)_{10} = (00010110)_2$ を左に2ビット算術シフトする。



2進数は各桁に2のべき乗の重みをもっている。したがって、同じ1という数字でも、2桁目にある1と3桁目にある1とでは意味が違ってくる。

2桁目にある1： $(10)_2 \rightarrow 2^1=2$

3桁目にある1： $(100)_2 \rightarrow 2^2=4$

このことから、算術シフトの演算法則は、次のようにまとめられる。

【算術シフトの演算法則】

- ・左にnビット算術シフトすると、元の数の 2^n 倍になる。
- ・右にnビット算術シフトすると、元の数の 2^{-n} 倍になる。
(元の数を 2^n で除算した値になる)

(2) 論理シフト

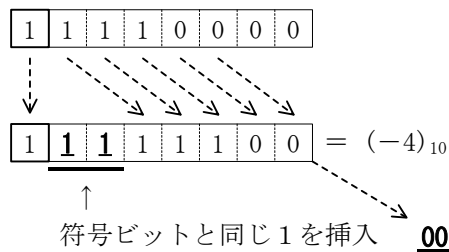
論理シフトは、ビットの位置を変えるときに使用するシフト演算である。算術シフトとの大きな違いは、符号ビットを特別扱いしないことである。

〔論理シフトの法則〕

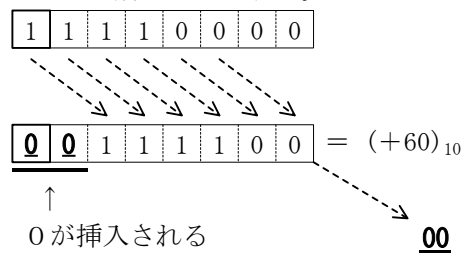
- ・符号ビットもシフトする（動かす）。
- ・シフトした結果、はみ出したビットは切り捨てる。
- ・シフトした結果、空いたビット位置には0を格納する。

例. $(-16)_{10} = (11110000)_2$ を右に2ビット算術シフトした結果と論理シフトした結果を比較する。

① 右に2ビット算術シフトする。

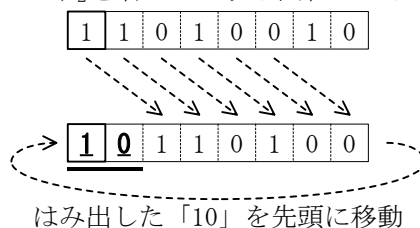


② 右に2ビット論理シフトする。



なお、論理シフトの一種に**回転シフト**（循環シフト）と呼ばれるものがある。回転シフトでは、はみ出したビットを空いた位置に循環させる。

例. $(11010010)_2$ を右に2ビット回転シフトする。



3 中央処理装置と主記憶装置

コンピュータのデータ処理は、「入力→処理→出力」という手順で行われる。この一連の手順の中で「処理」を担当する中央処理装置（CPU）は、制御装置と演算装置により構成されていることは第1章で説明した。この章では、中央処理装置のさらに詳細な動作原理と中央処理装置と関連が深い主記憶装置について学習する。

3-1 中央処理装置の構成

ストアプログラム方式のコンピュータでは、主記憶装置に記録されているプログラム（命令）を一つずつ中央処理装置に読み込み、その命令の内容により制御装置が各装置に指示を出して処理を行っている。この動作を行うため、中央処理装置はさまざまな装置、要素により構成されている。

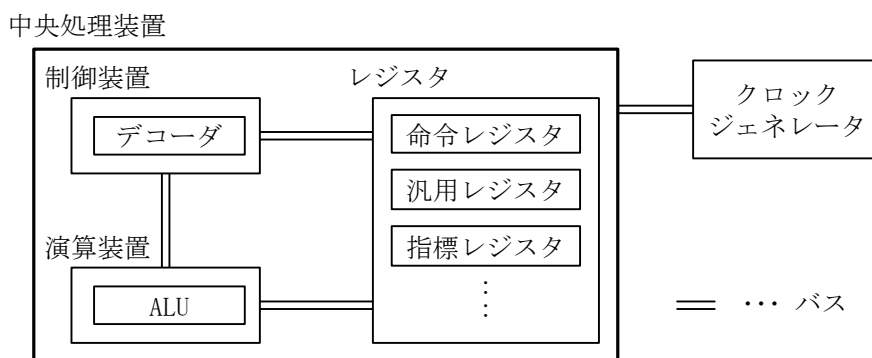


図 1-12 中央処理装置の構成要素

(1) 制御装置

制御装置は、実行する命令を解釈して各装置に指示を出す装置である。**デコーダ**（実行する命令を解釈する**命令解読器**又は**復号器**）などにより構成される。

(2) 演算装置

演算装置は、コンピュータ内部での演算を行う装置である。加算を行うための**加算器**や補数を求めるための**補数器**を用いた **ALU**（Arithmetic and Logic Unit；算術論理演算装置）などにより構成される。

(3) レジスタ

レジスタは、中央処理装置内で、さまざまなデータを一時的に記録するために使用される装置である。記録するデータの種類の合わせて、さまざまなレジスタが存在する。代表的なレジスタは、次のとおりである（各レジスタの詳細については、後に説明する）。

- ・ **命令レジスタ**

実行する命令を格納する。命令部とアドレス部により構成される。

- ・ **命令アドレスレジスタ（プログラムカウンタ**又は**プログラムレジスタ）**

次に実行する命令のアドレス（主記憶装置での格納位置）を格納する。

- ・ **汎用レジスタ**

処理対象となるデータを格納するなど、さまざまな用途に使用する。

- ・ **アキュムレータ**

演算を行うためのデータを格納する。汎用レジスタで代用することもある。

- ・ **ベースアドレスレジスタ**

プログラムの先頭アドレスを格納する。

- ・ **指標レジスタ（インデックスレジスタ）**

アドレス修飾を行うための指標を格納する。

- ・ **フラグレジスタ**

ある状態（演算結果が正か負か、など）の情報を格納する。フラグレジスタの状態により、条件付き分岐命令の分岐先などが決定される。

- ・ **PSW（Program Status Word；プログラム状態語）**

プログラムの実行状態（プログラムカウンタの値やフラグレジスタの値など）を格納する。

(4) クロックジェネレータ

クロックジェネレータは、コンピュータ内部で各装置間の動作のタイミングをとるための信号（クロック信号）を発生させる装置である。クロック信号が発生する速さを**クロック周波数**で表し、単位として **MHz**（メガヘルツ：1秒間に100万回）が用いられる。通常は信号の立上がり又は立下がりのどちらか一方を同期タイミングとして利用するが、両方を利用する**ダブルデータレート（DDR：Double Data Rate）**という技術もある。

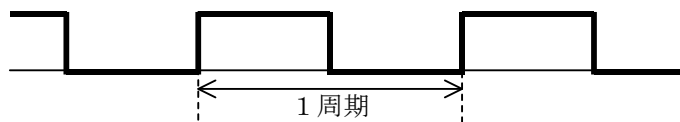


図 1-13 クロック周波数の 1 周期

(5) バス

バスとは、各種装置やレジスタをつなぎ、データや制御信号を伝送するための信号路である。バスには、データを1ビットずつ順番に送信する**シリアルバス**と、複数ビットを同時に送信する**パラレルバス**がある。バスの性能は、1回のクロック信号で送信できるビット数（**バス幅**）とクロック信号など、バスごとに定められた**アクセスモード**で決まる。

〔接続先による分類〕

- ① **内部バス**（CPU 内部バス）
中央処理装置内で使用されるバスである。
- ② **外部バス**（CPU 外部バス）
中央処理装置と外部機器を接続するバスである。外部バスのクロック周波数は、中央処理装置のクロック周波数とは一般に異なっている。
 - ・ **システムバス**：CPU から直接外部に接続されるバスの総称
 - ・ **メモリバス**：主に、主記憶装置と接続するバス
 - ・ **入出力バス**：主に、入出力装置と接続するバス
- ③ **拡張バス**
PC 本体と拡張カードを接続するバスである。

〔用途による分類〕

- ① **アドレスバス**：主記憶装置に参照アドレスを指定するためのバス
- ② **コントロールバス**：各装置に制御装置から指示を出すためのバス
- ③ **データバス**：データをやり取りするためのバス

従来のバス（**ノイマン型アーキテクチャ**）では、命令とデータのロードに同一のバスを利用していた。しかし、現在の PC では、命令とデータのロードに、それぞれ独立したバスを使用する**ハーバードアーキテクチャ**が採用されている。

現在、PC で使用されている代表的なバスは、以下の二つである。なお、PC 以外では、さまざまなバス（メーカー固有のバスもある）が使用されている。

- ・ **PCI** (Peripheral Component Interconnect)
アメリカのインテル社が策定した外部バスの規格である。バス幅 32 ビット、バスクロック 33MHz、転送レート 132Mバイト／秒の PCI 規格が一般的である。
- ・ **PCI Express** (PCIe)
PCI-SIG が策定した、PCI に代わる外部バスの規格である。転送レートは全二重通信時 500Mバイト／秒で、伝送路を 16 本束ねた“PCI Express x16”の規格はグラフィックボードを接続する **AGP** (Accelerated Graphics Port) 規格の後継として利用されている。

3-2 主記憶装置の構成

3-2-1 記憶素子

記憶素子とは、主記憶装置やレジスタを構成する素子である。特に、ICを利用した記憶素子を**半導体メモリ**又は**ICメモリ**と呼ぶ。

〔記憶素子の分類〕

① **MOS型** (MOS: Metal Oxide Semiconductor)

集積度が高く、消費電力も少ないが、動作速度がやや低速な半導体素子である。現在は、自由電子と正孔によって電荷を運搬することで動作速度が速くなるように改良された**CMOS** (Complementary MOS) が主流となっている。

② **バイポーラ型**

動作速度が高速であるが、集積度が低く、消費電力も大きい半導体素子である。バイポーラ型記憶素子の代表例には、バイポーラトランジスタのみで構成された論理集積回路**TTL** (Transistor-Transistor Logic) がある。

(1) RAM (Random Access Memory)

RAMは、データの読み書きが自由に行えるICメモリである。電源を切るとデータが消えてしまう性質 (**揮発性**) をもつため、データの長期保存などには適していない。

・ **SRAM** (Static RAM ; 静的RAM)

動作速度は高速であるが、高価格である。記憶の仕組みとして、直前の状態を保持し続ける**フリップフロップ回路**を使用するため、情報を記録する電荷をいつまでも保ち続けられる。しかし、回路の構成が複雑になるため集積度が低く、DRAMに比べて記憶容量が小さくなる欠点がある。主として、レジスタなどに使用される。情報を保持するための消費電力は小さいが、情報を読み書きするときの消費電力は大きくなる。

・ **DRAM** (Dynamic RAM ; 動的RAM)

動作速度はやや低速であるが、記憶の仕組みが、蓄電器 (**コンデンサ**又は**キャパシタ**) により電荷を保持するという単純な回路であるため、集積度が高く低価格で大容量のものを作りやすい。しかし、時間が経つと情報を記録する電荷が薄れるため、情報の再書込み (**リフレッシュ**) が必要となる。主記憶装置に用いられる**SDRAM** (Synchronous DRAM) や**DDR SDRAM** (Double Data Rate SDRAM) などがある。

(2) ROM (Read Only Memory)

ROM は、元々はデータの読出し専用の IC メモリである。電源を切ってもデータが消えない性質（**不揮発性**）をもっている。

- ・ **マスク ROM**

利用者がデータを書き込むことができない ROM である。工場などでプログラムやデータが記録されるメモリで、その情報を参照する目的だけに利用される。

- ・ **ユーザプログラマブル ROM**

利用者がデータを書き込むことができる ROM である。書込み回数の制限や方法により、次のように分類される。

- ・ **PROM** (Programmable ROM)

利用者が一度だけ情報を書き込める ROM である。

- ・ **UV-EPROM** (UltraViolet-Erasable PROM)

紫外線を照射することにより情報を消去して、再書込みができる ROM である。

- ・ **EEPROM** (Electrically EPROM)

電氣的に情報を全部又は一部分を消去して、再書込みができる ROM である。劣化による寿命があり、書込み回数は数万～数百万回が限度である。

- ・ **フラッシュメモリ**

電氣的な操作によってブロック単位でデータを消去してから、再書込みができる半導体メモリである。フラッシュメモリは EEPROM の一種であるため、書込み回数に限度があるが、持ち運びに便利な記憶媒体として、さまざまな用途で利用されている。

3-2-2 主記憶装置の構成要素

主記憶装置は、大きく分けて三つの部分で構成されている。

- ・ **記憶部**

データを記録するメモリセル（記憶素子）が配置されている。

- ・ **読取り書込み機構**

記録領域（メモリセルの集合）にあるデータを読み書きする。

- ・ **アドレス選択機構**

指定されたアドレスを解釈して、データの記録領域を選択する。

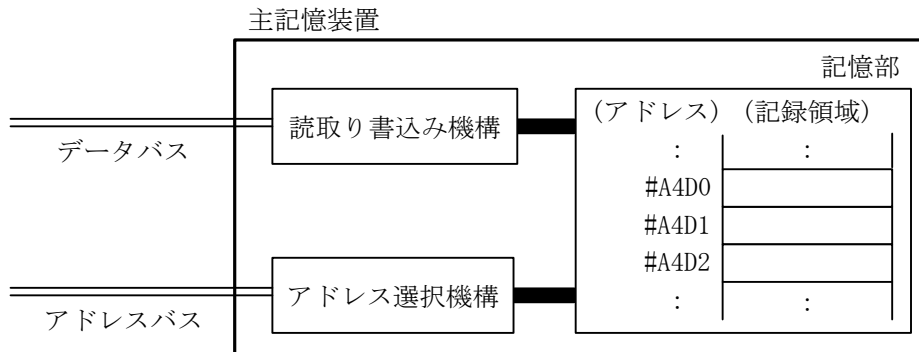


図 1-14 主記憶装置の構成要素

主記憶装置に対する**アクセス動作**は、次のように行われる。

【主記憶装置に対するアクセス動作】

- 1) アドレスバス経由で、指定アドレスがアドレス選択機構に渡される。
※アドレスバスの信号線は、アドレスのビット数分必要 [図 1-14 の例では 16 本]
- 2) アドレス選択機構がアドレスデコーダを用いて指定アドレスを解読し、アクセスする記録領域を選択する（**アドレス選択動作**）。
- 3) 選択された記録領域に対して、読取り書込み機構がデータの読み書きを行う。読取りの場合には、記録領域から読み取ったデータをデータバス経由で CPU に渡す。書込みの場合には、CPU からデータバス経由で渡されたデータを記録領域に書き込む。※データバスの信号線は、データのビット数分必要

3-2-3 主記憶装置の容量拡張

市販されている PC などでは、主記憶装置の容量はあらかじめ定められている。この容量を、さらに大きく拡張する方法として、次の二つがあげられる。

・拡張メモリ（増設メモリ）

デスクトップ PC などでは、装置や電子基盤（部品）を追加できるように準備された拡張スロットに増設するメモリである。DRAM メモリチップを小さな基盤にまとめて取り付けした **SIMM**（Single In-line Memory Module ; シム）や、**DIMM**（Dual In-line Memory Module ; ディム）がある。

・メモリカード

ノート PC の容量増設に使用される IC メモリで、フラッシュメモリを使用したものが代表的である。**JEIDA**（日本電子工業振興協会）や **PCMCIA**（アメリカの業界団体）が規格化している。現在では、ほとんど利用されていない。

3 - 3 命令とアドレッシング

3-3-1 命令の種類と構成

中央処理装置は、主記憶装置に記録されている命令を一つずつ読み込み、解釈して実行する。中央処理装置が解釈できる命令は0と1の組合せで表現された**機械語命令**であり、この命令を記述するプログラム言語（プログラムを記述する言語）を**機械語**と呼ぶ。

代表的な機械語命令の種類には、次のようなものがある。

- ・ **算術演算命令**
加算・減算などの算術演算を行う命令
- ・ **論理演算命令**
論理積演算・論理和演算などの論理演算を行う命令
- ・ **転送命令**
ロード・ストアなどのデータ転送を行う命令
- ・ **比較命令**
二つの値の大小関係などを比較する命令
- ・ **分岐命令**
フラグレジスタの値により制御を分岐（ジャンプ）する命令
- ・ **シフト命令**
算術シフト・論理シフトなどのシフト演算を行う命令
- ・ **入出力命令**
入力装置又は出力装置とデータのやり取りをする命令

また、機械語命令の構成には、主記憶装置の1語（1アドレスに対応する記憶領域）に記録される命令（**1語長命令**）と、複数の語に連続して記録される命令（2語に記録される場合を**2語長命令**という）がある。ただし、命令の長さは異なっても、実行する処理を意味する命令コードが記録された**命令部**と、処理対象となるアドレスを指示するための**アドレス部**（**オペランド部**）によって構成される点は同じである。

命令部	アドレス部 (オペランド部)

図 1-15 機械語命令の構成

なお、このアドレス部（オペランド部）の数によって、**0アドレス命令**、**1アドレス命令**、**2アドレス命令**、**3アドレス命令**と呼び分ける場合もある。

3-3-2 命令の実行手順

命令の一般的な実行手順を，図 1-16 に示す。このとき，命令を取り出すまでの処理を**命令取出し段階（フェッチサイクル）**，命令を解説して実行が終わるまでの処理を**命令実行段階（エグゼキューションサイクル）**という。

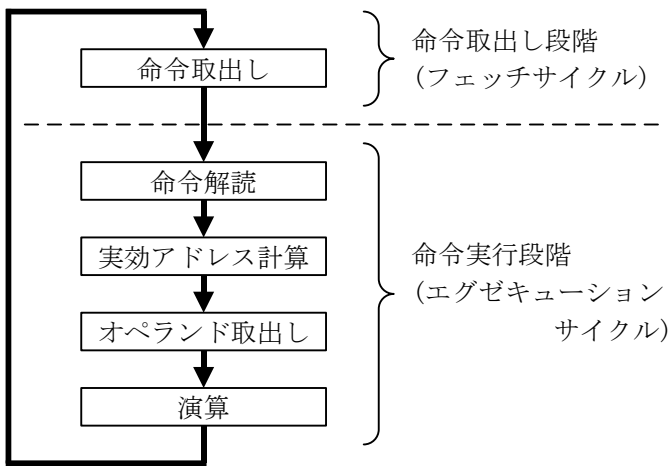


図 1-16 命令の実行手順

“命令取出し” から “演算” までの詳細な処理手順を，次の図を用いて順番に説明していく。

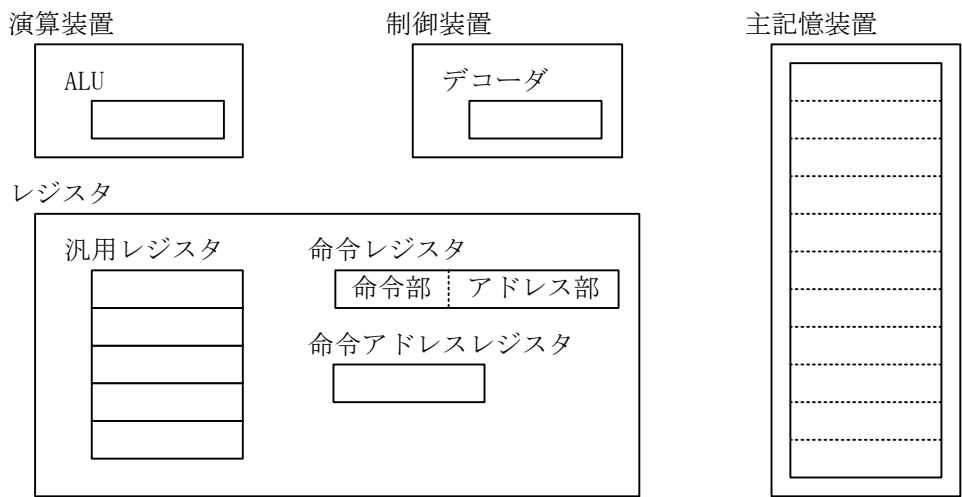


図 1-17 中央処理装置と主記憶装置

(1) 命令取出し（命令フェッチ）

命令取出し（命令フェッチ）では、制御装置からの指示により、命令アドレスレジスタが示すアドレスに格納されている命令を主記憶装置から取り出して、命令レジスタに格納する。命令取出し後、命令アドレスレジスタには、次の命令を取り出すために命令語長L（取り出した1命令が記録されていた語数）を加算する。

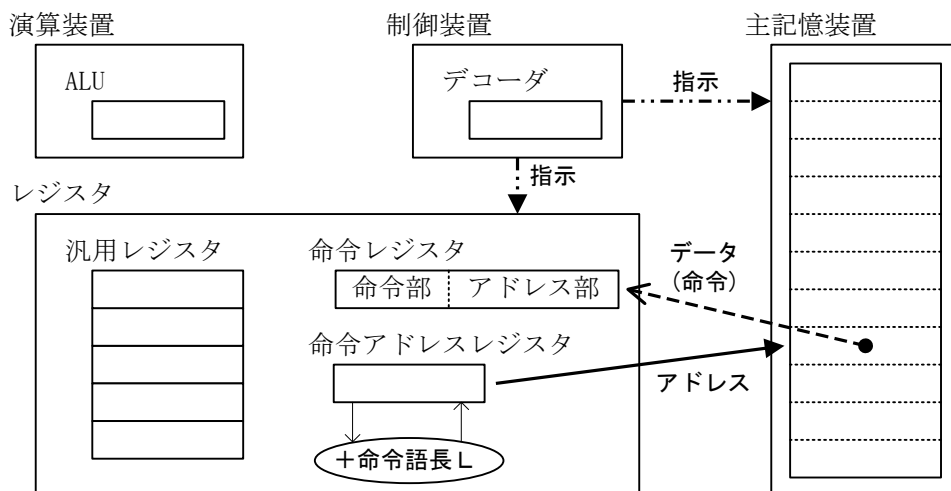


図 1-18 命令取出し（命令フェッチ）

(2) 命令解読

命令解読では、命令レジスタに取り出した命令の命令部を、制御装置のデコーダ（命令解読器）により解読する。

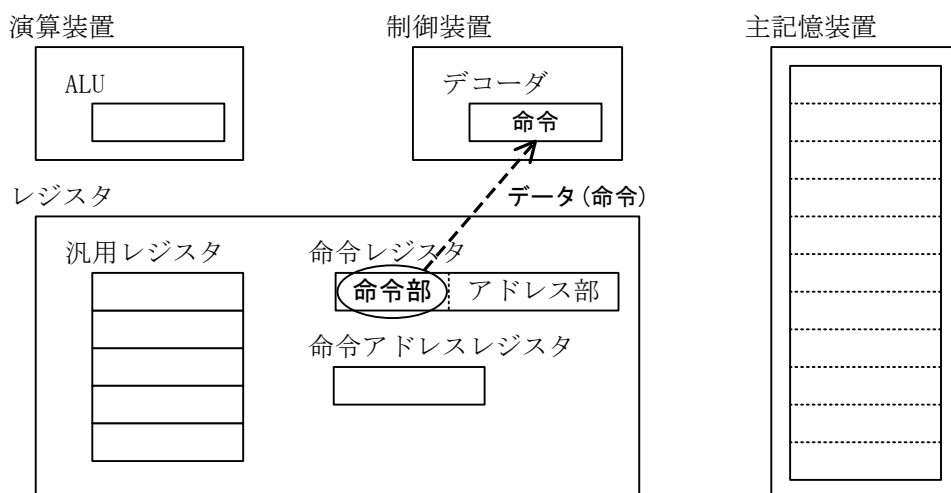


図 1-19 命令解読

(3) 実効アドレス計算／オペランド取出し

実効アドレス計算では、命令のアドレス部から主記憶装置に格納されているデータの格納位置（**実効アドレス**又は**有効アドレス**）を求める（これを**アドレス修飾**という）。

オペランド取出しでは、求めた実効アドレスを主記憶装置に送り、演算命令などの演算対象となる値や変数（**オペランド**）を汎用レジスタなどに読み込む。なお、命令の種類によってはオペランドがない場合もあり、この処理が省略されることもある。

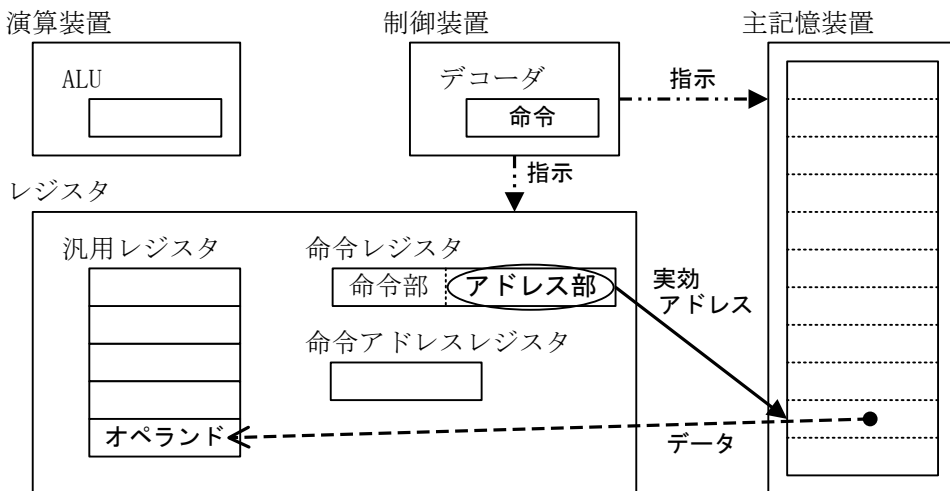


図 1-20 実効アドレス計算及びオペランド取出し

(4) 演算（命令の実行）

演算（命令の実行）では、解読された命令に従い、演算装置で演算を行う。演算結果は、汎用レジスタに記録されたり、主記憶装置に書き込まれたりする。

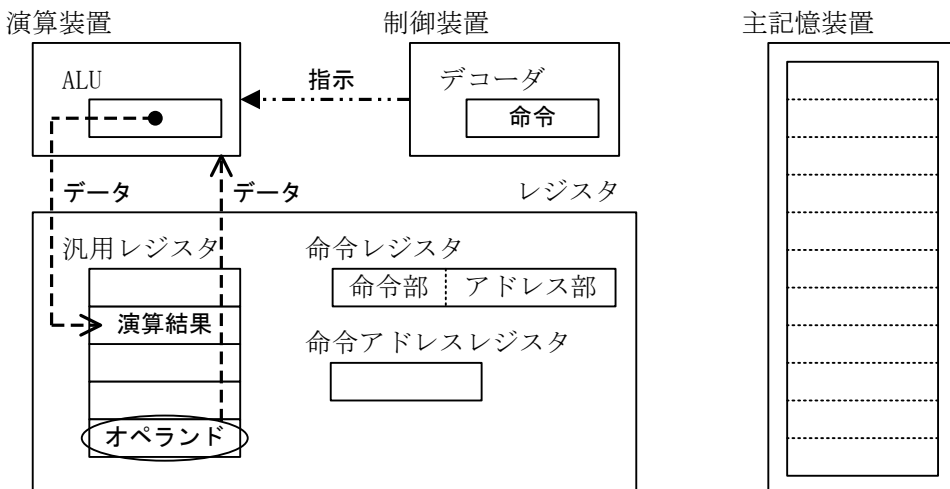


図 1-21 演算（命令の実行）

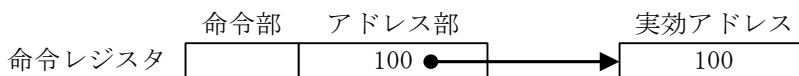
3-3-3 アドレス指定方式

アドレス指定方式とは、命令のアドレス部に記録された値から実効アドレスを求めて、オペランドを取得するための方法である。アドレス指定方式は、命令のアドレス部の値から実効アドレスを求める（**アドレス修飾**）方法と、実効アドレスからオペランドを求める方法によって分類される。

(1) アドレス部の値から実効アドレスを求める方法（アドレス修飾）

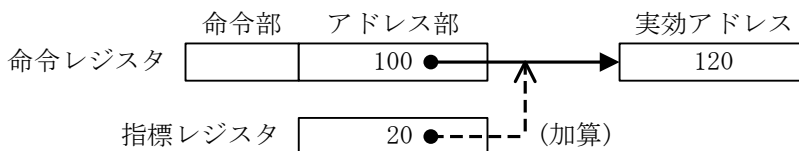
① 絶対アドレス指定方式

アドレス部の値を、そのまま実効アドレスとする方式である。



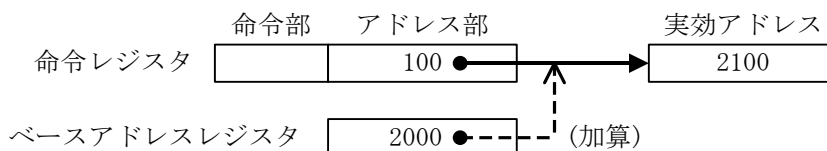
② 指標アドレス指定方式

アドレス部の値に指標レジスタ（インデックスレジスタ）の値を加算して、実効アドレスを求める方式である。指標レジスタとして、汎用レジスタを利用する場合もあり、その場合はどの汎用レジスタを使用するのかも指示する必要がある。



③ ベースアドレス指定方式

アドレス部の値にベースアドレスレジスタの値を加算して、実効アドレスを求める方式である。プログラムの先頭からの相対的な位置を意味する。



④ 相対アドレス指定方式

アドレス部の値に命令アドレスレジスタ（プログラムカウンタ）の値を加算して、実効アドレスを求める方式である。実行中の命令からの相対的な位置を意味する。



(2) 実効アドレスからオペランドを求める方法

① 即値アドレス指定方式

実効アドレスを、そのままオペランドとして使用する方式である。主記憶装置を参照しないため、実行速度が他のアドレス指定方式に比べて若干速くなる。

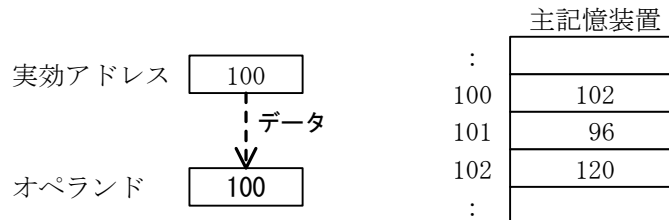


図 1-22 即値アドレス指定方式

② 直接アドレス指定方式

実効アドレスが参照する主記憶装置の内容（値）を、オペランドとして使用する方式である。一般的には、アドレス修飾に関する記述がない直接アドレス指定方式は絶対アドレス指定と考え、アドレス修飾だけが指定されている場合は直接アドレス指定と考える。

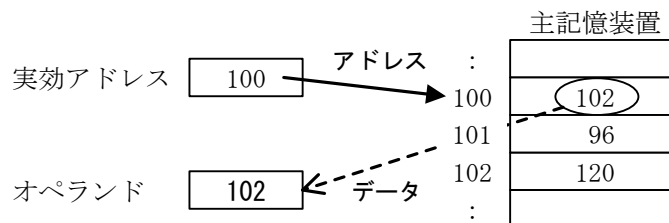


図 1-23 直接アドレス指定方式

③ 間接アドレス指定方式

実効アドレスが参照する主記憶装置の内容（値）を、オペランドのアドレスとして使用する方式である。二重（図 1-24 の例では、アドレス 120 に格納されたデータをオペランドとして取り出す）、三重の間接アドレス指定も可能である。

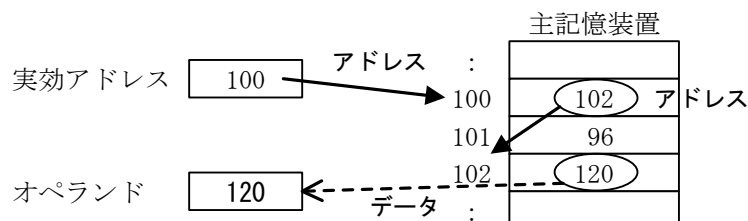


図 1-24 間接アドレス指定方式

3-3-4 割込み

割込みとは、一連の処理（命令）を実行中に別の処理（命令）を実行することである。割込み処理は、通常利用されている**ユーザモード**（CPUの使用に制限があるモード）ではなく、**特権モード**（CPUの使用に制限がないモード）で実行される。

〔割込み発生時のプロセッサの処理手順〕

- 1) ユーザモードから特権モードへ移行する。
- 2) 各種レジスタ（プログラムカウンタなど）の値を退避する。
- 3) 割込み処理ルーチン（割込みプログラム）の開始番地を決定する。
- 4) 割込み処理ルーチンを実行する。
- 5) 割込み処理ルーチンの実行が終了したら、退避していた各種レジスタの値を復元する。
- 6) 特権モードからユーザモードへ移行し、中断していた処理を再開する。

割込みは、発生する条件によって、次のように分類することができる。なお、複数の割込みが同時に発生する場合もあるため、割込みには優先順位が付けられている。

① **外部割込み**：実行中の処理とは関係のない原因で発生する割込み

- ・ **タイマ割込み**
インターバルタイマやウォッチドッグタイマなど、タイマの計測時間が指定時間を超過（タイムアウト）したときに発生する割込み
- ・ **入出力割込み**（入出力完了割込み）
入出力動作が終了したときに発生する割込み
- ・ **機械チェック割込み**
ハードウェアの誤動作や電源異常などで生じる割込み
- ・ **リスタート割込み**
利用者が外部のリスタートスイッチを押すことにより発生する割込み

② **内部割込み**：実行中の処理が原因となって発生する割込み。割出しともいう。

- ・ **SVC 割込み**（SVC：SuperVisor Call）
入出力命令の使用や記憶保護例外などで、**スーパーバイザ**（基本機能を提供するプログラム）に処理を要求したときに発生する割込み
- ・ **プログラム割込み**
実行中のプログラムのエラー（ゼロによる除算、オーバフローなど）により発生する割込み

3-4 ALUの構成回路

ALU（算術論理演算装置）は、算術演算や論理演算を行う装置である。この装置は、さまざまな回路により構成されている。

3-4-1 論理回路

論理回路とは、コンピュータが命令を論理的に処理するための論理演算を実現する回路である。論理演算とは、真(1)又は偽(0)の2通りの真理値しかもたない演算である。

〔代表的な論理演算〕

① 論理積演算 (AND 演算)

二つの入力値がともに真(1)のときに真(1)，そうでない（どちらか一方でも偽(0)の）ときに偽(0)を出力する。

② 論理和演算 (OR 演算)

二つの入力値がともに偽(0)のときに偽(0)，そうでない（どちらか一方でも真(1)の）ときに真(1)を出力する。

③ 否定演算 (NOT 演算)

入力値が真(1)のときに偽(0)を，偽(0)のときに真(1)を出力する。

④ 排他的論理和演算 (XOR 演算又は EOR 演算)

二つの入力値が同じときに偽(0)を，二つの入力値が異なるときに真(1)を出力する。

それぞれの論理演算の入力と出力の関係を表にまとめると、次のようになる。このような表のことを真理値表という。

入力		論理積 (X AND Y)	論理和 (X OR Y)	否定 (NOT X)	排他的論理和 (X XOR Y)
X	Y				
真(1)	真(1)	真(1)	真(1)	偽(0)	偽(0)
真(1)	偽(0)	偽(0)	真(1)	偽(0)	真(1)
偽(0)	真(1)	偽(0)	真(1)	真(1)	真(1)
偽(0)	偽(0)	偽(0)	偽(0)	真(1)	偽(0)

なお、論理積は“X AND Y”のほかに“ $X \cdot Y$ ”や“ $X \wedge Y$ ”，論理和は“X OR Y”のほかに“ $X + Y$ ”や“ $X \vee Y$ ”，否定は“NOT X”のほかに“ $\bar{X}$ ”や“ $\neg X$ ”，排他的論理和は“X XOR Y”のほかに“ $X \oplus Y$ ”や“ $X \nabla Y$ ”と表すこともある。

(1) 組合せ回路

組合せ回路は、そのときの入力によって出力が決定する論理回路のことである。基本的な論理演算（論理積演算，論理和演算，否定演算）を実現するための基本回路から，これらの基本回路を組み合わせることで構成される回路まで，さまざまな種類がある。

なお，論理演算を表すために使用されている回路の記号は，電子回路を表現する方式の一つで **MIL 記号**（MIL : Military Specifications and Standards）という。

① AND 回路

論理積演算（AND 演算）を行う回路である。AND 回路の真理値表と MIL 記号は，次のとおりである。

X	Y	X AND Y
0	0	0
0	1	0
1	0	0
1	1	1

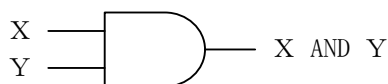


図 1-25 AND 回路

② OR 回路

論理和演算（OR 演算）を行う回路である。OR 回路の真理値表と MIL 記号は，次のとおりである。

X	Y	X OR Y
0	0	0
0	1	1
1	0	1
1	1	1

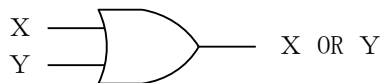


図 1-26 OR 回路

③ NOT 回路

否定演算（NOT 演算）を行う回路である。NOT 回路の真理値表と MIL 記号は，次のとおりである。

X	NOT X
0	1
1	0

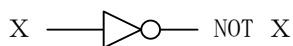


図 1-27 NOT 回路

④ NAND 回路

否定論理積演算 (NAND 演算) を行う回路である。否定論理積演算は、論理積演算の結果を否定する演算である。NAND 回路の真理値表と MIL 記号は、次のとおりである。NAND 回路は、AND 回路と NOT 回路を組み合わせた構成の回路である。

X	Y	X NAND Y
0	0	1
0	1	1
1	0	1
1	1	0

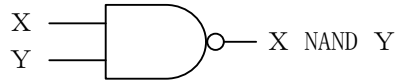


図 1-28 NAND 回路

⑤ NOR 回路

否定論理和演算 (NOR 演算) を行う回路である。否定論理和演算は、論理和演算の結果を否定する演算である。NOR 回路の真理値表と MIL 記号は、次のとおりである。NOR 回路は、OR 回路と NOT 回路を組み合わせた構成の回路である。

X	Y	X NOR Y
0	0	1
0	1	0
1	0	0
1	1	0

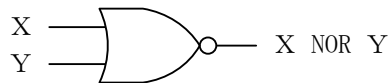


図 1-29 NOR 回路

⑥ XOR 回路

排他的論理和演算 (XOR 演算) を行う回路である。XOR 回路の真理値表と MIL 記号は、次のとおりである。

X	Y	X XOR Y
0	0	0
0	1	1
1	0	1
1	1	0

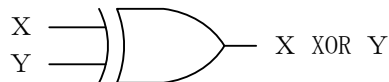


図 1-30 XOR 回路

XOR 回路も、基本回路の組合せで構成される。しかし、NAND 回路や NOR 回路のように単純な回路の組合せでは実現できない。このようなときに、目的とする結果（真理値）を得るための基本回路の組合せを考えることを**論理設計**又は**回路設計**という。

論理設計では、真理値表から得られた**論理式**（**論理関数**）を、論理演算の法則（**論理法則**）などを利用して簡素化し、最適な基本回路の組合せ方法を求める（“最適”とは、性能、効率、コストなどを考慮して設計することを意味する）。

〔代表的な論理法則〕

- ・べき等則

$$A \text{ OR } A = A, \quad A \text{ AND } A = A$$

- ・交換則

$$A \text{ OR } B = B \text{ OR } A, \quad A \text{ AND } B = B \text{ AND } A$$

- ・結合則

$$A \text{ OR } (B \text{ OR } C) = (A \text{ OR } B) \text{ OR } C$$

$$A \text{ AND } (B \text{ AND } C) = (A \text{ AND } B) \text{ AND } C$$

- ・分配則

$$A \text{ OR } (B \text{ AND } C) = (A \text{ OR } B) \text{ AND } (A \text{ OR } C)$$

$$A \text{ AND } (B \text{ OR } C) = (A \text{ AND } B) \text{ OR } (A \text{ AND } C)$$

- ・吸収則

$$A \text{ OR } (A \text{ AND } B) = A, \quad A \text{ AND } (A \text{ OR } B) = A$$

- ・ド・モルガンの法則

$$\text{NOT } (A \text{ OR } B) = (\text{NOT } A) \text{ AND } (\text{NOT } B)$$

$$\text{NOT } (A \text{ AND } B) = (\text{NOT } A) \text{ OR } (\text{NOT } B)$$

- ・その他

$$A \text{ OR } 0 = A, \quad A \text{ AND } 0 = 0$$

$$A \text{ OR } 1 = 1, \quad A \text{ AND } 1 = A$$

$$A \text{ OR } (\text{NOT } A) = 1, \quad A \text{ AND } (\text{NOT } A) = 0$$

$$\text{NOT } (\text{NOT } A) = A \quad (\text{二重否定により元に戻る})$$

【XOR 回路の論理設計】

1) 真理値表の出力が1となる部分に着目し、そのときの入力で1となる論理積演算を求める。

- ・ $(X=0, Y=1)$ のとき : $(\text{NOT } X) \text{ AND } Y$

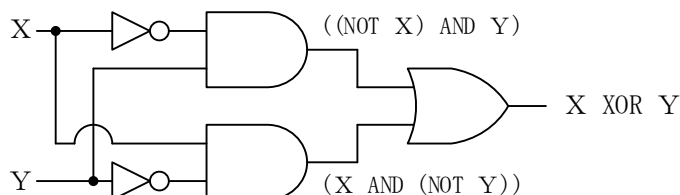
- ・ $(X=1, Y=0)$ のとき : $X \text{ AND } (\text{NOT } Y)$

2) 求めた論理積演算のどちらかが1であれば出力が1となる回路を作成するので、二つの論理積演算を論理和演算で結んだ論理式Fを求める。

$$\text{論理式 } F = ((\text{NOT } X) \text{ AND } Y) \text{ OR } (X \text{ AND } (\text{NOT } Y))$$

3) 論理式Fを表すように、基本回路を組み合わせる回路図を求める。

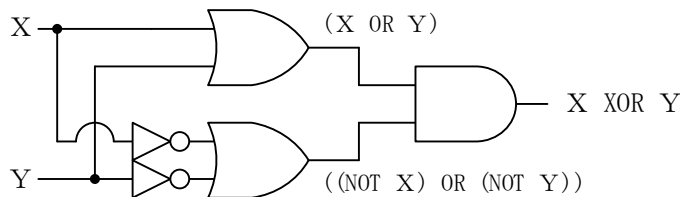
【回路図】



このとき，【XOR 回路の論理設計】で求めた論理式Fを，論理法則を利用して展開すると異なる論理回路を求めることができる。

$$\begin{aligned}
 \text{論理式 } F &= ((\text{NOT } X) \text{ AND } Y) \text{ OR } (X \text{ AND } (\text{NOT } Y)) \\
 &= (((\text{NOT } X) \text{ AND } Y) \text{ OR } X) \text{ AND} \\
 &\quad ((\text{NOT } X) \text{ AND } Y) \text{ OR } (\text{NOT } Y)) \quad \cdots \text{ 分配則} \\
 &= (((\text{NOT } X) \text{ OR } X) \text{ AND } (Y \text{ OR } X)) \text{ AND} \\
 &\quad (((\text{NOT } X) \text{ OR } (\text{NOT } Y)) \text{ AND } (Y \text{ OR } (\text{NOT } Y))) \quad \cdots \text{ 分配則} \\
 &= (1 \text{ AND } (Y \text{ OR } X)) \text{ AND} \\
 &\quad (((\text{NOT } X) \text{ OR } (\text{NOT } Y)) \text{ AND } 1) \quad \cdots A \text{ OR } (\text{NOT } A) = 1 \\
 &= (X \text{ OR } Y) \text{ AND} \\
 &\quad ((\text{NOT } X) \text{ OR } (\text{NOT } Y)) \quad \cdots A \text{ AND } 1 = A, \text{ 交換則}
 \end{aligned}$$

【回路図】



なお，実際の論理設計では，NAND 回路，NOR 回路，XOR 回路も，そのまま一つの回路（基本回路）として利用するほうが一般的である。

(2) 順序回路

順序回路は，そのときの入力と，それ以前の状態によって出力が決まる論理回路のことである。代表的な順序回路である**フリップフロップ回路**は，二つの安定状態をもつ回路であり，SRAM の記憶セルに使用されている。参考までに，図 1-31 にフリップフロップ回路の回路図と真理値表を示す。

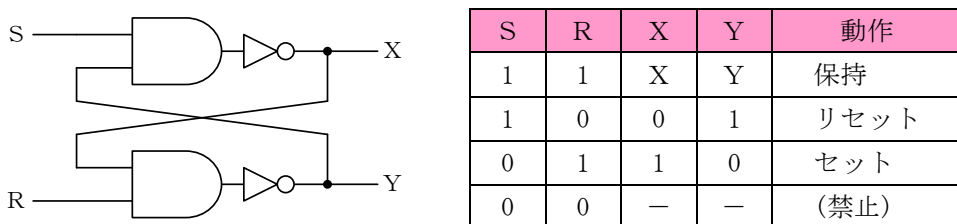


図 1-31 フリップフロップ回路の回路図と真理値表

3-4-2 算術演算回路

コンピュータ内部では、補数表現を用いることにより減算を加算で行うことができる。同様に、乗算は加算の繰返しで、除算は減算の繰返し（すなわち、加算の繰返し）で行えるため、四則演算はすべて加算のみで行えることになる。したがって、コンピュータ内部で算術演算（四則演算）を行うには、加算回路と補数回路があればよい。

まず、加算回路について考えてみる。コンピュータ内部の加算は、1ビットと1ビットを加算する2ビットの加算が基本となる。そこで、2ビットの加算を実現する回路について考える。2ビットの加算としては、次の4種類が考えられる。

$$\begin{array}{r} X: \quad \quad 0 \quad \quad 0 \quad \quad 1 \quad \quad 1 \\ Y: + \quad 0 \quad + \quad 1 \quad + \quad 0 \quad + \quad 1 \\ \hline \quad 0 \ 0 \quad \quad 0 \ 1 \quad \quad 0 \ 1 \quad \quad 1 \ 0 \end{array}$$

この2ビットの加算の演算結果を表にまとめると、次のようになる。

【2ビットの加算の演算結果】

加算する値		演算結果	
X	Y	桁上がり (c)	同桁の和 (s)
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

この表から、桁上がり（キャリー）は入力された二つの値が1のときのみ1になることから論理積演算（AND 演算）、同桁の和は二つの入力と同じ場合に0、違う場合に1になることから排他的論理和演算（XOR 演算）であることがわかる。そこで、次のような構成の回路図が導き出される。この回路を**半加算器**という。

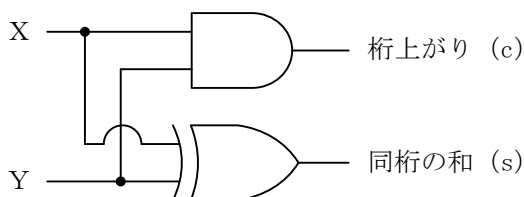


図 1-32 半加算器の回路図

コンピュータの加算回路は、この半加算器を基本として構成される。ただし、半加算器では、下位からの桁上りを考慮していないため、複数ビットで表される数値の演算は行えない。そのため、下位からの桁上りを含めた3ビットの加算回路が必要となる。

下位からの桁上がり (c0) を考慮した 3 ビットの加算の演算結果を表にまとめると、次のようになる。

〔3 ビットの加算の演算結果〕

加算する値			演算結果	
X	Y	c0	桁上がり (c1)	同桁の和 (s)
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
1	0	0	0	1
0	1	1	1	0
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

この表から回路の構成を導き出すために、桁上がり (c1) を求めるための論理式 F1 と同桁の和 (s) を求めるための論理式 F2 を考えると、次のようになる。

【桁上りを求める論理式 F1】

- 真理値表の出力が 1 となる入力で、結果が 1 となる論理積演算を求める。
 - ・ (X=0, Y=1, c0=1) のとき : (NOT X) AND Y AND c0
 - ・ (X=1, Y=0, c0=1) のとき : X AND (NOT Y) AND c0
 - ・ (X=1, Y=1, c0=0) のとき : X AND Y AND (NOT c0)
 - ・ (X=1, Y=1, c0=1) のとき : X AND Y AND c0
- 求めた論理積演算を論理和演算で結んで論理式 F1 を求める。

$$\text{論理式 F1} = ((\text{NOT X}) \text{ AND Y AND c0}) \text{ OR } (X \text{ AND } (\text{NOT Y}) \text{ AND c0}) \\ \text{OR } (X \text{ AND Y AND } (\text{NOT c0})) \text{ OR } (X \text{ AND Y AND c0})$$

【同桁の和を求める論理式 F2】

- 真理値表の出力が 1 となる入力で、結果が 1 となる論理積演算を求める。
 - ・ (X=0, Y=0, c0=1) のとき : (NOT X) AND (NOT Y) AND c0
 - ・ (X=0, Y=1, c0=0) のとき : (NOT X) AND Y AND (NOT c0)
 - ・ (X=1, Y=0, c0=0) のとき : X AND (NOT Y) AND (NOT c0)
 - ・ (X=1, Y=1, c0=1) のとき : X AND Y AND c0
- 求めた論理積演算を論理和演算で結んで論理式 F2 を求める。

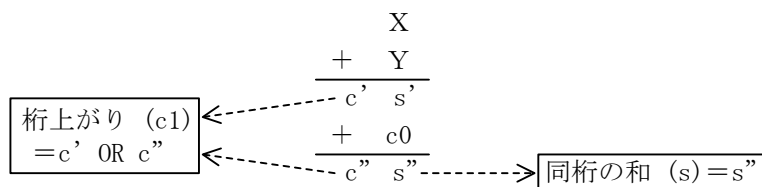
$$\text{論理式 F2} = ((\text{NOT X}) \text{ AND } (\text{NOT Y}) \text{ AND c0}) \\ \text{OR } ((\text{NOT X}) \text{ AND Y AND } (\text{NOT c0})) \\ \text{OR } (X \text{ AND } (\text{NOT Y}) \text{ AND } (\text{NOT c0})) \\ \text{OR } (X \text{ AND Y AND c0})$$

この論理式 F1 と論理式 F2 を，論理法則を利用して展開（簡素化）すると，3 ビットの加算回路を実現する回路の構成を導き出すことができる。

しかし，この方法では論理式の展開も大変で，回路の構成も複雑になると予測される。そこで，3 ビットの加算を2 ビットの加算に分解して半加算器を利用する。

【3 ビットの加算を2 ビットの加算に分解】

- 1) X と Y を加算し，桁上がり (c') と同桁の和 (s') を求める。
- 2) s' と $c0$ を加算し，桁上がり (c'') と同桁の和 (s'') を求める。ここで求められた s'' は，全体の演算における同桁の和 (s) となる。
- 3) どちらかの加算で桁上がりがあった場合，演算全体として桁上がりが発生する。そこで，全体の演算における桁上がり ($c1$) は c' と c'' の論理和演算 (OR 演算) で求める (両方の演算で桁上がりが1になることはない)。



この考え方をもとに構成した加算回路が，図 1-33 の**全加算器**である。コンピュータ内部では，この全加算器を演算ビット数分並べて，算術演算回路を構成している。

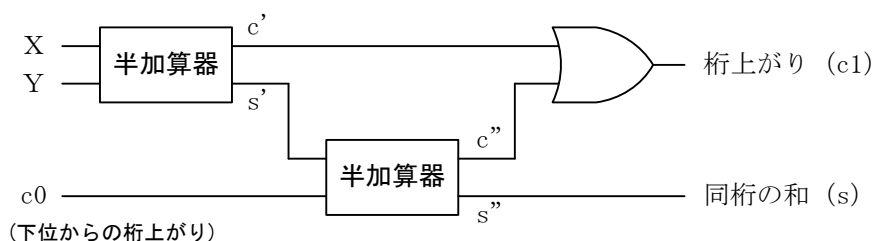


図 1-33 全加算器の回路図

次に，補数回路について考えてみる。算術演算（四則演算）を加算器（半加算器／全加算器）だけで実現するには，減算を加算で行うために補数を求める回路（**補数器**）が必要となる。コンピュータ内部では2進数の演算が行われるため，1の補数を求める**1の補数器**と2の補数を求める**2の補数器**の2種類の補数器を利用するのが一般的である。2進数における1の補数は，元のビット列を反転（0を1に，又は1を0に）することで求められるため，1の補数器はNOT回路だけで実現できる。一方，2の補数器の実現方法はいろいろあるが，1の補数器を利用して求めた1の補数に，加算器を用いて1を加算する考え方が比較的簡単な方法である。

3-5 高速化技術

高速化技術とは、各装置を応用して新たな仕組みを作ることによって、コンピュータの動作速度（処理能力）を高める技術である。

3-5-1 メモリアクセスの高速化

メモリはデータを記録する装置や媒体を意味する言葉であるが、単に“メモリ”と呼ぶ場合は主記憶装置（メインメモリ）を意味することが多い。

メモリアクセスの高速化とは、処理装置のレジスタ（SRAM）と比較して、アクセス速度が低速な主記憶装置（DRAM）からのアクセスを少しでも高速化することで、アクセス速度の違い（アクセスギャップ）によるボトルネックを解消するための技術である。

(1) 緩衝記憶装置

緩衝記憶装置とは、アクセス速度が高速な記憶装置と低速な記憶装置の間に設置する、中速の記憶装置のことである。緩衝記憶装置の一つである**キャッシュメモリ**は、高速なレジスタと低速な主記憶装置の間に設置する、SRAMで構成された中速の記憶装置である。

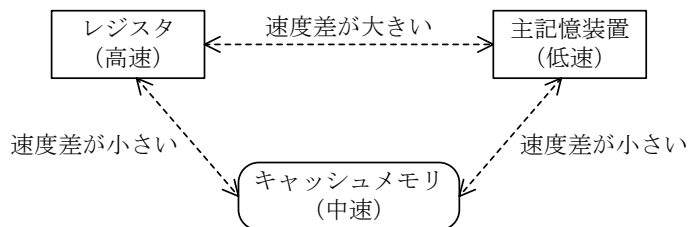
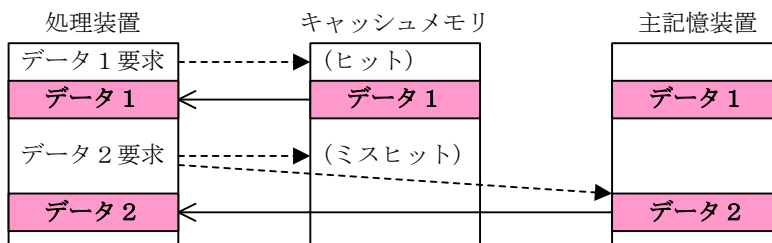


図1-34 キャッシュメモリの働き

キャッシュメモリを採用した場合、処理装置が必要とするデータがキャッシュメモリにある（ヒットした）場合はキャッシュメモリにアクセスする。一方、データがキャッシュメモリにない（ミスヒットした）場合は、主記憶装置にアクセスする（正確には、ハードウェア又はOS（コンピュータを制御するためのソフトウェア）の指示により、主記憶装置からキャッシュメモリの該当ブロックにデータが読み込まれる）。



そのため、処理装置から見た**平均アクセス時間（実効アクセス時間）**は、必要なデータがキャッシュメモリ内でどのくらい見つかるかによって決まる。したがって、キャッシュメモリに必要なデータがある確率（**ヒット率**）と、必要なデータがない確率（**NFP**：Not Found Probability）を考慮した、キャッシュメモリのアクセス時間と主記憶装置のアクセス時間の**加重平均**で求める。なお、“ヒット率+NFP=1”は必ず成立する。

例．次の処理装置の平均アクセス時間を求める。なお、必要なデータがキャッシュメモリにない確率（NFP）を0.1とする。

アクセス先	アクセス時間（ナノ秒）
キャッシュメモリ	50
主記憶装置	500

処理装置の平均アクセス時間

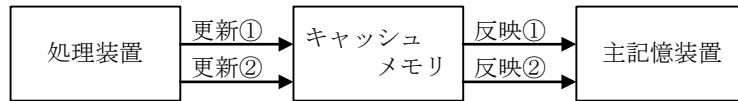
$$\begin{aligned}
 &= \text{キャッシュメモリのアクセス時間} \times \text{ヒット率} \\
 &\quad + \text{主記憶装置のアクセス時間} \times \text{NFP} \\
 &= 50 \text{ ナノ秒} \times (1 - 0.1) + 500 \text{ ナノ秒} \times 0.1 \\
 &= 95 \text{ ナノ秒}
 \end{aligned}$$

この例からは、ヒット率が高くなるほど、キャッシュメモリへのアクセスが増えて平均アクセス時間が短縮されることがわかる。命令とデータを、それぞれ独立してロードする**ハーバードアーキテクチャ**が採用されている場合、キャッシュメモリも**命令キャッシュ**と**データキャッシュ**に分離する。このとき、頻繁に実行される処理（命令）部分をまとめたプログラムを作成すると、命令キャッシュのヒット率が高くなり、平均アクセス時間を短縮する（アクセス速度が向上する）効果が期待できるようになる。

また、キャッシュメモリを利用する場合、更新（変更）されるのはキャッシュメモリ内にあるデータである。しかし、本来のデータは主記憶装置に記録されているため、キャッシュメモリ内のデータ更新を反映させる（主記憶装置に書き込む）必要がある。この書込み方式には、更新を反映させるタイミングにより次の二つがある。

- ① **ライトスルー方式**：データの更新ごとに、主記憶装置に更新を反映させる方式である。確実性は高いが、処理ごとに主記憶装置へのアクセスが発生するため、動作速度がやや遅いという欠点がある。
- ② **ライトバック方式**：キャッシュメモリ上でデータを更新している間は主記憶装置に更新を反映させず、キャッシュメモリからデータを退避するときにだけ主記憶装置に更新を反映させる方式である。動作速度は速いが、ある時点において主記憶装置のデータに不整合が発生する場合もある。

【ライトスルー方式】



【ライトバック方式】

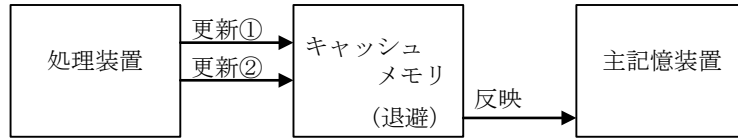


図 1-35 ライトスルー方式とライトバック方式のイメージ

なお、現在の PC では、処理装置と主記憶装置の速度差が大きいため、キャッシュメモリを **1次キャッシュ** と **2次キャッシュ** の2段階に分ける、多段階のキャッシュ構成が主流である。この場合、処理装置は1次キャッシュ → 2次キャッシュ → 主記憶装置の順にアクセスすることになる。

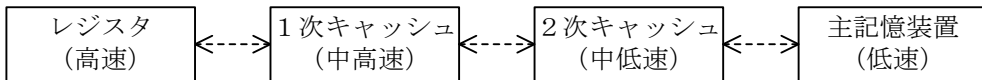


図 1-36 多段階のキャッシュ構成

一方、もう一つの緩衝記憶装置として**ディスクキャッシュ**がある。これは、主記憶装置と補助記憶装置(磁気ディスク)の間になれる緩衝記憶装置で、主記憶装置の一部を利用する方法と半導体メモリを設置する方法がある。

基本的には、記憶装置はアクセス速度が高速なものほど、記憶容量が小さくなる。そのため、図 1-37 のように高速小容量の記憶装置と低速大容量の記憶装置を組み合わせることで、全体として高速大容量の記憶装置を構成する。この考え方(記憶装置をアクセス速度と容量によって階層化したもの)を**記憶階層**という。

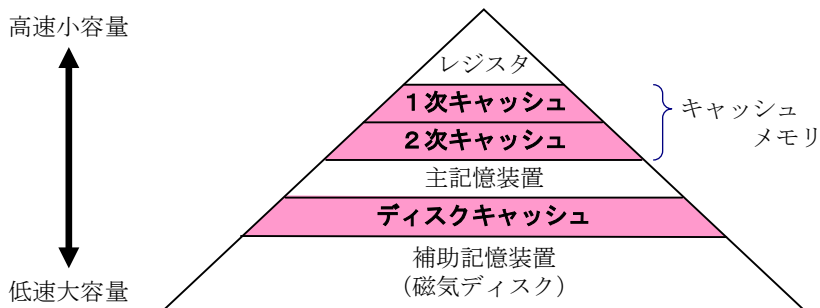


図 1-37 記憶階層

(2) メモリインタリーブ

メモリインタリーブは、主記憶装置への平均アクセス時間を改善するために、主記憶装置をいくつかの区画（バンク）に分けて構成し、並行して独立にアクセスできるようにする方式である。

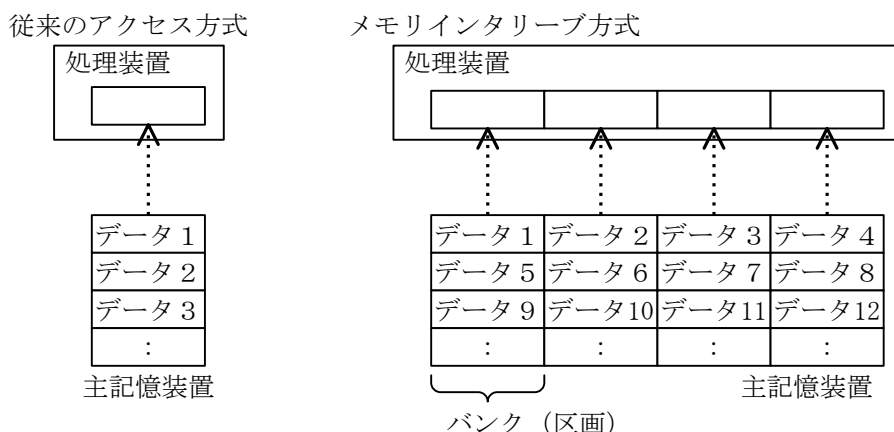


図 1-38 メモリインタリーブ方式

メモリインタリーブ方式は、データの先読みをする方式であり、主記憶装置の連続する領域にあるデータを処理するときには効果的です。ただし、データの参照は必ずしも記憶領域の順番どおりとはならないため、バンクの数と平均アクセス時間が比例するとは限りません（バンクが四つあっても、平均アクセス時間が $1/4$ にはなりません）。

3-5-2 プロセッサの高速化

プロセッサの高速化とは、処理装置（プロセッサ）の動作速度を高速化するための技術である。プロセッサ自体の性能を向上させる技術から、命令の実行手順を工夫することで見かけ上の動作速度を向上させる技術まで、さまざまな高速化の手法がある。

(1) マルチコアプロセッサ

マルチコアプロセッサとは、一つのチップ（LSI）に二つ以上のプロセッサコアを集積したものである。これに対して、一つのチップ（LSI）にプロセッサコアが一つだけ入っているものを**シングルコアプロセッサ**と呼ぶ。マルチコアプロセッサは、複数のプロセッサコアで処理を分担することで処理性能を向上させる（ n 個のプロセッサコアが集積されていれば、処理性能は約 n 倍になる）。また、シングルコアプロセッサを複数搭載する場合に比べて、消費電力を抑えながらプロセッサ全体の処理性能を高められる。ただし、複数のプロセッサコアが同時に動作すると、共有資源の競合などが発生する場合もある。

(2) パイプライン制御

パイプライン制御（パイプライン処理方式）とは、命令の実行サイクルを複数のステージ（段階）に分け、各ステージの処理を別々に実行することによって、複数の命令を少しずつ段階をずらしながら並列的に同時実行する方式である。

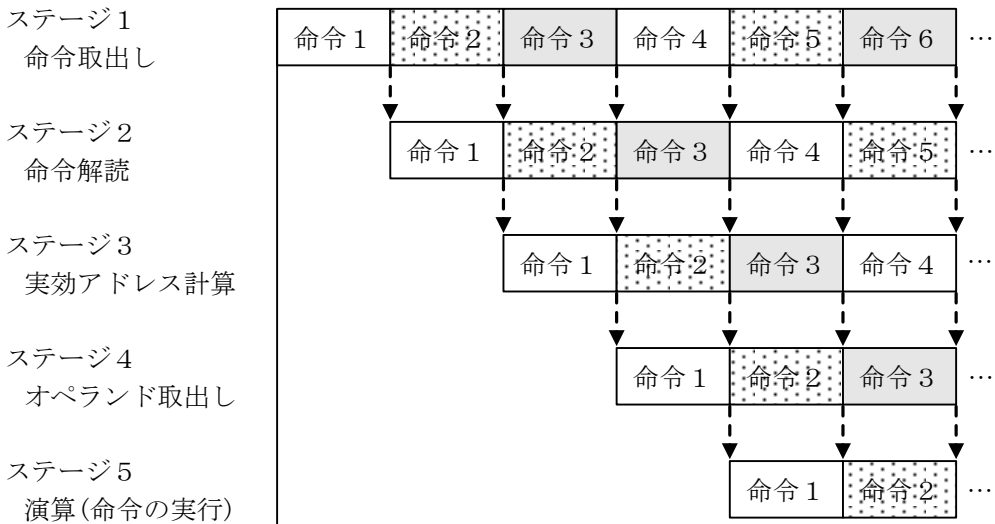


図 1-39 パイプライン制御

パイプライン制御は、命令の先行制御を行う方式であるため、実行する命令の順番が変わってしまう命令（ジャンプ命令）があると効率が悪くなる（これについては、ジャンプ命令による分岐先が決まる前に、予測した分岐先の命令の実行を開始する**投機実行**という考え方もある。ただし、予測が外れた場合は先行して実行された命令は破棄されるため、効率は悪くなる）。また、各命令の実行時間が異なる場合も、ステージの実行待ちが発生して効率が悪くなる。

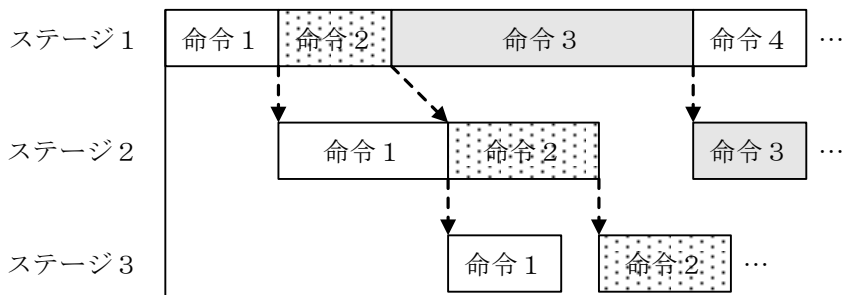


図 1-40 効率の悪いパイプライン制御

パイプライン制御を効率良く行うためには、すべての命令の実行時間をほぼ同じにする必要がある。プロセッサ（コンピュータ）のアーキテクチャのうち、この考え方に適しているものが **RISC** であり、あまり適していないものが **CISC** である。

〔プロセッサのアーキテクチャ〕

① **RISC** (Reduced Instruction Set Computer)

縮小命令セットコンピュータ。命令セット（命令の集まり）を使用頻度の高い基本命令に限定して命令語長を固定することで、単一の命令に対する命令の実行時間をできるだけ短く、統一したコンピュータである。命令の実行はハードウェア（ワイヤドロジック）により実現される。

・ワイヤドロジック制御方式

ある機能をもった論理回路に信号を流すことで、命令を実行する方式である。信号を伝達するための信号線を結んで論理を構成することから、**結線論理制御方式**ともいう。

② **CISC** (Complex Instruction Set Computer)

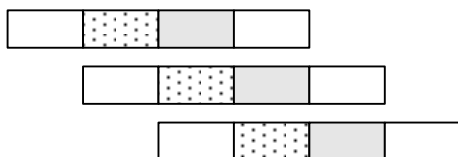
複合命令セットコンピュータ。複雑な命令をソフトウェア（マイクロプログラム）で実現するため、多機能の命令を取り扱えるコンピュータである。しかし、各命令の長さや実行時間が大きく異なるため、パイプライン制御には向いていない。

・マイクロプログラム制御方式

各種の論理回路やレジスタなどのハードウェアに対して、制御の指示を与えるものをマイクロ命令という。このマイクロ命令を組み合わせたマイクロプログラムで命令を実行する方式である。**ストアドロジック制御方式**ともいう。

パイプライン制御では、命令の実行サイクルを“命令取出し”，“命令解読”，“実効アドレス計算”，“オペランド取出し”，“演算”のような意味のあるステージに分割している。この分割を、さらに細かくしたものを**スーパーパイプライン**という。スーパーパイプラインは、ステージ数（パイプラインの深さ）を増やして、一つのステージの処理時間（パイプラインピッチ）を短くすることで、性能向上を目指す方式である。

〔パイプライン〕



〔スーパーパイプライン〕

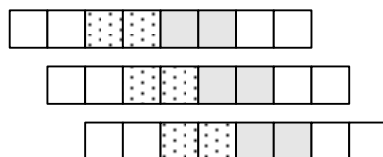


図 1-41 パイプラインとスーパーパイプラインの比較イメージ

(3) スーパスカラ／VLIW (Very Long Instruction Word)

スーパスカラと**VLIW**は、どちらも複数の命令を同時に実行することでプロセッサの性能向上を目指す方式である。

スーパスカラは、プログラムを実行する段階で、同時実行可能な複数の命令を抽出し、使用する演算器などを動的に決めながら並列実行する方式である。そのため、プログラムに記述された命令の順序と、実際に実行される命令の順序が異なることもある（このような実行方式を**アウトオブオーダー型実行**という）。

一方、**VLIW**は、プログラムをコンパイル（機械語に翻訳）して目的プログラム（機械語に翻訳されたプログラム）を生成する段階で、同時実行可能な複数の命令を一つの命令にまとめて、使用する演算器なども静的に割り当てる方式である（VLIWという名前は、複数の命令（機能）が一つの命令にまとめられ、命令語長が長くなることを表している）。VLIWでは、実行時に同時実行可能な命令を動的に選択することがないため、目的プログラムに記述された命令の順序と、実際に実行される命令の順序が異なることはない（このような実行方式を**インオーダー型実行**という）。

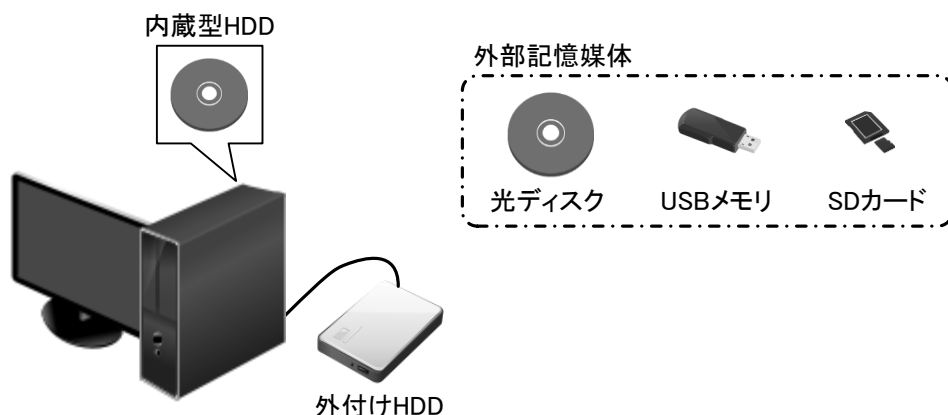
(4) 並列処理

パイプライン／スーパーパイプライン、スーパスカラ、VLIWは、プロセッサ内で命令を並列実行してプロセッサの性能向上を目指す方式である。一方、プロセッサを並列化してコンピュータの性能向上を目指すという方法もある。このような並列コンピュータのアーキテクチャは、命令の流れとデータの流の関係から4種類に分類される。

- **SISD** (Single Instruction stream Single Data stream)
単一命令で単一データを処理するコンピュータ。並列化されていない一般的なコンピュータが該当する。
- **SIMD** (Single Instruction stream Multiple Data stream)
単一命令で複数データを処理するコンピュータ。一つの命令で配列中の複数のデータを同時に演算する**ベクトルプロセッサ**（**アレイプロセッサ**）を搭載した、**ベクトルコンピュータ**などが該当する。
- **MISD** (Multiple Instruction stream Single Data stream)
複数命令で単一データを処理するコンピュータ。該当するコンピュータは実用化されていない（パイプライン制御に近い部分もあるが、厳密には異なる）。
- **MIMD** (Multiple Instruction stream Multiple Data stream)
複数命令で複数データを処理するコンピュータ。複数の処理装置を多重化した**マルチプロセッサ**などが該当する。

4 補助記憶装置

コンピュータ内部で処理されるデータは、あらかじめ主記憶装置に記録されていなければならない。しかし、主記憶装置の容量には制限があるため、すべてのデータを記録しておくことはできない。そこで、別の記憶装置にデータを記録しておき、必要なデータを主記憶装置に格納して処理する。このとき、主記憶装置に入りきらないデータを記録しておくのが、補助記憶装置の役割である。したがって、補助記憶装置は基本的には主記憶装置よりも大容量で、不揮発性の記憶装置である。この章では、代表的な補助記憶装置の種類と仕組みについて学習する。



4-1 磁気ディスク

磁気ディスクは、代表的な補助記憶媒体として、最も広く使用されている記憶媒体である。磁気ディスクは、表面に磁性体を塗布した円盤状の記憶媒体で、磁気の変化によってデータを記録する。データの記録は同心円上で行われ、この一つの同心円を**トラック**、トラックを一定の長さに分割した単位を**セクタ**という。現在の磁気ディスクでは、外周側のトラックほどセクタ数を増やす定記録密度方式（ゾーンセクタ方式）が一般的である。

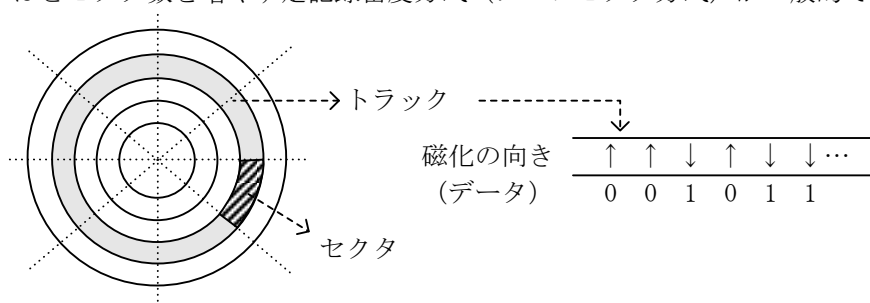


図 1-42 磁気ディスク

磁気ディスクの磁化の向きを読み込んだり，変化させたりするのは**磁気ヘッド**である。磁気ヘッドは，電磁石が組み込まれた部品であり，アクセスアームの先端に固定されて目的のトラック上からデータを読み取ったり，書き込んだりする。

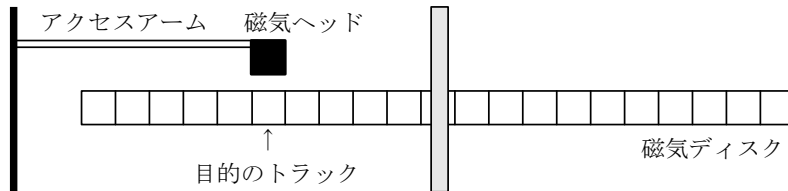


図 1-43 磁気ディスクと磁気ヘッド

この磁気ディスクを用いた補助記憶装置（補助記憶媒体）として，**ハードディスク装置**（HDD：Hard Disk Drive）がある。

ハードディスク装置は，ほとんどのコンピュータに標準装備されている補助記憶装置である。コンピュータ内部に格納されている内蔵型 HDD のほかに，コンピュータに接続して使用する外付け HDD などもある。密閉された筐体内に複数の磁気ディスクを重ね合わせて格納しておき，大容量のデータを記録できる。重ね合わせた複数の磁気ディスクからデータを読み込むために，磁気ヘッドを付けたアクセスアームも複数装備されている。

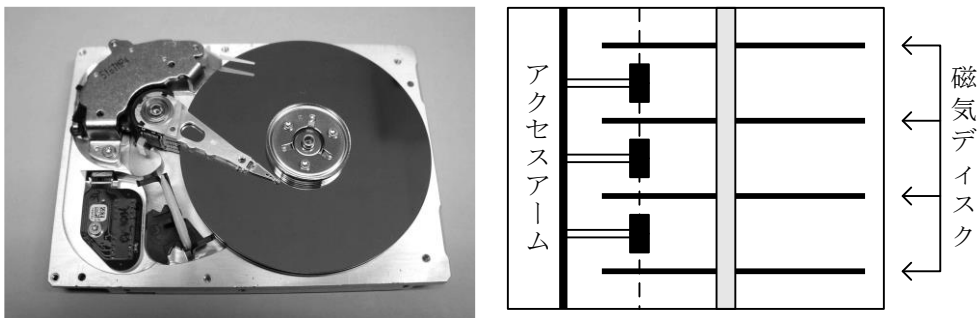


図 1-44 ハードディスク装置の構造

ハードディスク装置からのデータの読み書きは，アクセスアームを左右に動かして磁気ディスク上の目的のトラックに磁気ヘッドを位置付けて行う。このとき，図 1-44 からわかるように，同じアクセスアームの位置で複数のトラック（破線上のトラック）を読み書きすることができる（このトラックの集合を**シリンダ**という）。データの読み書きは，このシリンダ単位で行われる。

なお，ハードディスク装置では磁気ディスクが常に高速で回転しているため，磁気ヘッドを接触させずに情報の読み書きを行う（非接触式）。そのため，振動に弱いという欠点がある。

4-1-1 磁気ディスク装置の記憶容量

記憶容量とは、記憶装置に記録できるデータ量のことである。一般的な磁気ディスク装置の構成は、“磁気ディスク装置＞シリンダ＞トラック＞セクタ”となっている。

したがって、磁気ディスク装置の記憶容量は、構成単位の小さいほうから大きいほうへと、順番に求めていけばよい。

〔磁気ディスク装置の記憶容量の求め方〕

- 1) 1セクタに記録できるデータ量と1トラックのセクタ数から、1トラック当たりの記憶容量を求める。
- 2) 1トラックに記録できるデータ量と1シリンダのトラック数から、1シリンダ当たりの記憶容量を求める。
- 3) 1シリンダに記録できるデータ量と磁気ディスク装置のシリンダ数から、磁気ディスク装置の記憶容量を求める。

なお、磁気ディスクの枚数が少ない場合は、シリンダの代わりに磁気ディスクの面数を用いて記憶容量を求める場合もある。

例. 次の仕様の磁気ディスク装置の記憶容量を求めよ。

シリンダ数／装置	300 シリンダ
トラック数／シリンダ	20 トラック
セクタ数／トラック	30 セクタ
バイト数／セクタ	500 バイト

- ① 1トラックの記憶容量
＝1セクタの記憶容量×1トラックのセクタ数
＝500 バイト／セクタ×30 セクタ／トラック
＝15,000 バイト／トラック
- ② 1シリンダの記憶容量
＝1トラックの記憶容量×1シリンダのトラック数
＝15,000 バイト／トラック×20 トラック／シリンダ
＝300,000 バイト／シリンダ
- ③ 磁気ディスク装置の記憶容量
＝1シリンダの記憶容量×磁気ディスク装置のシリンダ数
＝300,000 バイト／シリンダ×300 シリンダ／装置
＝90,000,000 バイト／装置 …… 90Mバイト／装置

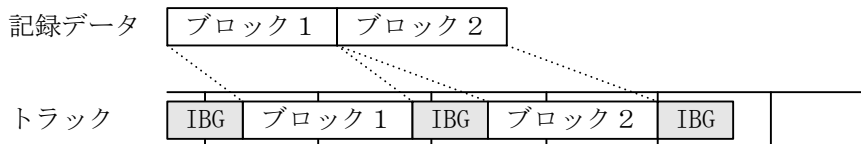
4-1-2 磁気ディスク装置の記録領域

記録領域とは、記憶装置にデータを記録するのに必要な領域のことである。磁気ディスク装置でデータの記録に必要な記録領域の大きさを求めるためには、データをどのような単位で扱うか（データの記録方式）が重要となる。一般的な磁気ディスク装置では、論理的な意味をもつ単位である**レコード**を複数まとめて、**ブロック**を入出力の単位として扱う（通常のレコードを**論理レコード**、ブロックを**物理レコード**という場合もある）。複数のレコードをブロックにまとめることを**ブロックキング**、ブロックにまとめたレコードの数を**ブロック化因数**（**ブロック化係数**）という。

〔データの記録方式〕

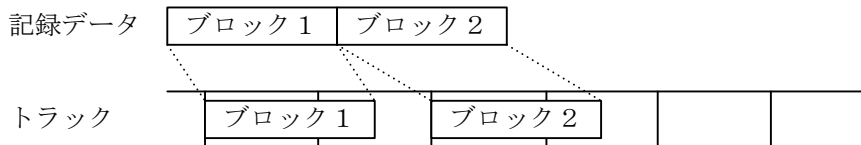
① バリアブル方式

データの読み書きを、ブロック単位で行う方式である。トラックに記録するとき、ブロックとブロックの間に**ブロック間隔**（**IBG**）を入れて区切りとする。



② セクタ方式

データの読み書きを、セクタ単位で行う方式である。同一セクタ内に複数のブロックを記録しないことでブロックの区切りとする。なお、複数セクタをまとめて入出力単位とする**クラスタ**という概念を用いることもある。



現在のPC（磁気ディスク装置）で一般的に使用されるセクタ方式における、データの記録に必要な記録領域の大きさの求め方は、次のとおりである。

〔磁気ディスク装置の記録領域の大きさの求め方〕

- 1) 1ブロックの大きさから、記録に必要なセクタ数を求める。
- 2) 記録する全データのブロック数を求める。
- 3) 全データを記録するのに必要なセクタ数を求め、必要なトラック数、必要なシリンダ数と順番に求める。

例. 次の仕様の磁気ディスク装置に、1 レコード 300 バイトのレコードをブロック化因数 10 で、10,000 レコード記録するのに必要なシリンダ数を求めよ。なお、磁気ディスクの記録方式は、セクタ方式である。

シリンダ数／装置	300 シリンダ
トラック数／シリンダ	45 トラック
セクタ数／トラック	30 セクタ
バイト数／セクタ	512 バイト

- ① 1 ブロックの大きさ

$$= 1 \text{ レコードの大きさ} \times \text{ブロック化因数}$$

$$= 300 \text{ バイト} / \text{レコード} \times 10 \text{ レコード} / \text{ブロック}$$

$$= 3,000 \text{ バイト} / \text{ブロック}$$
- ② 1 ブロックの記録に必要なセクタ数

$$= 1 \text{ ブロックの大きさ} \div 1 \text{ セクタの記憶容量}$$

$$= 3,000 \text{ バイト} / \text{ブロック} \div 512 \text{ バイト} / \text{セクタ}$$

$$= 5.85\cdots \text{ セクタ} / \text{ブロック}$$

※5セクタでは足りないので、切り上げて6セクタ／ブロック
- ③ 記録するブロック数

$$= \text{全レコード数} \div \text{ブロック化因数}$$

$$= 10,000 \text{ レコード} \div 10 \text{ レコード} / \text{ブロック}$$

$$= 1,000 \text{ ブロック}$$
- ④ 全レコードの記録に必要なセクタ数

$$= 1 \text{ ブロックの記録に必要なセクタ数} \times \text{記録するブロック数}$$

$$= 6 \text{ セクタ} / \text{ブロック} \times 1,000 \text{ ブロック}$$

$$= 6,000 \text{ セクタ}$$
- ⑤ 全レコードの記録に必要なトラック数

$$= \text{全レコードの記録に必要なセクタ数} \div 1 \text{ トラックのセクタ数}$$

$$= 6,000 \text{ セクタ} \div 30 \text{ セクタ} / \text{トラック}$$

$$= 200 \text{ トラック}$$
- ⑥ 全レコードの記録に必要なシリンダ数

$$= \text{全レコードの記録に必要なトラック数} \div 1 \text{ シリンダのトラック数}$$

$$= 200 \text{ トラック} \div 45 \text{ トラック} / \text{シリンダ}$$

$$= 4.44\cdots \text{ シリンダ}$$

※4シリンダでは足りないので、切り上げて5シリンダ

※②、⑥の計算で、計算結果を切り上げている意味に注意すること。

4-1-3 磁気ディスク装置の平均アクセス時間

データを読み込んだり、書き込んだりする動作を**アクセス**といい、アクセス要求を出してから結果が返ってくるまでの時間を**アクセス時間**（**アクセスタイム**）という（これに対して、アクセス要求を受け付けてから、次のアクセス要求を受け付けることができるまでの時間を**サイクル時間**（**サイクルタイム**）という）。

磁気ディスク装置におけるデータの読み込み（書き込み）は、次の手順で行われる。

① **位置決め**（**シーク**）

磁気ヘッド（アクセスアーム）を、目的のトラック（シリンダ）まで移動する。このための時間を**位置決め時間**（**シーク時間**）という。

② **回転待ち**（**サーチ**）

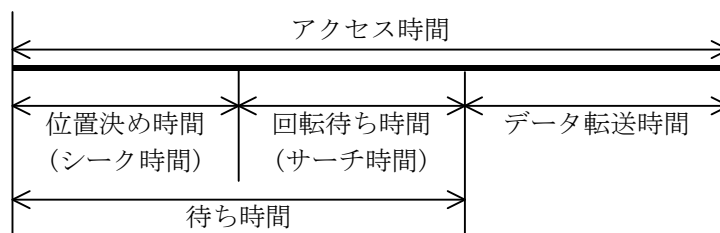
読み込むデータの先頭（又は書き込みを開始する位置）が磁気ヘッドの真下に来るまで待つ。このための時間を**回転待ち時間**（**サーチ時間**）という。

③ **データ転送**

磁気ヘッドがデータを読み込む（書き込む）。このための時間を**データ転送時間**という。

したがって、アクセス時間は、これらの時間を加算した値となる。

〔磁気ディスク装置のアクセス時間〕



$$\text{アクセス時間} = \text{位置決め時間} + \text{回転待ち時間} + \text{データ転送時間}$$

一方、**平均アクセス時間**とは、1入出力単位 of データを読み込む（書き込む）ためのアクセス時間の平均値である。この場合、1入出力単位 of データ量は同じであるから、データ転送時間は毎回同じとなる。しかし、位置決め時間や回転待ち時間は毎回異なるので、それぞれの平均値（平均位置決め時間、平均回転待ち時間）を利用する。

〔磁気ディスク装置の平均アクセス時間〕

$$\text{平均アクセス時間} = \text{平均位置決め時間} + \text{平均回転待ち時間} + \text{データ転送時間}$$

平均アクセス時間を求めるために、平均位置決め時間、平均回転待ち時間、データ転送時間として利用する値は、それぞれ次のとおりである。なお、平均待ち時間を利用する場合は、“平均位置決め時間＋平均回転待ち時間”で求めることができる。

① **平均位置決め時間（平均シーク時間）**

磁気ヘッドが目的のトラック（シリンダ）まで移動する時間であるため、アクセスを開始する時点での磁気ヘッドの位置と、目的とするトラックの位置によりシーク時間は毎回異なる。そのため、一般的には磁気ディスクの仕様として明示されている平均位置決め時間（平均シーク時間）を利用する。

② **平均回転待ち時間（平均サーチ時間）**

磁気ディスクは一定速度で回転しているため、データの先頭が最初から真下にあつて、まったく待ち時間がない場合もある（最短回転待ち時間）。一方、データの先頭がちょうど通り過ぎた後で、ディスクが一周するまで待つ場合もある（最長回転待ち時間）。したがって、最短回転待ち時間と最長回転待ち時間の平均（磁気ディスクが1回転する時間の1/2）を利用する。

$$\text{平均回転待ち時間} = \frac{\text{磁気ディスクが1回転する時間}}{2}$$

③ **データ転送時間**

データの読み書きは、目的のデータが磁気ヘッドの下を通過するときに行われる。つまり、磁気ディスクが1回転すると、磁気ヘッドの下をトラックが一周することになるので、1トラックに記録されているデータが読み書きされることになる。そのため、磁気ディスクの**データ転送速度**は、次のように求められる。

$$\text{データ転送速度} = \frac{\text{1トラックの記憶容量}}{\text{磁気ディスクが1回転する時間}}$$

あとは、1入出力単位（読み書きするデータ）のデータ量（転送データ量）を、データ転送速度で除算してデータ転送時間を求める。

平均回転待ち時間とデータ転送時間は、ディスクの回転速度と密接な関係がある。そのため、ディスクの回転速度を上げることが時間短縮につながる。一方、アクセスアームの移動にかかる物理的な時間である位置決め時間は、アクセス時間に占める割合が大きくなる傾向がある。そのため、データを連続領域に記録し、アクセスアームの移動量を少なくして位置決め時間を短縮することが、平均アクセス時間の短縮には効果的である。なお、データが断片的に記録される状態（**フラグメンテーション**）では平均アクセス時間が長くなるので、断片化を解消する**デフラグメンテーション**が必要となる。

例. 次の仕様の磁気ディスク装置に記録されている1ブロック 2,000 バイトのデータを読み込むための平均アクセス時間を求めよ。

回転速度	3,000 回転/分
バイト数/トラック	10,000 バイト
平均位置決め時間	30 ミリ秒

① 磁気ディスクが1回転する時間

$$3,000 \text{ 回転} : 1 \text{ 分} = 1 \text{ 回転} : x \text{ ミリ秒}$$

$$3,000 \text{ 回転} : 1 \times 60 \times 10^3 \text{ ミリ秒} = 1 \text{ 回転} : x \text{ ミリ秒}$$

$$3,000 \text{ 回転} \times x \text{ ミリ秒} = 1 \times 60 \times 10^3 \text{ ミリ秒} \times 1 \text{ 回転}$$

$$x \text{ ミリ秒} = 1 \times 60 \times 10^3 \text{ ミリ秒} \times 1 \text{ 回転} \div 3,000 \text{ 回転}$$

$$= 20 \text{ ミリ秒}$$

② 平均回転待ち時間

$$= \text{磁気ディスクが1回転する時間} \div 2$$

$$= 20 \text{ ミリ秒} \div 2$$

$$= 10 \text{ ミリ秒}$$

③ データ転送速度

$$= 1 \text{ トラックの記憶容量} \div \text{磁気ディスクが1回転する時間}$$

$$= 10,000 \text{ バイト/トラック} \div 20 \text{ ミリ秒/回転 (トラック)}$$

$$= 500 \text{ バイト/ミリ秒}$$

④ データ転送時間

$$= \text{転送データ量} \div \text{データ転送速度}$$

$$= 2,000 \text{ バイト} \div 500 \text{ バイト/ミリ秒}$$

$$= 4 \text{ ミリ秒}$$

⑤ 平均アクセス時間

$$= \text{平均位置決め時間} + \text{平均回転待ち時間} + \text{データ転送時間}$$

$$= 30 \text{ ミリ秒} + 10 \text{ ミリ秒} + 4 \text{ ミリ秒}$$

$$= 44 \text{ ミリ秒}$$

最後に、ここで求めた平均アクセス時間は、あくまでも磁気ディスク装置単体としての値である点を補足しておく。例えば、**ディスクキャッシュ**が利用されているときは、処理装置（主記憶装置）から見た磁気ディスク装置（補助記憶装置）の平均アクセス時間は、ここで求めた平均アクセス時間にならない。この場合は、**ヒット率**又は**NFP**（Not Found Probability）を考慮して、ディスクキャッシュのアクセス時間と磁気ディスク装置のアクセス時間の加重平均で**平均アクセス時間（実効アクセス時間）**を求めなければならないからである。

4-2 光ディスク

光ディスクは、現在一般的に利用されている補助記憶媒体（**リムーバブルメディア**）である。光ディスク装置（ドライブ）は、装置により仕組みに違いがあるものの、レーザ光によりデータの読み書きを行う大容量の補助記憶装置であることは共通している。光ディスク装置（ドライブ）には、オーディオ機器のようにディスクをそのまま挿入口に入れる**スロットイン方式**と、ディスクを乗せるトレイが出てくる**トレイ方式**がある。

光ディスクの構造は基本的に磁気ディスクと同じであるが、データを記録するトラックが螺旋（らせん）状になっている点が大きく異なっている。データの読み込みはレーザ光を当てた反射光の違いを検出する光ヘッドで行われるが、磁気ヘッドに比べて重いので位置決め時間（シーク時間）が長くなる。その結果、光ディスク装置の平均アクセス時間は、磁気ディスク装置よりも長くなる傾向がある。

光ディスクには、標準記憶容量が700Mバイト程度の**CD**（Compact Disc）と、多層化やレーザ光の波長を短くすることで大容量化（標準記憶容量：片面一層4.7Gバイト～片面二層8.5Gバイト）した**DVD**（Digital Versatile Disc）などがある。

(1) 読出し専用型

読出し専用型の光ディスクとは、工場などでディスク面上にピットと呼ばれる小さな穴をあけ、レーザ光の反射を変えることで情報を記録する光ディスクである。原盤をプレスすることで大量に生産できるため安価であるが、利用者が新たにデータを書き込むことはできない。代表例として、**CD-ROM**（CD-Read Only Memory）や**DVD-ROM**がある。

なお、CD-ROMは元々音楽用に開発されたものであることから、音楽用データとデジタルデータを混在させることもできる。また、CD-ROMやDVD-ROMを大量生産する場合、アルミ蒸着層（反射層）をアクリルで挟んだものを接着剤で装着した上でプレスされる。この接着剤がはく離すると、表面を覆う樹脂（アルミ蒸着剤）が劣化（酸化）してデータが読み取れなくなるため、CD-ROMやDVD-ROMの寿命は20～30年程度といわれている。

(2) 追記型

追記型の光ディスクとは、工場などで書き込まれたデータの削除や修正は行えないが、データを記録できる部分を作っておくことでデータの追加（記録）を可能とした光ディスクである。ディスク上に塗布してある有機色素を、強いレーザ光により焦がしてピットを作ることでデータを書き込むことができる。ただし、焦がした有機色素は元に戻せないため、追記した内容であっても書き換えることはできない（WO: Write Once）。代表例として、**CD-R**（CD-Recordable）や**DVD-R**、DVD+Rなどがある。

(3) 書換型

書換型の光ディスクとは、データを書き換えることができる光ディスクである。記録の仕組みとしては、光と熱を使い、ディスクの記録面にかける温度を制御して、結晶状態と非結晶状態の二つの状態を作る金属相変化技術によってデータを記録する。追記／削除／修正が行えるため、磁気ディスクと同様に扱える補助記憶媒体である。代表例としては、**CD-RW** (CD-ReWritable) や **DVD-RW**, **DVD-RAM** (DVD-Random Access Memory), DVD+RW などがある。追記型と書換型は、共通の再生・記録装置 (**CD-R/RW ドライブ**, **DVD-R/RW ドライブ**) を利用することができる。しかし、データ記録方法の違いなどによりレーザ光の反射率が異なることから、読出し専用型用の再生装置 (**CD-ROM ドライブ**, **DVD-ROM ドライブ**) ではデータを正しく読み込めないこともある。

【光ディスクの書き込み方式】

- ・ **ディスクアットワンス** (DAO : Disk At Once)
ディスク全体に、一度にまとめてデータを書き込む方式である。
- ・ **トラックアットワンス** (TAO : Track At Once)
トラック単位にデータを書き込む方式である。
- ・ **セッションアットワンス** (SAO : Session At Once)
セッション単位でデータを書き込む方式である (セッションは、始まりを表すリードイン、データ、終りを表すリードアウトで構成された記録単位)。
- ・ **マルチセッション**
1枚のディスクに複数のセッションを記録できる方式である。この方式による記録は、TAO や SAO をサポートした装置で実現される。
- ・ **パケットライティング**
磁気ディスクと同様に、データを小さな単位 (パケット) で書き込む方式である。この方式により、光ディスクの用途が広がっている。

(4) ブルーレイディスク (Blu-ray Disc)

ブルーレイディスクは、従来の光ディスクに使用されているレーザ光よりも、さらに波長の短い青紫色レーザ光を使用することにより大容量化を実現している。ブルーレイディスク用の再生・記録装置である **ブルーレイドライブ**は、CD や DVD などにも対応しているものが多い (これらは**光学ドライブ**と総称されている)。エンターテインメント系の利用目的が主である (そのため、映像コンテンツの著作権保護技術 **AACS** も採用されている) が、PC の補助記憶媒体としても利用されている。

4-3 半導体メモリ

半導体メモリ（ICメモリ）は、RAM／ROM／フラッシュメモリなどのICを利用した記憶素子である。主記憶装置やレジスタを構成する記憶素子として使用される以外に、補助記憶装置の記憶媒体としても利用されている。

(1) SSD（Solid State Drive；ソリッドステートドライブ）

SSD（ソリッドステートドライブ）は、ハードディスク装置（HDD）に代わる装置として期待されている、フラッシュメモリを用いた補助記憶装置である。SSDでは、フラッシュメモリを利用して情報の読み書きを電氣的に行うので、ハードディスク装置のように駆動部（ディスクやアクセスアームなど）を必要としない。

そのため、物理的な位置決めや回転待ちが発生しないのでアクセス時間が短く、データの記録順序に関係なく読み書きする直接アクセス（ランダムアクセス）も高速に行える。また、小型・軽量化が可能で、消費電力も少なく、振動や衝撃にも強いという利点がある。ただし、データの書き込み回数（耐久性）に上限があり、データの書換えが多発する用途には適さないという欠点もある。

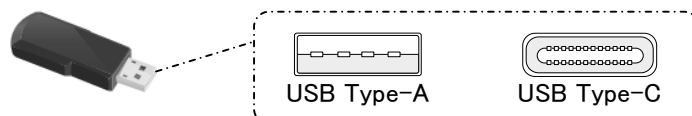


(2) RAM ディスク

RAM ディスクは、主記憶装置（DRAM）の一部を外部記憶装置と仮想化して利用する補助記憶装置である。高速なアクセスが可能であるが、電源を切るとデータが消えてしまう揮発性のため、データの保存には適さない（保存するデータはハードディスクなどに退避する必要がある）。また、RAM ディスクに記録されたデータ（ファイル）を**RAM ファイル**、データを管理する仕組みを**RAM ファイルシステム**と呼ぶこともある（あるソフトウェアで利用される拡張子“.ram”のファイルもRAMファイルと呼ばれるが、これとは異なる）。

(3) USB メモリ

USB メモリは、フラッシュメモリにUSBコネクタを接続して、コンピュータとの着脱を簡単にした補助記憶媒体（リムーバブルメディア）で、データの持ち運びに適している。PC用のUSBメモリのUSBコネクタ及び接続口の規格は**USB Type-A**が主流であったが、最近では上下・左右の区別がない規格の**USB Type-C**の製品もシェアを広げている。



(4) SD カード (SD メモリカード)

SD カード (SD メモリカード) は、フラッシュメモリをチップ状にし、コンピュータやデジタルカメラ／スマートフォン／携帯電話などの SD スロットに差し込んで使用する補助記憶媒体 (リムーバブルメディア) である。著作権保護技術として **CPRM** (又は強化版の CPXM) を採用している。

〔SD カードの分類・規定内容〕

項目	名称	規定内容
サイズ	SD カード	32mm×24mm
	microSD カード	15mm×11mm (SD カードの 4 分の 1 以下)
容量	SD	最大 2 GB (ファイルシステム : FAT16)
	SDHC	最大 32GB (ファイルシステム : FAT32)
	SDXC	最大 2 TB (ファイルシステム : exFAT)
スピード	SD スピードクラス	② (2 MB/秒) ~ ⑩ (10MB/秒)
	UHS スピードクラス	1 (10MB/秒) ~ 3 (30MB/秒)
	ビデオスピードクラス	V6 (6 MB/秒) ~ V90 (90MB/秒)

また、GPS、カメラ、無線 LAN アダプタなどの周辺機能を、ハードウェアとしてカードに搭載した **SDIO カード** などもある。

4 - 4 その他の補助記憶装置

(1) 磁気テープ装置

磁気テープ装置は、磁性体を塗布したテープ (**磁気テープ**) を磁化することによって、データを記録する補助記憶装置である。磁気テープ装置のアクセス速度は非常に低速で、データを順番に連続して読み書きする順次アクセスしかできない (磁気テープ装置のようにデータを連続して記録する装置のことを、**ストリーマ**と呼ぶ)。しかし、磁気テープは単位情報当たりのコストが非常に安価であり、大量のデータを記録できる補助記憶媒体であるため、ハードディスクのバックアップなどに利用されている。従来は、やや大きくてかさ張るため、持ち運びには不便なオープンリール式の磁気テープが主流であった。しかし、現在では、音楽用に開発された **DAT** (Digital Audio Tape) の技術を利用した、持ち運びに便利なカートリッジタイプの **DDS** (Digital Data Storage) が主流である。

5 入出力装置

コンピュータを利用するには、コンピュータにデータを入力し、内部で処理した結果を出力する必要がある。このとき、データを入力するために使用する装置を入力装置、出力するために使用する装置を出力装置という。

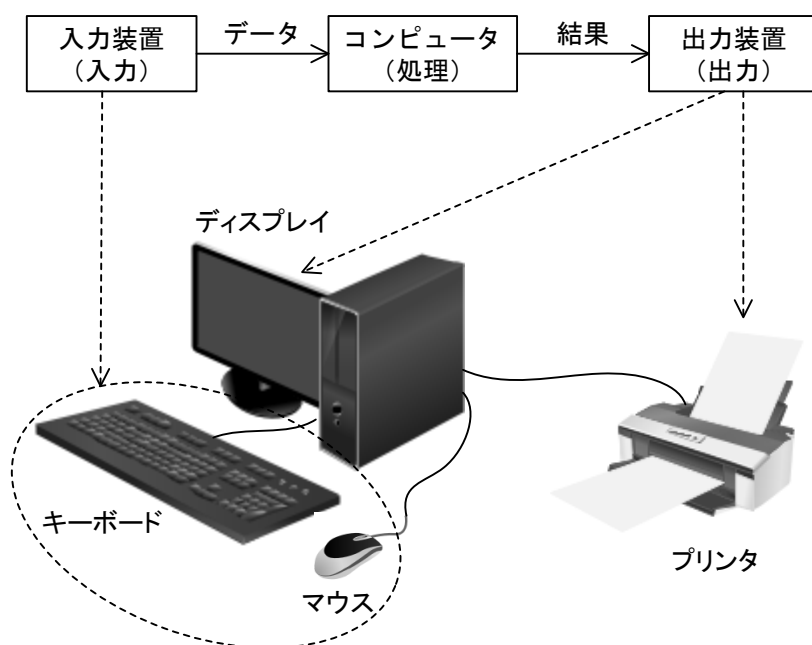


図 1-45 データ処理の流れと入出力装置

図 1-45 に示したキーボード、マウス、ディスプレイ、プリンタは、PC などで使用されている代表的な標準入出力装置である。これらの装置は、多くの人が実際に利用しているのではないだろうか。

しかし、このような代表的な入出力装置以外にも、入力するデータや出力する形式に合わせた機能をもつ、さまざまな種類の入力装置や出力装置が存在する。この章では、それらの装置も含めた入出力装置の種類と特徴、および仕組みについて学習する。

5-1 入力装置

入力装置とは、コンピュータにデータを入力するための装置である。キーボードのように文字や数値を信号で入力する装置、マウスのように位置情報などを入力する装置（**ポインティングデバイス**）のほかにも、用途に応じたさまざまな入力装置がある。

5-1-1 文字や数値を信号で入力する装置

(1) キーボード

キーボードは、もっとも汎用的な入力装置であり、PCなどの標準入力装置として必ず付いているものである。キーボード上のキー（鍵盤）を押すと、対応する文字や数値が信号（符号）として入力される装置である。



キーボード

なお、古いキーボードを操作するとき、キーを一度押しただけで、同じキーが複数回押されたように動作することがある。これを**チャタリング**と呼ぶこともあるが、本来、チャタリングとは、機械式接点の押しボタンを1回押すことに対して、ボタンを押してから短時間（数ミリ秒）で、複数回のON/OFFの信号が発生する現象のことである。

5-1-2 ポインティングデバイス

(1) マウス

マウスは、マウス本体の移動量や方向を入力する装置である。元々は、底部のボールを回転させることにより、その回転量や方向を入力する装置であった。しかし、回転するボールがごみや埃を巻き込むなどの問題があり、現在は赤色LEDや青色LEDなどを利用した光学式マウスが主流になっている。また、ケーブルで接続する有線式マウスのほかに、ワイヤレス（ケーブルなし）で接続する無線式マウスもある。



有線式マウス



無線式マウス

(2) タッチスクリーン（タッチパネル）

タッチスクリーン（タッチパネル）は、ディスプレイの画面に直接指で触れることにより、位置情報を入力する装置である。

〔タッチスクリーンの入力方式〕

- ・ **赤外線遮光方式（赤外線方式）**

赤外線ビームが遮られて起こる赤外線反射の変化で位置を検出する。

- ・ **静電容量方式**

タッチした部分の静電気による表面電荷の変化で位置を検出する。

- ・ **アナログ抵抗膜方式（抵抗膜方式）**

タッチした部分の抵抗膜の抵抗値の変化で位置を検出する。

- ・ **マトリックススイッチ方式**

マトリックス上に並んだ電極スイッチの押された部分で位置を検出する。

(3) デジタイザ／タブレット

デジタイザは、パネル上の図形をなぞることにより、2次元や3次元の図形情報を座標位置情報として入力する装置である。机上で使える小型の装置は、（ペン）**タブレット**と呼ばれる。CADと呼ばれる設計支援システムの図面入力や、NC（数値制御）工作機械のデータ作成などに利用される。

(4) その他のポインティングデバイス

① トラックボール

上向きに設置されたボールを指で動かすことにより、その回転量や方向を入力する装置である。ノートPCの入力装置として搭載されていた時期もあったが、現在ではマウスと同じ形状のトラックボールマウスとして販売されていることが多い。

② ポインティングスティック／タッチパッド

主に、キーボードとセットでノートPCに搭載されているポインティングデバイスである。スティックやパッドを指でなぞることで位置情報を入力する。

③ ジョイスティック

操作棒を前後左右に動かして、倒した方向とその角度によって位置情報を入力する装置である。ゲームソフトの操作などで使われる。

④ ライトペン

ディスプレイの画面上を直接指すことにより、位置情報を入力する装置である。先端の光センサの働きにより、ディスプレイ上の位置を検出することができる。

5-1-3 画像入力装置

(1) イメージスキャナ (スキャナ)

イメージスキャナ (スキャナ) は、紙面上の図形や写真などを、ファクシミリと同じ原理で光学的に読み取り、点の集合体であるドットイメージとして入力する装置である。また、紙面を固定して読取装置を移動させる形式の**イメージセンサ**や、立体物の形状を感知して 3D データとして出力する**3D スキャナ**などもある。



イメージスキャナ

(2) デジタルカメラ

デジタルカメラ は、撮影した画像をコンピュータに入力する装置である。通常のカメラのフィルムの代わりに **CCD** (Charge Coupled Device) を使用して、画像イメージをデジタルデータ化する。画像イメージの記憶媒体として、**スマートメディア** と呼ばれるフラッシュメモリの一種が利用されることが多い。また、動画の記録には**デジタルビデオカメラ** が利用されるが、リアルタイムで映像を確認できる **Web カメラ** などもある。



デジタルカメラ



Webカメラ

(3) OCR／OMR／MICR

OCR (Optical Character Reader ; **光学式文字読取装置**) は、手書きされた文字や印刷された文字を反射光の強弱により光学的に読み取り、データとして入力する装置である。手紙やはがきに記入された郵便番号の読取りなどにも利用されているが、誤読 (読み間違い) の危険性が残るため、入力データの十分なチェックが必要である。

OMR (Optical Mark Reader ; **光学式マーク読取装置**) は、マークシート上の黒く塗りつぶされた (マークされた) 部分を反射光で光学的に読み取り、データとして入力する装置である。受験者が多い試験の答案用紙の読取り (採点) などに利用されている。

MICR (Magnetic Ink Character Reader ; **磁気インク文字読取装置**) は、磁性体を混ぜた特殊なインクで書かれた、特殊な字体の文字を入力する装置である。小切手や手形の読取装置として利用されている。

5-1-4 その他の入力装置

(1) 生体認証装置

生体認証装置は、人間の身体的／行動的特徴から生体認証を行うための情報を入力する装置である。顔認証装置、掌認証装置、指紋認証装置、虹彩認証装置など、認証方法ごとに、認証対象者の情報を入力（認証）する。



生体認証装置
(指紋認証装置)

(2) バーコードリーダ（バーコード読取装置）

バーコードリーダ（バーコード読取装置）は、異なる太さのバー（線）と異なる間隔の組合せによってデータを表現する**バーコード**を、光学的に読み取る装置である。代表的な利用方法としては、小売店のレジなどで使用されるPOS システムの入力装置がある。



バーコードリーダ

〔代表的なバーコード〕

・ 1 次元バーコード

太さの異なる縦線を帯状にして横に並べたバーコードである。単に“バーコード”という場合は、1次元バーコードを意味することが多い。包装材や容器に直接印刷でき、印刷寸法も自由に拡大・縮小できるため、さまざまな場面で利用しやすく、コストも比較的安いという利点がある。

JAN コード（標準商品表示バーコード）、**ITF コード**（標準物流バーコード）、**Code128**（ASCII 文字を扱うバーコード）などが利用されている。

・ 2 次元バーコード（QR コード）

情報を縦横のマス目で表示するマトリックス方式を利用したバーコードである。切出しシンボルという3個の検出用シンボルがあり、回転角度と読取り方向が認識できるので、どの方向からでも情報を読み取れる。また、専用の読取装置を使わずに、スマートフォンや携帯電話でも読み取れる。



1 次元バーコード



2 次元バーコード

(3) 音声入力装置

音声入力装置は、人間の音声（アナログ信号）を識別し、電子回路（A/D コンバータ）を利用してデジタル信号に変換したデータを入力する装置である。音声パターンや声紋を認証する生体認証装置としても利用されている。

代表的な音声入力装置の一つであるマイクには、一方向からの音声に集中する一般的な単一指向性マイクのほかに、どの方向からの音声でも均等に收音していく全指向性マイクがある。全指向性マイクは無指向性マイクとも呼ばれ、大勢の参加者が集まって意見を述べ合う会議などに用いられる。



単一指向性マイク



全指向性マイク

(4) カード読取装置

カード読取装置は、カードに記録されている情報を読み取る装置である。代表的なものとして、磁気カード読取装置やICカード読取装置などがある。磁気カード読取装置は、銀行のキャッシュカードやクレジットカードなどに、磁気で記録されている情報を読み取る。また、ICカード読取装置は、ICを組み込んだICカード（社内への入館許可証やカード型キーなどに利用されている）に記録されている情報を読み取る。



磁気カード読取装置
(例. 銀行のATM)



ICカード読取装置
(例. 入退室管理)

5-2 出力装置

出力装置とは、処理した結果をコンピュータ外部に出力するための装置である。出力装置には、結果を表示するディスプレイ、結果を印刷するプリンタなどがある。

5-2-1 ディスプレイ

ディスプレイは、光の3原色（**RGB**；Red：赤，Green：緑，Blue：青）を組み合わせ、カラー表示をする装置である。画面は点（**ドット**）の集合体であり、ドットが密集しているほど高画質な画像を表示することができる。これを画面の**解像度**といい、1インチ当たりのドット数である **dpi** (dots per inch) という単位で表す。

〔ディスプレイの画面解像度の規格〕

規格名	画面サイズ	縦横比	規格名	画面サイズ	縦横比
VGA	640× 480	4 : 3	HD	1,280× 720	16 : 9
SVGA	800× 600		FHD	1,920×1,080	
XGA	1,024× 768		4K (QFHD)	3,840×2,160	
SXGA	1,280×1,024	5 : 4	8K (FUHD)	7,680×4,320	

現在のPCのディスプレイは、いろいろな解像度に対応できる**マルチスキャン方式**が主流である。ただし、高解像度に対応するには、高解像度に対応したグラフィックメモリ（**VRAM**：Video RAM）が必要である。VRAMに必要な記憶容量は、画面の解像度と1ドット当たりの色数（色を表現するためのビット数）によって決まる。

例. 1,280×1,024の解像度をもつディスプレイで、RGBの各色に256段階の濃淡をつけて約1,600万色表示するとき、VRAMに必要な記憶容量を求めよ。

- ① 1色の表現に必要なビット数
 $256=2^8$ より、8ビット／色
- ② 1ドットの色情報を表現するのに必要なビット数
 $=8 \text{ ビット／色} \times 3 \text{ 色／ドット}$
 $=24 \text{ ビット／ドット}$
- ③ 1画面に必要なビット数
 $=1 \text{ ドットに必要なビット数} \times 1 \text{ 画面のドット数}$
 $=24 \text{ ビット／ドット} \times (1,280 \times 1,024) \text{ ドット／画面}$
 $=31,457,280 \text{ ビット／画面} \quad \cdots \quad \text{約} 4 \text{ Mバイト／画面}$

また、画面の表示速度や画像の精密度を高めるには、**グラフィックスアクセラレータ**（ビデオチップ）などが必要になる。なお、画面を1行ずつ走査するラスタスキャン方式のディスプレイには、**インタレースモード**という出力方式がある。この方式は、走査線を1本おきに抽出して1枚の画像を2回に分けて出力することで、表示速度を上げる方法である。ただし、この方式では画面にちらつきが生じやすいため、多くの場合、走査線を上から順に出力する**ノンインタレースモード**が利用されている。

(1) 液晶ディスプレイ (LCD : Liquid Crystal Display)

液晶ディスプレイ (LCD) は、電圧により光の透過度が変わる液晶を利用した出力装置である。薄型・軽量で、消費電力も小さいため、ノートPCで使用されることが多いが、デスクトップPCにも使用されている。なお、液晶自体は発光しないため、反射板を組み込む方式か、後ろからライトを当てるバックライト方式が採用されている。複数の画素を一つの半導体で制御する**STN 液晶ディスプレイ**よりも、画素ごとに一つのトランジスタで制御する**TFT 液晶ディスプレイ**のほうが反応速度は速く、視野角も広い。

(2) 有機 EL ディスプレイ (EL : ElectroLuminescence)

有機 EL ディスプレイは、電圧をかけると発光する有機化合物を使用した出力装置である。液晶ディスプレイと異なり、ガラス基板上で自ら発光する。反応速度も速く、画質も高コントラストで輝度が高く、視野角も広い。その上、消費電力も小さいため、次世代のディスプレイとして期待されているが、寿命が短いともいわれている。

(3) プラズマディスプレイ (PDP : Plasma Display Panel)

プラズマディスプレイ (PDP) は、2枚のガラスの間にガスを封入し、電圧をかけることにより発光させる出力装置である。他の方式と比較して輝度が高く、視野角が広いが、高い電圧を必要とするため、消費電力が大きい。

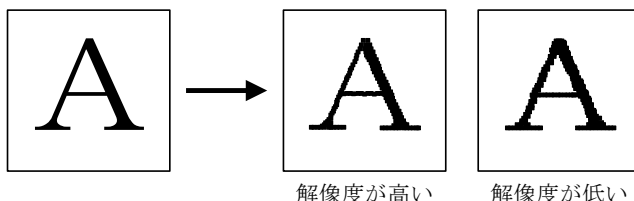
(4) CRT ディスプレイ (CRT : Cathode Ray Tube)

CRT ディスプレイは、テレビと同じブラウン管を使用した出力装置である。蛍光面に電子ビームを当てて発光させ、ビームの強度により任意の文字や画像を表示する。画面が見やすく、表示速度も速いという長所はあったが、奥行きがあつて場所をとり、消費電力も大きいため、現在はあまり見かけなくなっている。また、同じ画面を長時間表示させると焼付けが起これ、それを防ぐ目的で利用されていたソフトウェア（**スクリーンセーバ**）は、現在ではセキュリティ（**クリアスクリーン**）の観点で用いられている。

5-2-2 プリンタ

プリンタは、色の3原色（**CMY**；Cyan：シアン[明るい水色]，Magenta：マゼンタ[明るい赤紫色]，Yellow：黄色）を利用してカラー印刷をする装置である。これに、キー（Key）となる黒色を加えたCMYKの4色を利用する場合が多い。

多くのプリンタは、文字や図形を点（**ドット**）の集まりとして印刷する。したがって、単位面積当たりのドット数が多いほど、きれいに印刷できることになる。これをプリンタの**解像度**といい、ディスプレイと同じ **dpi** (dots per inch) という単位で表す。



プリンタは、印字ヘッドでインクリボンなどを打ち付けて印刷する**インパクトプリンタ**と、インクやトナーを転写して印刷する**ノンインパクトプリンタ**に大別される。前者にはカーボン紙を利用した同時印刷が可能という利点はあるが、動作音が大きいため、動作が静かで、図形や写真も印刷できる**ノンインパクトプリンタ**のほうが一般に普及している。

また、文字単位で印刷する**シリアルプリンタ**、1行分の文字をまとめて印刷する**ラインプリンタ**もあるが、現在はページ単位で印刷する**ページプリンタ**が主流である。

(1) レーザプリンタ

レーザプリンタは、レーザ光で感光ドラムにトナー（粉末インク）を貼り付け、用紙に転写して印刷する**ノンインパクトプリンタ**である。プリンタ又はプリンタドライバがビットマップ形式の印刷イメージを作成して印刷するため、文字の大きさや行の間隔などを自由に調整することができ、画像なども印刷できる（ページイメージを作成するためのプリンタ制御コードを**ページ記述言語**といい、**PostScript**や**ESC/Page**などがある）。性能を表す単位として、1分間の印刷ページ数（**ppm**：page per minute）も用いられる。

レーザプリンタは、印刷速度が速く、1枚当たりの印刷コストも比較的安いことから、主に企業などで使用されている。

(2) インクジェットプリンタ

インクジェットプリンタは、印字ヘッドからインクを吹き付けて印刷する**ノンインパクトプリンタ**である。最近では、4色以上の多色インクを使用して**フォトクオリティ**（写真並みの画像品質）を実現する**フォトプリンタ**が一般的に使用されている。

印刷速度はやや遅いが、高解像度で色を忠実に再現できる高品質の印刷が可能であり、主に一般家庭で使用されている。

(3) 3D プリンタ

3D プリンタは、立体物を積層造形する装置である。造形方法としては、加熱することで軟化する性質の熱可塑性樹脂を用いる**熱溶解積層方式**、紫外線を照射することで硬化する性質の光硬化性樹脂を用いる**光造形方式**などがある。立体物を切削造形する**3D プロッタ**を含めることもある。

5-2-3 その他の出力装置

① プロッタ

CADなどで作成した図面を出力する図形印刷用の装置である。ペンをX軸とY軸に動かす**フラットベッド型プロッタ (XY プロッタ)**と、ペンをY軸方向に動かし、ドラムに巻き付けた用紙をX軸方向に動かす**ドラム型プロッタ**がある。

② プロジェクタ

コンピュータ内部のデータを投影する出力装置で、一般的には大型スクリーンなどにディスプレイの画像を拡大投影するために使用される。プロジェクタの光源の明るさは、**ルーメン (lm)**という単位で表す。画像などを建物、家具といった凹凸のある立体物に投影する**プロジェクションマッピング**などにも利用される。

③ 電子ペーパー

紙の特性を備えた、柔らかい素材の極薄 (0.1mm 程度) の表示装置である。消費電力は微量で、データを繰り返し再生・消去できる。ナノメートル単位の微細電気機械素子及び製造技術である **MEMS** (Micro-Electro-Mechanical Systems) の応用技術の一つである。

④ 音声合成装置

コンピュータ内部のデータ (デジタルデータ) を電子回路 (**D/A コンバータ**) を利用してアナログ信号に変換し、人間の音声などを人工的に作り出して出力する装置である。ディスプレイに表示された情報を読み上げるスクリーンリーダー (画面読上げソフト) や、公共機関・交通機関・イベント会場のアナウンスなどに活用されている。また、サンプリングした人の声を利用し、メロディと歌詞を与えることで歌を歌わせる**ボーカロイド**なども、音声合成装置の活用方法の一つといえる。

⑤ デジタルサイネージ

屋外や交通機関などに設置したディスプレイに、映像や文字などの情報を表示する電子看板である。従来の看板、ポスターとは異なり、表示する情報を簡単に変更することができるため、さまざまな目的に用いられている。

5 - 3 その他の入出力装置

コンピュータにデータを入力するために使用する装置を入力装置、内部で処理した結果を出力するために使用する装置を出力装置と定義する場合、一般的な入出力装置のほかに次のような装置も入出力装置に分類される。

(1) 通信制御装置

通信制御装置とは、コンピュータとネットワークを接続するときに、さまざまな制御をする装置のことである。データの直並列変換（文字の組立て／分解）、データに対する誤り制御などを行う。ネットワークからデータを取得したり、ネットワークに向けてデータを発信したりする装置であることから、入出力装置の一種に分類されることもある。

〔LAN インタフェースカード〕

コンピュータ（主にPC）をLAN（Local Area Network；限定された範囲に敷設された私設通信網）に接続するために使用する、カード型のインタフェース装置である。ケーブルをつないで接続する**有線 LAN インタフェースカード**、電波などの無線で接続する**無線 LAN インタフェースカード**などがある。

(2) 駆動装置

駆動装置とは、装置や機械を動作させる装置のことである。コンピュータからの指示を機械的な動作（例えば、工場などで利用されるロボットアームの動き）に変換して、外部に伝えることから、出力装置の一種といえる。なお、駆動装置が組み込まれた機械などで利用される**センサ**（計測値を電気信号に変換する装置）や**スイッチ**なども、広い意味では入力装置の一種といえる。

・チャタリング

機械式接点の押しボタンスイッチを1回押すことに対して、押してから数ミリ秒の間、複数回のON/OFFの信号が発生する現象である。誤動作を防ぐ対策としては、ON/OFFの信号が安定してから利用する方法などがある。

(3) 撮像装置

撮像装置とは、映像などを入力する装置のことである。デジタルカメラやデジタルビデオカメラも、撮像装置に分類される。そのほかにも、入力画像をコンピュータ内部にデジタルデータとして取り込んで分析する、電子顕微鏡や電波望遠鏡などがある。

5 - 4 入出力制御方式

入出力制御方式は、入出力装置と主記憶装置間におけるデータのやり取りを制御する方式である。実行中のプログラムが入出力命令を発行した場合、**SVC 割込み**が発生して入出力制御の割込みプログラム（入出力制御プログラム）が実行される。その後、入出力動作が完了すると**入出力割込み**が発生し、入出力の完了が通知される。

(1) プログラム制御方式（直接制御方式）

プログラム制御方式（直接制御方式）は、CPU の演算装置で入出力制御プログラムを解釈して入出力装置に指示を出し、CPU を経由して入出力装置と主記憶装置がデータをやり取りする方式である。もっとも基本的な制御方式であるが、CPU と入出力装置では動作速度に大きな差があるため、CPU の利用効率が悪くなる。

(2) DMA 制御方式（DMA : Direct Memory Access ; 直接記憶アクセス）

DMA 制御方式（直接記憶アクセス制御方式）は、DMAC（DMA Controller）という専用装置が入出力を制御することで、CPU を介さずに入出力装置と主記憶装置がデータをやり取りする方式である。CPU から DMAC に DMA 命令（入出力指示）が送られると、DMAC は CPU と独立して入出力を制御するので、その間、CPU は別の処理を行えて利用効率が良くなる。主に、PC の入出力制御として使用されている。

(3) チャネル制御方式

チャネル制御方式は、入出力専用の処理装置（**入出力チャネル**又は**チャネル**という）を使用する方式である。CPU から入出力チャネルに**チャネルプログラム**が送られると、入出力チャネルは自立的に必要な情報を取得して命令を解釈し、CPU と独立して入出力を制御する。主に、汎用コンピュータの入出力制御として使用されている。

【入出力チャネルの分類】

- ・セレクトチャネル …… 1 台の入出力装置を専用で制御する。
- ・マルチプレクサチャネル …… 複数の入出力装置を同時に制御する。
 - ・バイトマルチプレクサチャネル
 バイト単位の低速データ転送を制御する。
 - ・ブロックマルチプレクサチャネル
 ブロック単位の高速データ転送を制御する。

5 - 5 入出力インタフェース

入出力インタフェースとは、入力装置や出力装置などの周辺装置とコンピュータを接続するための取決め（規格）のことである。また、入出力インタフェースによって接続される周辺装置（デバイス）を制御するソフトウェアを**デバイスドライバ**という。

5-5-1 入出力インタフェースの分類

入出力インタフェースは、**シリアルインタフェース**（ビット単位のシリアルデータ転送方式）と、**パラレルインタフェース**（バイトやワード単位のパラレルデータ転送方式）に分類できる。また、コンピュータ内ではデジタル信号が利用されるため、アナログ信号対応の周辺装置を接続するには**アナログ入出力インタフェース**（**アナログ入出力ボード**）を使用して、アナログ信号とデジタル信号を変換する必要がある。

〔周辺装置を接続する際の接続形態（トポロジ）〕

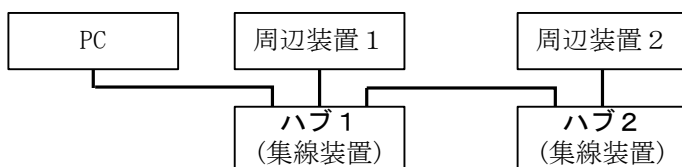
・スター接続

ハブ（集線装置）を経由して、複数の機器を接続する方式である。



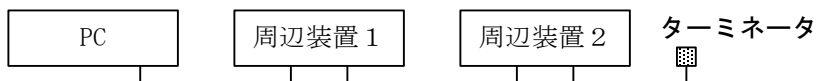
・カスケード接続

ハブ（集線装置）を多段階接続して、複数の機器を接続する方式である。スター接続やカスケード接続のようにツリー状に接続する形態を、**ツリー接続**と総称することもある。



・デジチェーン接続

PC 本体につないだ装置からデジチェーン（数珠つなぎ）方式で、複数の機器を接続する方式である。一般的には、接続の終端に**ターミネータ**と呼ばれる終端装置（終端抵抗）を接続する。



5-5-2 入出力インタフェースの種類と特徴

(1) USB (Universal Serial Bus)

USB は、1996 年に規格化されたシリアルインタフェースで、キーボードやマウス、モデムなど、さまざまな周辺装置を接続できるようにした規格である。USB 規格のハブを用いて、ツリー状に最大 127 台まで装置を接続できる。USB には、一般的に PC などとの接続に使用される規格のほかに、デジタルカメラのような家電製品などとの接続に使用される **miniUSB** や、スマートフォンのような携帯端末などとの接続に使用される **microUSB** という規格もある。また、コネクタの規格には、上下・左右の区別がある **USB Type-A** と **USB Type-B**、上下・左右の区別がない **USB Type-C** がある。

【USB の規格】

- **USB2.0**

ロースピードモード (1.5Mビット/秒)、フルスピードモード (12Mビット/秒)、ハイスピードモード (480Mビット/秒) という 3 種類のデータ転送モードがある。

- **USB3.0**

USB2.0 の三つのデータ転送モードに、スーパースピードモード (5 Gビット/秒) が加えられ、更に通信制御の消費電力を抑えるように規格化されている。USB2.0 と区別しやすいように、コネクタの中を青色にしている製品が多いが、そうでない製品もあるので、注意が必要である。

- **USB3.1**

USB3.0 に、スーパースピードモードプラス (10Gビット/秒) が加えられ、給電能力も向上している。なお、USB3.1 を USB3.1 Gen2 と表記し、USB3.0 を USB3.1 Gen1 と表記している場合もある。

【USB の機能】

- **バスパワー方式** (バスパワー)

接続ケーブルを経由して、PC から電力を供給する方式である。バスパワー方式により供給される電力量は限られているため、消費電力が大きい機器の動作は保証されない。

- **ホットプラグ**

コンピュータの電源を入れたままで、ケーブル (機器) の着脱が可能な方式である。

- **アイソクロナス転送**

一定のデータ転送帯域 (データ転送量) を保証する転送方式である。

(2) IEEE 1394

IEEE 1394 は、デジタルカメラなど、マルチメディア関連機器の接続に用いられる、高速転送可能なシリアルインタフェースである。3種類の基本データ転送モード（100Mビット/秒、200Mビット/秒、400Mビット/秒）があり、PCを介さずに機器同士を接続できることや、複数の機器に同時送信できるブロードキャスト転送モードを利用できるなどの特徴がある。**IEEE** (Institute of Electrical and Electronics Engineers ; 電気電子学会) の規格であり、アップル社の FireWire や、ソニー社の i.Link は IEEE 1394 の商標名である。なお、前ページの USB の三つの機能は、IEEE 1394 にも共通している。

(3) RS-232C (Recommendation Standard-232C)

RS-232C は、主にモデムなどの通信機器との接続に使用されるシリアルインタフェースで、マウスとの接続などにも使用できる。**EIA** (Electronic Industries Association ; 米国電子工業会) で規格化され、**ECIA** (Electronic Components Industry Association ; 電子部品産業協会) に引き継がれている。

・インタフェースボード

PC などの拡張スロットに装着することで、ポート（差込口）を増設することができる基盤（ボード）である（アナログ入出力ボードなど）。データ転送方式に応じて、シリアルインタフェースボード、パラレルインタフェースボードなどの製品がある。例えば、RS-232C インタフェースのポートを増やしたい場合、シリアルインタフェースボード又は RS-232C インタフェースボードを装着する。

(4) SCSI (Small Computer Systems Interface)

SCSI は、主に高速な周辺装置（ハードディスク装置など）との接続に使用されるパラレルインタフェースである。**ANSI** (American National Standards Institute ; 米国規格協会) により規格化され、高速なデータ転送に使用される。SCSI は、PC 本体に接続した装置からデジチェーン接続で最大7台までの周辺装置を接続することが可能である。この場合、接続している機器を識別するために SCSI ID（識別番号：0～7）を割り当てるが、ID が異なっていれば接続順序については問わない（0～7の順でなくてもよい）。

上位規格の Ultra SCSI として、Ultra SCSI（8ビットバス、20Mバイト/秒）、Ultra2 SCSI（8ビットバス、40Mバイト/秒）、Ultra Wide SCSI（16ビットバス、40Mバイト/秒）、Ultra2 Wide SCSI（16ビットバス、80Mバイト/秒）、Ultra160 SCSI（16ビットバス、160Mバイト/秒）、Ultra320 SCSI（16ビットバス、320Mバイト/秒）のほかに、シリアルインタフェース方式を採用したシリアル SCSI（SAS : Serial Attached SCSI）などが規格化されている。

(5) IrDA (Infrared Data Association)

IrDA は、赤外線を使ってデータ通信を行うための規格を標準化している団体、および制定されたシリアルインタフェースである。通信距離は最大 1 m で、パーティション（間仕切り）などの遮蔽物がない状態で、通信するポートを ±15 度以内に配置させる必要がある。接続ケーブルを必要としない赤外線通信は、スマートフォン、タブレット端末などの携帯端末同士や、携帯端末とノート PC のデータ交換などで使われる。

(6) Bluetooth

Bluetooth は、携帯電話や PC などの機器を接続する無線技術の規格で、世界中で利用できる 2.4GHz (2.402~2.480GHz) の周波数帯を利用してデータや音声の通信を行う方式（インタフェース）である。赤外線と異なり、各デバイス間に障害物があっても通信可能なため、低いパーティションで隔てられた機器間での転送が可能である。なお、10cm 程度の近距離無線通信技術には、ソニーとフィリップスが共同開発して、国際標準規格として承認された **NFC** (Near Field Communication) がある。また、スリープ状態からの復帰が、Bluetooth よりも速い **Zigbee** という規格もある。

[Bluetooth の各種規約]

・接続台数

一般的な規格では、同時に接続可能な機器は論理的に最大 7 台であるが、アドレスビットを 3 ビットから 32 ビットに拡張して、無制限（約 2^{32} 台）に接続可能とする規格もある。

・通信速度

通信速度は、バージョンにより定められている。Bluetooth3.0 の場合、最大通信速度は 24Mビット/秒である。なお、上位規格である Bluetooth4.0 では、最大通信速度を 1 Mビット/秒に抑えることで低消費電力モード（BLE : Bluetooth Low Energy）に対応している。

・電波到達距離

電波到達距離は、クラス（Class）により定められている。各クラスの電波到達距離の目安は、表のとおりである。

クラス	電波到達距離
Class1	100m程度
Class2	10m程度
Class3	1m程度

(7) シリアル ATA (Serial Advanced Technology Attachment)

シリアル ATA は、**ANSI** (American National Standards Institute ; 米国規格協会) によって標準化された **IDE** (Integrated Device Electronics) の正式な規格であったパラレルインタフェースの ATA (ATA/ATAPI-4) を、シリアルインタフェースに変更した高速インタフェースである。一般的には、内蔵型ハードディスク装置や CD-ROM ドライブなどの光ディスク装置のように、高速なデータ転送機器を接続するために使用される。現在では、通信速度が 6 G ビット/秒を実現したシリアル ATAⅢが主流となっている。シリアル ATA は原則、1 ポート (コントローラ) に機器を 1 台ずつ接続する形態であるが、**ポートマルチプライヤ**を用いることでポート数を拡張することができる。なお、従来の ATA には、1 本のケーブルに主従関係のある 2 台のデバイス (マスタ、スレーブ) を接続するマスタスレーブ接続という方式が採用されていたが、シリアル ATA では廃止されている。

(8) HDMI (High-Definition Multimedia Interface)

HDMI は、映像と音声をデジタル信号で伝送する、通信インタフェースの標準規格である。アメリカのシリコン・イメージ社が中心となって、日立製作所、松下電器産業、フィリップス、トムソン・マルチメディア、ソニー、東芝が共同で制定した。映像・音声・制御信号を一体化して伝送するシングルケーブルにより、AV 機器の配線を 1 本に簡略化できる。主に、家電製品や AV 機器向けの入出力インタフェースとして開発されたが、現在では PC にも多く採用されている。なお、映像信号を光の 3 原色 (RGB) のアナログ信号で送る **アナログ RGB** に代わって、デジタル信号で伝送することで高品質な画像を実現したシリアルインタフェース **DVI** (Digital Visual Interface) が HDMI の基になっている。

DVI の後継規格としては、**DisplayPort** というシリアルインタフェースもある。基本仕様は HDMI に近いが、データ転送の高速化や機能拡張が容易であることから、今後の大きな飛躍が期待されている。

(9) その他の入出力インタフェース

① セントロニクス

プリンタとの接続に使用されるパラレルインタフェースで、拡張仕様までを含めて **IEEE** (Institute of Electrical and Electronics Engineers ; 電気電子学会) により、IEEE 1284 として標準化されている。

② GPIB (General Purpose Interface Bus)

計測機器との接続に使用されるパラレルインタフェースで、IEEE 488.1 として規格化されている。最大 31 台まで周辺装置を接続できる。

5-5-3 デバイスドライバ

(1) デバイスドライバ

デバイスドライバは、周辺装置（デバイス）を制御するソフトウェアである。デバイスドライバは周辺装置ごとに必要となる固有のソフトウェアであり、デバイスドライバがないと周辺装置は使用できない。例えば、プリンタを使用するためには、そのプリンタ用のデバイスドライバ（プリンタドライバ）が必要となる。また、デバイスドライバを管理するソフトウェアを**デバイスマネージャ**という。なお、複数のデバイスが接続されている場合、それぞれのデバイスの同期をとる（動作タイミングを合わせる）のも、デバイスドライバの役割である。

基本的な周辺装置のデバイスドライバは、あらかじめ準備されているが、その他の周辺装置（新しく購入した周辺装置など）のデバイスドライバは、同梱されている CD-ROM やインターネットなどから取得し、個別に**インストール**する（コンピュータにソフトウェアを組み込む）必要がある。

(2) BIOS (Basic Input/Output System)

BIOS は、コンピュータの標準的な入出力手段を提供するソフトウェア（プログラム）である。コンピュータ起動時には、BIOS により検出された周辺装置のデバイスドライバが OS（コンピュータを制御するためのソフトウェア）によって組み込まれることにより、周辺装置を利用できるようになる。

(3) プラグアンドプレイ (PnP : Plug and Play)

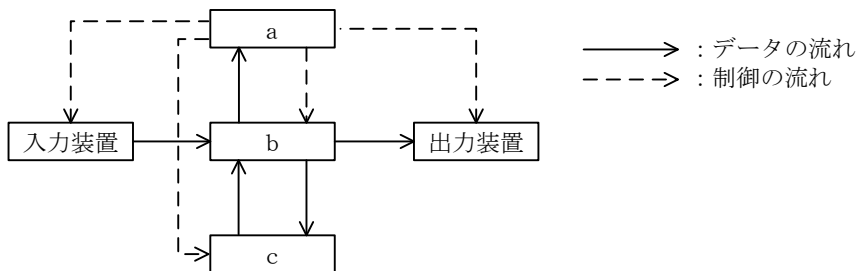
プラグアンドプレイ (PnP) とは、周辺装置をコンピュータに接続するだけで、すぐに使用できるようになる機能（仕組み）である。基本的な仕組みは、周辺装置が接続された時点で OS が接続デバイスを認識して、自動的にデバイスドライバを設定して利用できるようにする。ユーザが手動で設定しなくてもよいことから、デバイスドライバを知らない初心者でも簡単に PC や周辺装置を利用できるようになる。

なお、USB や IEEE 1394 が提供する**ホットプラグ**（電源を入れたまま機器の着脱を可能にする機能）を実現するには、対象となるコンピュータの OS がプラグアンドプレイに対応している必要がある。

第1部 演習問題

問1

コンピュータの基本構成を表す図中の a ～ c に入れるべき適切な字句の組合せはどれか。



	a	b	c
ア	演算装置	記憶装置	制御装置
イ	記憶装置	制御装置	演算装置
ウ	制御装置	演算装置	記憶装置
エ	制御装置	記憶装置	演算装置

問2

32 ビットで表現できるビットパターンの個数は、24 ビットで表現できる個数の何倍か。

- ア 8 イ 16 ウ 128 エ 256

問3

10 の整数乗倍を表す接頭語の記号 G (ギガ), k (キロ), M (メガ), T (テラ) の四つについて、その大小関係を正しく表しているものはどれか。

- ア $G < k < M < T$ イ $k < G < T < M$
 ウ $k < M < G < T$ エ $M < T < G < k$

問4

2進数の 101.11 を 10進数で表したものはどれか。

- ア 5.11 イ 5.3 ウ 5.55 エ 5.75

問5

すべて同じ値を表している 2進数, 8進数, 10進数, 16進数の組合せはどれか。

	2進数	8進数	10進数	16進数
ア	111	10	8	8
イ	1010	12	10	A
ウ	1100100	256	100	64
エ	11111111	377	256	FF

問6

EUC コードの説明として、適切なものはどれか。

- ア ANSI によって設定された 7ビットからなるコードで、8ビット目をパリティビットとして用いる。
 イ IBM 社が開発した 8ビットからなるコードで、汎用機を中心に普及している。
 ウ UNIX を中心に普及している 2バイトからなるコードで、漢字も表現できる。
 エ アップル社、ノベル社、マイクロソフト社などが提唱した 2バイトコードで、ISO によって規格化されている。

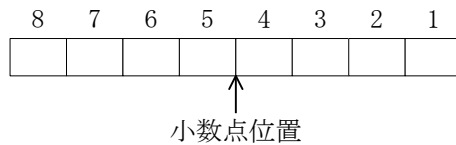
問7

数値の部分が 6桁の符号付き 10進数を、パック 10進表記法で表すとき、必要なバイト数は幾らか。

- ア 3 イ 4 ウ 6 エ 7

問8

10 進数 -5.625 を，8 ビット固定小数点形式による 2 進数で表したものはどれか。
ここで，小数点位置は，4 ビット目と 5 ビット目の間とし，負数は 2 の補数表現を用いる。



ア 01001100

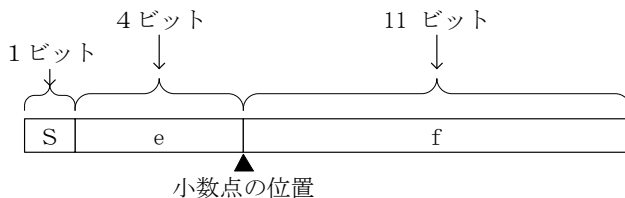
イ 10100101

ウ 10100110

エ 11010011

問9

数値を図に示す 16 ビットの浮動小数点形式で表すとき，10 進数 0.25 を正規化した表現はどれか。ここでの正規化は，仮数部の最上位桁が 0 にならないように，指数部と仮数部を調節する操作とする。



S : 仮数部の符号 (0 : 正, 1 : 負)
e : 指数部 (2 を基数とし, 負数は 2 の補数で表現)
f : 仮数部 (2 進数 絶対値表示)

ア

0	0001	100000000000

イ

0	1001	100000000000

ウ

0	1111	100000000000

エ

1	0001	100000000000

問10

2 進法で表現された正の整数を 32 倍するには，左に何ビットシフトすればよいか。
ただし，あふれは起こらないものとする。

ア 4

イ 5

ウ 6

エ 32

問11

電気信号によってデータの書換え，消去が可能なメモリであり，電源を切っても内容を保持できるものはどれか。

- ア DRAM
- イ SRAM
- ウ フラッシュメモリ
- エ マスク ROM

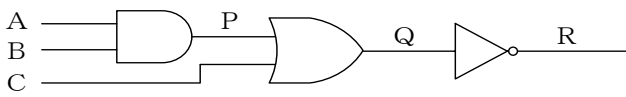
問12

命令は，命令部とアドレス部から構成されている。アドレス部から実効番地を生成する方式のうち，絶対アドレス方式のものはどれか。

- ア 基準アドレスとしてスタックポインタの値を用い，命令のアドレス部を基準アドレスからの変位として加算し，実効番地を生成する。
- イ 基準アドレスとしてプログラムカウンタの値を用い，命令のアドレス部を基準アドレスからの変位として加算し，実効番地を生成する。
- ウ 基準アドレスとしてベースレジスタの内容を用い，命令のアドレス部を基準アドレスからの変位として加算し，実効番地を生成する。
- エ 命令のアドレス部を実効番地とする。

問13

図の論理回路において， $A=1$ ， $B=0$ ， $C=1$ のとき， P ， Q ， R の値の適切な組合せはどれか。ここで， は AND 回路， は OR 回路， は NOT 回路を表す。



	P	Q	R
ア	0	1	0
イ	0	1	1
ウ	1	0	1
エ	1	1	0

問14

キャッシュメモリと主記憶に関するアクセス時間とヒット率の組合せのうち、主記憶の実効アクセス時間が最も短くなるのはどれか。

	キャッシュメモリ		主記憶
	アクセス時間 (ナノ秒)	ヒット率 (%)	アクセス時間 (ナノ秒)
ア	10	60	70
イ	10	70	70
ウ	20	70	50
エ	20	80	50

問15

あるコンピュータでは、1 命令が表のステップ 1～6 の順序で実行される。図のパイプライン処理を利用して 6 命令を実行すると、何ナノ秒かかるか。ここで、各ステップの実行時間は 10 ナノ秒とし、パイプライン処理の実行を乱す分岐命令などはないものとする。

表 命令の実行ステップ

ステップ	処理内容
1	命令コード部の取出し
2	命令の解読
3	アドレス部の取出し
4	実効番地の計算
5	データの取出し
6	演算の実行

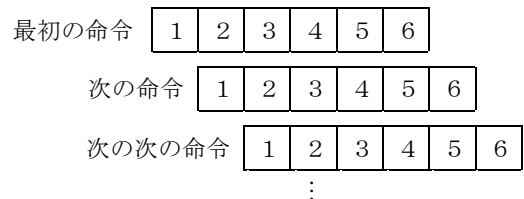


図 命令実行のパイプライン処理

- ア 50 イ 60 ウ 110 エ 300

問16

一つの命令で複数のデータに対して同じ処理を並列に行うので、マルチメディア系の処理に適しているのはどれか。

- ア MIMD イ MISD ウ SIMD エ SISD

問17

回転速度が 5,000 回転／分，平均シーク時間が 20 ミリ秒の磁気ディスクがある。この磁気ディスクの 1 トラック当たりの記憶容量は，15,000 バイトである。このとき，1 ブロックが 4,000 バイトのデータを，1 ブロック転送するために必要な平均アクセス時間は何ミリ秒か。

- ア 27.6 イ 29.2 ウ 33.6 エ 35.2

問18

デジタルカメラの画像データや携帯音楽プレーヤの音楽データの記録媒体として利用されているリムーバブルメディアはどれか。

- ア DDS イ RAM ディスク
ウ SD カード エ SSD

問19

2次元や3次元の位置情報をコンピュータに入力するための座標読取装置の一種であり，CAD の図面入力や，NC（数値制御）工作機械のデータ作成などに利用されるものはどれか。

- ア OCR イ XY プロッタ
ウ デジタイザ エ ライトペン

問20

電圧を加えると自ら発光するのでバックライトが不要なディスプレイであり，低電圧駆動，低消費電力を特徴とするものはどれか。

- ア CRT ディスプレイ イ TFT 液晶ディスプレイ
ウ プラズマディスプレイ エ 有機 EL ディスプレイ

問21

レーザプリンタの性能を表す指標として、最も適切なものはどれか。

- ア 1 インチ (2.54 cm) 当たりのドット数と 1 分間に印刷できるページ数
- イ 1 文字を印字するのに使われる縦横のドット数と 1 秒間に印字できる文字数
- ウ 印字する際の行間隔と 1 秒間に印字できる行数
- エ 印字する際の文字の種類と 1 秒間に印字できる文字数

問22

USB の説明はどれか。

- ア PC に内蔵される CD-ROM 装置, DVD 装置などを接続するためのパラレルインタフェースである。
- イ 磁気ディスク, プリンタなどをデジータチェーンで接続するパラレルインタフェースである。
- ウ ハブを介してツリー状に周辺装置を接続できるシリアルインタフェースである。
- エ プリンタなどに赤外線を使ってデータを転送するシリアルインタフェースである。

第2部

情報処理システム

1 情報処理システムの処理形態

さまざまなところで利用されているコンピュータシステムに共通していることは、何らかのデータが入力され、一定の手順で処理が行われ、その結果を出力しているということである。これらを情報処理システムと総称するが、情報処理システムは、利用方法や機器構成によってさまざまな形態に分けられる。

1-1 非対話型処理システムと対話型処理システム

(1) 非対話型処理システム

非対話型処理システムとは、コンピュータに対する指示をひとまとめにして与えるシステムである。コンピュータが処理を開始したら、利用者（人間）は処理にまったく介入できなくなる。

(2) 対話型処理システム

対話型処理システムとは、コンピュータを利用する人間が、ディスプレイに表示される情報などを見ながら、一つひとつ指示を与えてデータを処理していくシステムである。人間がコンピュータと会話しているような形で仕事を進めていく方式といえる。

対話型処理システムでは、人間もシステムの一部に組み込まれているため、利用者の生産性を高めることが必要である。そのためには、人間にとって使いやすいシステムを実現するために**ユーザインタフェース**を充実させる必要がある。代表的なユーザインタフェースとしては、図 2-1 に示す**ウィンドウ**や**アイコン**などがある。

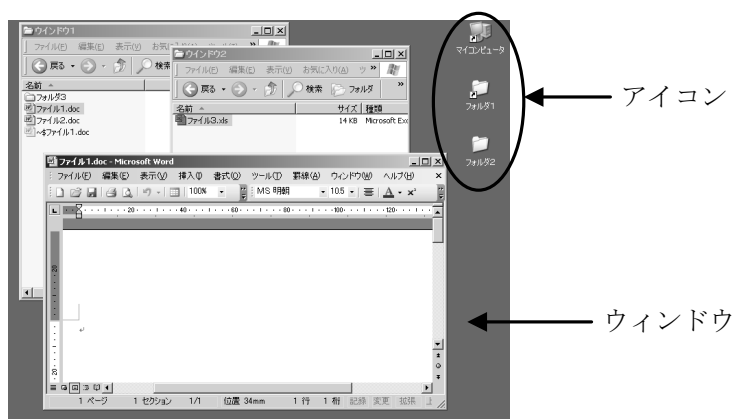


図 2-1 ユーザインタフェースの例

1-2 一括処理システムと即時処理システム

(1) 一括処理システム（バッチ処理システム）

一括処理システム（バッチ処理システム）は、コンピュータで処理するデータをいったん蓄積しておき、ある時点で一斉に処理するシステムである。即時性を要求されない定期的な業務（例えば、社員の給与計算など）で利用される。

① センタバッチ処理

処理要求の発生地点にデータを蓄積して、ある期間や期日ごとにセンタにデータを持って行き、センタのコンピュータで処理を行う方式である。

② リモートバッチ処理

遠隔地の RJE 端末（RJE：Remote Job Entry）などから、発生した処理要求をセンタのコンピュータに送信して蓄積し、定期的に処理する方式である。

(2) 即時処理システム（リアルタイム処理システム）

即時処理システム（リアルタイム処理システム）は、コンピュータに対する処理要求が発生するごとに、直ちに処理を開始するシステムである。即時性を要求される（処理時間に強い制約がある）業務や、人間の代役を務めさせる業務で利用される。

エアバッグ制御システムなどのように、応答時間に強い制約があり、時間内に処理が終了しないと、生命を脅かすような致命的なダメージをこうむる**ハードリアルタイムシステム**と、座席予約システムなどのように、応答時間に制約はあるが、万一、時間内に処理が終了しなくても、致命的なダメージとはならない**ソフトリアルタイムシステム**がある。

① リアルタイム制御処理システム

人間の代役として、センサなどによって得られた情報を逐次解釈して処理するシステムである。工業用ロボットの制御や自動操縦システムなどに利用される。

② オンライントランザクション処理システム（OLTP システム）

遠隔地で発生した処理要求（トランザクション）をセンタのコンピュータに送信して、その場で処理するシステムである。**オンラインリアルタイムシステム**とも呼ばれ、銀行の ATM システムなどで利用される。トランザクション処理においては、Atomicity（**原子性**；すべて実行されるか一つも実行されない性質）、Consistency（**一貫性**；処理によって矛盾を生じない性質）、Isolation（**分離性**；トランザクションが互いに干渉しない性質）、Durability（**持続性**；結果が保持され続ける性質）という **ACID 特性**が要求される。

③ 時分割システム（TSS：Time Sharing System）

1 台のコンピュータを、複数の利用者が、あたかも自分ひとりで独占しているかのように、対話しながら処理を行うシステムである。

1-3 集中処理システムと分散処理システム

集中処理システムとは、データや資源を1か所に集中して処理するシステムであり、初期の情報処理システムはすべてこの形態であった。これに対して、**分散処理システム**は、ネットワーク技術の進歩によって発展した考え方で、データや資源を分散させて処理することにより**負荷分散**や**機能分散**などを実現するシステムである。

それぞれのシステムの特徴をまとめると、次のようになる。

システムの形態 比較項目	集中処理システム	分散処理システム
システムの構築と運用	易しい	難しい
コンピュータの負荷	高い（1台ですべてを行う）	低い（複数台で分散する）
システムの信頼性	低い（1台の故障によってシステム全体がダウンする）	高い（1台が故障しても、フォローできる）
セキュリティの確保	易しい	難しい

1-3-1 分散処理システムの分類

分散処理システムは、接続されたコンピュータ同士の関係によって、**水平分散型**と**垂直分散型**に分類される。

(1) 水平分散型

水平分散型は、接続されたコンピュータが、すべて対等の立場にある分散処理システムの形態である。

① ピアツーピア（P2P：Peer to Peer）

対等な立場にあるコンピュータを、直接、接続する水平分散処理システムである。システムの導入が容易でコストも低く抑えられるが、すべてのコンピュータが処理要求に対応できる機能を必要とする。また、ネットワーク内の共有資源も管理しなければならないため、大規模なシステムには適していない。



図 2-2 水平分散型（ピアツーピア）のイメージ

(2) 垂直分散型

垂直分散型は、接続されたコンピュータに、明確な上下関係をもたせる分散処理システムの形態である。

① クライアントサーバシステム

ネットワークに接続されたコンピュータを、処理を依頼する側（**クライアント**）と処理を提供する側（**サーバ**）に分類する垂直分散処理システムである。クライアントサーバシステムでは、クライアントとサーバが同じコンピュータやOS（コンピュータを制御するソフトウェア）である必要はない。また、サーバは、必要に応じて処理の一部を別のサーバに依頼するクライアント機能をもつことができる。

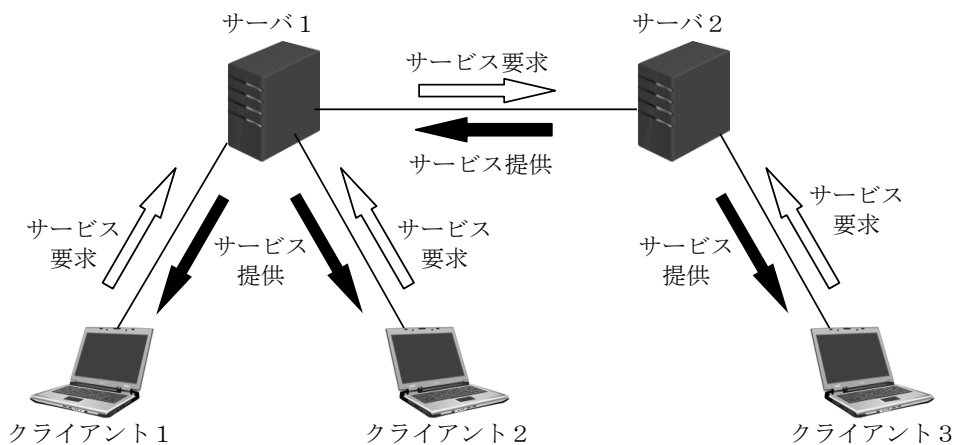


図 2-3 垂直分散型（クライアントサーバシステム）のイメージ

クライアントサーバシステムで利用される代表的なサーバの種類は、次のとおりである。なお、1台の機器に、複数のVM（Virtual Machine；**仮想マシン**）を論理的に構築する**仮想化技術**などにより、1台のサーバを複数台のサーバ（例えば、ファイルサーバとデータベースサーバなど）として利用することもできる。

- **ファイルサーバ**

大容量の補助記憶装置を備え、ファイル（データ）を一元管理する。

- **データベースサーバ**

大容量の補助記憶装置を備え、データベースを一元管理する。

- **PROXY サーバ（プロキシサーバ又は代理サーバ）**

外部ネットワーク（インターネットなど）との接続を、クライアントに代わって行う（代行する）。

※このほかにも、**プリントサーバ**、**コミュニケーションサーバ**などがある。

・ 3層クライアントサーバシステム

従来のクライアントサーバシステムは、データの加工や計算をクライアント側で行い、データベースの処理（検索や更新など）だけをサーバに依頼する**2層クライアントサーバシステム**が主流であった。しかし、2層クライアントサーバシステムでは、アプリケーションの機能拡張などを行うとき、すべてのクライアント端末に対してアプリケーションの更新作業などが必要となる。そこで、論理的に次の3層構造に分割する、**3層クライアントサーバシステム**が考えられた。

サーバ	データベースアクセス層
	ファンクション層
クライアント	プレゼンテーション層

- ① **データベースアクセス層**（データ層）
データベースにアクセスして、必要なデータを参照する。
- ② **ファンクション層**（アプリケーション層）
メッセージ（SQL 文）やデータの加工処理（計算や分析など）を行う。
- ③ **プレゼンテーション層**
利用者とデータをやり取りするためのユーザインタフェース（データ入力や結果表示）を実現する。

3層クライアントサーバシステムは、それぞれの機能を独立して管理することができる。また、ほとんどの機能をサーバ側にもつため、アプリケーションの開発作業や機能拡張の面からも効率的である。さらに、データベースアクセス層とファンクション層の機能をサーバが受け持つことで、クライアントとサーバ間の通信量を削減して通信負荷を軽減する効果もある。

〔3層クライアントサーバシステムに関連する機能〕

・ ストアドプロシージャ機能

利用頻度の高い命令群（データベースの検索や更新を行う、一連のSQL 文など）を、あらかじめサーバに用意して利用する方式である。データベースサーバを利用する2層クライアントサーバシステムで、クライアントとサーバ間のSQL 文通信がネットワーク負荷となることを解消するために考えられたが、3層クライアントサーバシステムでも利用されている。

・ グループコミットメント機能

複数の操作を並行して行うサーバ環境において、同期をとって複数の書き込み動作を行う機能である。

・シンククライアントシステム

サーバ側にアプリケーションなどを置き、クライアント側には最低限の機能だけもたせる考え方を**シンククライアント**という。シンククライアントシステムでは、この考え方を利用している（シンククライアントの仕組み（システム）を実現するソフトウェア又はサーバのことを**シンククライアントエージェント**という）。シンククライアントシステムでは、クライアント側にアプリケーション（プログラム）を記録する必要がないため、**シンククライアント端末**には補助記憶装置がない（データなどは、ネットワークを介して利用できる記憶装置である**ネットワークストレージ**などに記録する）。そのため、情報漏えいを防止する効果も期待できる。また、シンククライアントシステムのように、サーバ側のアプリケーション（プログラム）が提供する手続を、クライアント側が自分のコンピュータ上にあるかのように利用できる通信方式を**RPC**（Remote Procedure Call；遠隔手続呼出し）という。RPCは、プログラム間の通信方式の一つであり、処理の一部を他のコンピュータに委託することができる方式である。

・ファットクライアント

アプリケーションやデータなどを、すべてクライアント側に置いておく従来の方式である。

・リッチクライアント

必要に応じてアプリケーションをクライアント側にダウンロードして利用する方式である。

② Web システム

Web サービスと呼ばれるインターネットの情報発信サービスを提供する**Web サーバ**に、**Web ブラウザ**（Web サービスから情報を検索して表示するソフトウェア）から処理を依頼するシステムである（クライアントサーバシステム的一种といえる）。代表的なWebシステムには、データベースから情報を検索するシステムがある。この場合、3層クライアントサーバシステムのデータベースアクセス層をデータベースサーバ、ファンクション層を**アプリケーションサーバ**（Webサーバ）に分割する構成が用いられることが多い。クライアントは**HTML**（HyperText Markup Language；Web ページ用の記述言語）の記述に従って、受け取った結果を画面に表示するプレゼンテーション層の役割だけを担当する。そのため、Webシステムでは、最低限の機能だけをもたせたシンククライアント端末が用いられることが多い。

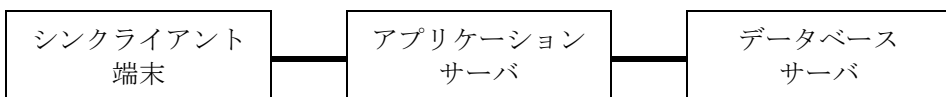


図 2-4 Web システムの構成例

③ クラウドコンピューティング

コンピュータの資源（ハードウェア、ソフトウェアなど）をネットワーク経由で提供することにより、利用者がスケーラビリティ（拡張性）やアベイラビリティ（可用性）の高いサービスを容易に受けられる形態（システム）である。クラウド（雲）とは、利用者が雲の向こうの仕組み（サービスを提供する仕組み）を知らなくても利用できるという意味を表している。利用者は必要最低限の環境（PCなどのクライアント端末、Webブラウザ、インターネット接続環境など）があれば、いつでもサービスを利用することができる。サーバなどの機器（ハードウェア）やソフトウェアを自身で運用する**オンプレミス**（自社運用）よりも導入は容易であるが、自社特有（固有）のサービスを実現しにくいなどのデメリットもある。

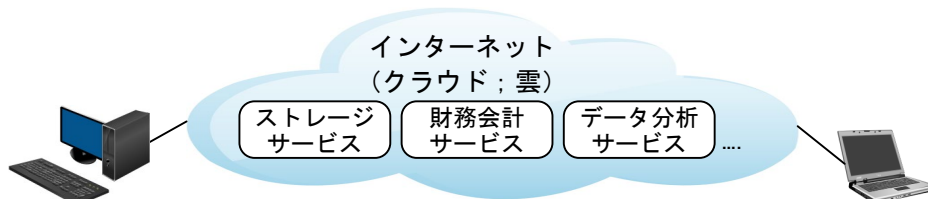


図 2-5 クラウドコンピューティングのイメージ

【クラウドコンピューティングによって提供される代表的なサービス】

- **SaaS** (Software as a Service)
ソフトウェアの機能を提供するサービス形態である。一般的に、複数の利用者が共同利用する**マルチテナント方式**が採用されている。
- **PaaS** (Platform as a Service)
アプリケーション実行用のプラットフォームを提供するサービス形態である。仮想化されたサーバや OS・ミドルウェア (DBMS) などが提供され、利用者が自分のアプリケーションを配置して運用する。
- **IaaS** (Infrastructure as a Service)
インフラを提供するサービス形態である。仮想化されたサーバだけが提供され、必要な OS やアプリケーションなどは利用者が準備する。
- **DaaS** (Desktop as a Service)
VDI (Virtual Desktop Infrastructure : デスクトップ仮想化) を併用して、デスクトップ環境を提供するサービス形態である。
- **BaaS** (Backend as a Service)
スマートフォンなどのモバイル端末で動作する、モバイルアプリやゲームのバックエンド機能を提供するサービス形態である。
- **FaaS** (Function as a Service)
機能 (Function) の実行環境を提供して、サーバレスの (サーバを意識しない) アプリケーション開発を実現するサービス形態である。

1-3-2 並列処理システム

並列処理システムは、分散処理を並列して行う（並列処理する）ことにより、処理時間の短縮や大規模な計算処理などを実現するシステムである。

① マルチプロセッサシステム

複数のコンピュータを組み合わせたシステムの総称である。ただし、単に“マルチプロセッサ”という場合は、複数のプロセッサ（プロセッサコア）が搭載された1台のコンピュータを表すこともある。なお、クラスタやグリッドコンピューティングも、広い意味ではマルチプロセッサシステムに含まれる。

② クラスタ（クラスタシステム）

複数のコンピュータを連携させて、全体を1台の高性能コンピュータであるかのように利用するシステムである（このようなシステムを構築する方式を**クラスタリング**という）。クラスタは、システムを構成する一部のコンピュータで障害が発生しても、ほかのコンピュータに処理を肩代わりさせることによってシステム全体の停止を防止できるため、信頼性の高いシステムとなる。

- ・ **負荷分散クラスタ構成（アクティブ-アクティブ構成）**

ロードバランサによって、複数のコンピュータに処理を分散する。

- ・ **HA クラスタ構成（アクティブ-スタンバイ構成）**

HAは、“High Availability”の略であり、主系コンピュータと同一仕様の待機系コンピュータを準備することで可用性が高められる。

③ グリッドコンピューティング

広域に分散する多数のコンピュータをネットワークで結び、並列処理することにより、仮想的に高性能なコンピュータとして利用するシステム（仕組み）である。ネットワークに接続されたコンピュータ資源（CPU、データベース、アプリケーションなど）を、その仕組みを意識せずに、いつでも、どこからでも利用できる方式である。自然現象を解明するためなどに行われる、非常に計算量の多い処理である**ハイパフォーマンส์コンピューティング**（HPC: High Performance Computing）などに活用されている。ハイパフォーマンส์コンピューティングは**大規模並列処理**とも呼ばれ、ヒトゲノム（遺伝子）を解明するプロジェクトなどがある。

【マイグレーション】

マイグレーションは、環境（プラットフォーム）を移行することである。仮想マシンで稼働しているOS・プログラムを停止させずに別の物理マシンに移行する**ライブマイグレーション**や、メインフレームなどを利用した汎用機系システムをオープンシステムに移行する**レガシーマイグレーション**などがある。

2 高信頼化システムの構成

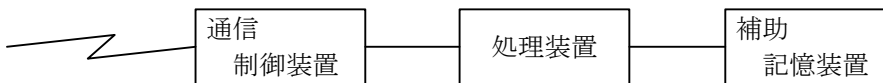
コンピュータや周辺機器で構成されたシステムにおいて心配しなければならないのは、機器の故障である。人間が作り出した機器である以上、絶対に故障しないということはないので、システムの重要度を考慮して、システム構成を考える必要がある。

2-1 直列システム

直列システムは、障害に対する予備システムを準備しないシステム構成である。コストは抑えられるが、信頼性が低いため、機器の故障がシステム停止につながっても問題にならないシステムで用いられる。

(1) シンプレックスシステム

シンプレックスシステムは、すべての装置を直列に接続したシステムである。オンライントランザクション処理システムにおける基本的な構成といえる。

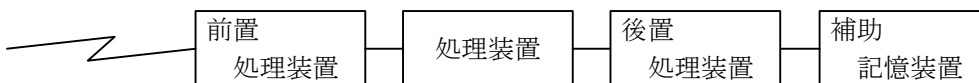


通信制御装置：CCU (Communication Control Unit)
複数の通信回線をまとめて取り扱い、データの組立て／分解やエラーチェックを行う装置

図 2-6 シンプレックスシステムの構成

(2) タンデムシステム

タンデムシステムは、処理装置の負荷を分散するために、複数の処理装置を直列に接続したシステムである。複数の処理装置のうち、1 台でも故障すると、システム全体が機能しなくなってしまう。



前置処理装置：FEP (Front End Processor)
主として、通信制御を専用に行う処理装置
後置処理装置：BEP (Back End Processor)
主として、データベース管理を専用に行う処理装置

図 2-7 タンデムシステムの構成

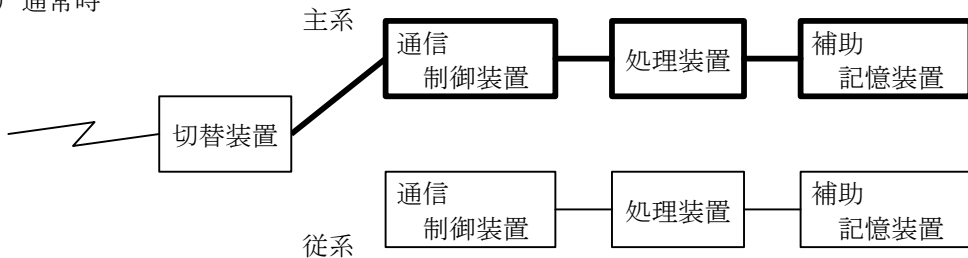
2-2 並列システム

並列システムは、あらかじめ予備システムを用意しておくシステム構成である。コストはかかるが、システムが故障すると社会的な影響を及ぼすような、高い信頼性を求められるシステムで利用される形態である。

(1) デュプレックスシステム

デュプレックスシステムは、予備システムを用意しておき、障害発生時に切り替える方式である。このシステムは、通常使用されている**主系（現用系）**と、故障に備えて待機している**従系（待機系）**から構成される。主系でオンライン業務、従系でバッチ処理などのローカル業務を行うのが一般的である。

a) 通常時



b) 障害発生時

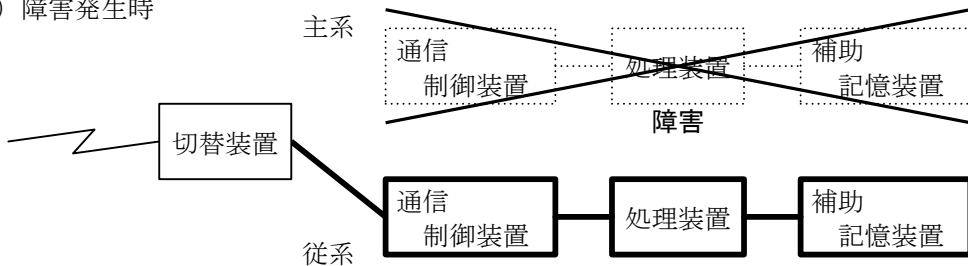


図 2-8 デュプレックスシステムの構成

- ・ **コールドスタンバイ方式**

従系（待機系）で別のローカル業務などを実行させておく、又は完全に停止させておく方式である。主系（現用系）で障害が発生したときに、従系（待機系）で主系（現用系）の業務システムを起動してから、処理を引き継ぐ。

- ・ **ホットスタンバイ方式**

従系（待機系）で主系（現用系）と同じ業務システムを起動して、待機させておく方式である。従系（待機系）は主系（現用系）を監視して、障害が発生したときに主系（現用系）の処理を直ちに引き継ぐ。

(2) デュアルシステム

デュアルシステムは、2系以上のシステムで完全に同じ処理を行い、結果を照合することにより正常性を確認（**クロスチェック**）する方式である。トランザクション当たりのコストはデュプレックスシステムよりも高くなるが、一つの系が故障しても残りの系で処理を続行するため信頼性は非常に高くなる。医療システムなどのように、絶対に停止することができない、人命に関わるようなシステムに利用される。

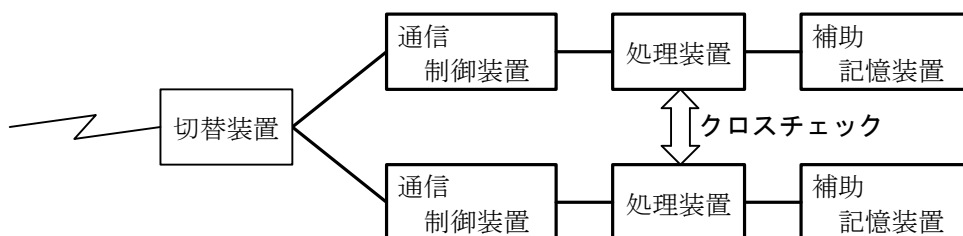


図 2-9 デュアルシステムの構成

【バックアップサイト】

Web システムやクラウドコンピューティングのように Web サイト（Web ページのまとまり）を利用する場合、万一の障害に備えて**バックアップサイト**を準備しておく。バックアップサイトは障害発生時に切り替えるので、Web サイトのデュプレックスシステムといえる。バックアップサイトには、障害からの復旧時間が短い順に次のような種類がある。

- **ホットサイト**

同一システムで構成される予備系サイトのデータなどを、ネットワークを介して常に最新の状態に更新し、稼働状態で待機させる。

- **ウォームサイト**

予備系サイトに、必要なデータなどを定期的に搬入して保管する。

- **コールドサイト**

予備系サイトだけを準備し、必要なデータなどは障害発生時に搬入する。

また、負荷分散などを目的とした Web サイトの複製（**ミラーサイト**）は、障害が発生した Web サイトを切り離してサービスを継続できるので、デュアルシステムの一つともいえる（クロスチェックは行わないので厳密には異なる）。

2-3 多重化システム

多重化システムは、複数の機器を使用することによって、信頼性を高めるシステム構成である。並列システムは、システム全体を対象にした多重化システムといえる。

(1) フォールトトレラントシステム

フォールトトレラントシステムは、システムの一部に障害が発生しても、システム全体としては必要な機能を維持できるように設計されたシステムのことである。

① ハードウェア的な実現方法

システムを構成するコンピュータ（サーバ）や磁気ディスクを二重化して、**可用性**（利用要求時に使用可能である特性）を向上させる。

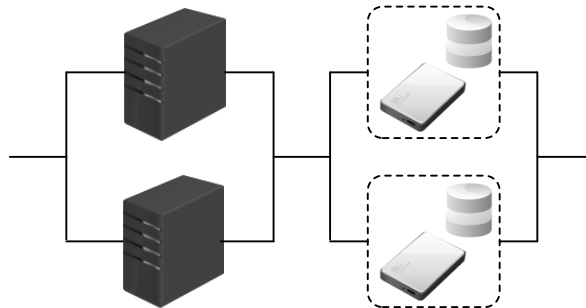


図 2-10 ハードウェア的な実現方法

② ソフトウェア的な実現方法

同一の機能をもつ複数のプログラムを同時に実行して結果を比較し、過半数を占めたものを採用する**Nバージョンプログラミング**などを利用する。

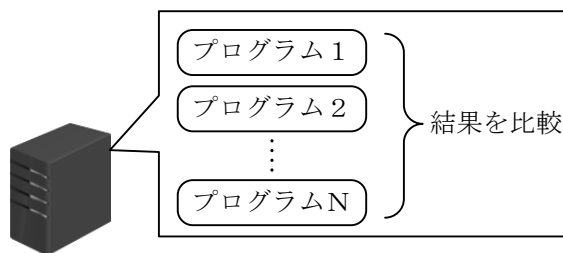


図 2-11 Nバージョンプログラミング

また、いくつかの機能を犠牲にしても、主となる機能を停止させないシステムの縮退運転などの**フェールソフト構造**も、フォールトトレラントシステムの一つである。

(2) マルチプロセッサシステム（多重プロセッサシステム）

マルチプロセッサシステム（多重プロセッサシステム）は、処理装置（プロセッサ）を多重化したシステムである。各処理装置は、一つの目的のために並列処理を行う。また、障害発生時には該当する処理装置を切り離し、残った処理装置で分担して処理を続行することができるため、信頼性も向上する。

マルチプロセッサシステムでは、多重化する処理装置を増やすごとに、処理性能（処理速度）は向上する。ただし、“プログラムの並列処理における処理性能の向上は、逐次処理部分によって制限される”という**アムダールの法則**があるため、処理装置の台数を n 倍に増やしても、処理速度が n 倍（処理時間が $1/n$ ）になるとは限らない。

① 密結合マルチプロセッサシステム

同一の OS（コンピュータを制御するソフトウェア）により制御された処理装置が、共有する主記憶装置を使用しながら処理を行うシステムである。システム内のタスク（処理する仕事を細分化した単位）を、どの処理装置でも実行することができる。結果的に、細かい単位で負荷を分散できることになるが、タスク間で**同期**（処理などのタイミング）をとる機能が必要になる。

② 疎結合マルチプロセッサシステム

それぞれの OS により制御された処理装置が、個々の主記憶装置を使用しながら処理を行うシステムである。主記憶装置の情報などを高速入出力ポートで伝達することにより、処理の整合性をとる。**クラスタ**は、この方式の一つといえる。

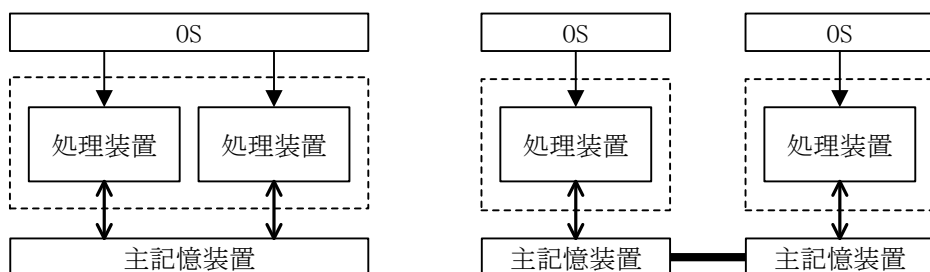


図 2-12 密結合／疎結合マルチプロセッサシステムのイメージ

・ポリプロセッサシステム

処理装置ごとに役割が分担されているマルチプロセッサシステムである。

・ロードシェアリングシステム（負荷分散システム）

状況に応じて仕事（負荷）を分散させるマルチプロセッサシステムである。

(3) RAID (Redundant Arrays of Inexpensive Disks)

RAID は、磁気ディスク装置の多重化システム又は装置多重化技術である。RAID は複数のハードディスクを一体化して扱えるようにして信頼性を高め、障害が発生したときも、システムを停止することなくアクセスできる。データと冗長ビット（誤り検出・修復に使用するビット）の記録方法・位置により、次のように分類されている。

- **RAID0（ストライピング又はディスクストライピング）**
複数のディスクに一つのデータを分割して書き込むことにより、アクセス速度の向上を図る。信頼性を高めるための予備ディスクはもっていない。
- **RAID1（ミラーリング又はディスクミラーリング）**
同じ容量をもつディスク 2 台に同じタイミングで同一内容を書き込み、1 台をバックアップに使用することで信頼性の向上を図る。データを二重化するため、ディスクの使用効率は全容量の 50%となる。
- **RAID2**
データ保存用の複数台のディスクとデータ修復（**ハミング符号**）用のディスクを使用して、アクセス速度の高速性と信頼性の向上を図る。
- **RAID3**
データ保存用の複数台のディスクとデータ修復（**パリティ符号**）用のディスクを使用して、アクセス速度の高速性と信頼性の向上を図る。
- **RAID4**
RAID3 ではビット単位で行われるデータアクセスを、セクタ単位（ブロック単位）で行うことによってアクセス速度の高速性と信頼性の向上を図る。
- **RAID5**
RAID4 において、データ修復用のパリティディスクに記録されるパリティデータを、データ保存用のディスクに分散した構成である。パリティディスクの負荷を少なくし、アクセス速度の高速性と信頼性の向上を図る。
- **RAID6**
RAID5 において、データ修復用のパリティ符号を二つ生成する方式である。

RAID は、**ネットワークストレージ**にも利用される。代表的なネットワークストレージには、**NAS** (Network Attached Storage) や **SAN** (Storage Area Network) などがある。

NAS は、直接ネットワークに接続してファイルサーバとして利用されるストレージである。Windows 系の CIFS や UNIX 系の NFS など、複数のプロトコルに対応しているファイルシステムがストレージに内蔵されているため、異なる OS や機種（サーバ）間でのデータ共有がファイル単位で簡単に行える。一方、SAN は、データストレージ専用のネットワークである。ネットワークに対する負荷が NAS よりも小さいが、ファイルシステムの異なる OS や機種（サーバ）間ではデータ共有が難しいという短所がある。

3 情報処理システムの評価

情報処理システムを運用するには、あらかじめそのシステムの機能について評価する必要がある。しかし、情報処理システムの形態は多種多様であり、その使用目的も異なるため、一つの指標だけで評価することは難しい。そこで、さまざまな観点から情報処理システムを評価する指標が必要となってくる。

3-1 処理能力の評価

情報処理システムの処理能力を評価する指標としては、システム全体としての処理能力を評価する指標と、システムの中核となる CPU の処理能力を評価する指標がある。

3-1-1 システム全体の評価

(1) 処理時間に関する評価指標

システムの処理時間に関する評価指標としては、**ターンアラウンドタイム**と**レスポンスタイム（応答時間）**がある。基本的に、これらの時間が短いほど評価は高くなる。

- ・ **ターンアラウンドタイム**

システムに対して処理依頼を開始してから、完全な出力結果を得られるまでの経過時間である。バッチ処理システムの性能評価の指標として用いられる。

例. バッチ処理システムのターンアラウンドタイム

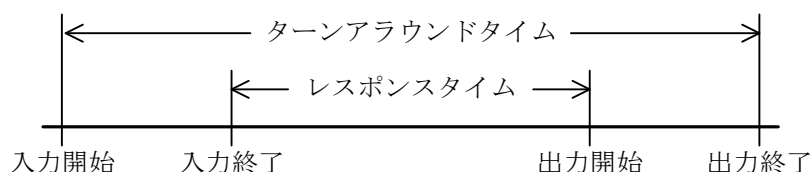
=CPU 処理時間+入出力時間+オーバーヘッド時間（処理待ち時間など）

- ・ **レスポンスタイム（応答時間）**

システムに処理依頼が完了してから、結果の出力が開始されるまでの経過時間である。オンラインシステムなどの性能評価の指標として用いられる。

例. オンライントランザクション処理システムのレスポンスタイム

=CPU 処理時間+オーバーヘッド時間（回線伝送時間、端末処理時間など）



(2) 仕事量に関する評価指標

システムの仕事量に関する評価指標としては、**スループット**がある。基本的に、処理できる仕事量が多いほど評価は高くなる。

・スループット

情報処理システムにおいて、単位時間（一定時間）内に処理される仕事の量のことである。単位として、**TPS**（Transaction Per Second；1秒間に処理できるトランザクション数）などが利用される。例えば、1トランザクションを処理するのにCPU時間が5ミリ秒かかるWebサーバで、CPU使用率の上限を80%とした場合、このWebサーバのスループットは、次のように求められる。

$$\begin{aligned}\text{Webサーバのスループット} &= 1 \text{ 秒} \div 5 \text{ ミリ秒} / \text{トランザクション} \times 0.8 \\ &= 160\text{TPS} \text{ (トランザクション/秒)}\end{aligned}$$

ただし、これはWebサーバ単体のスループットであり、複数の装置（サーバ）を組み合わせた情報処理システム全体のスループットは、処理能力が最も低い装置（サーバ）のスループットとなる（このように性能向上などの妨げになる装置や要因のことを**ボトルネック**という）。

なお、ターンアラウンドタイム／レスポンスタイム、スループットは、稼働している情報処理システムを評価する以外に、情報処理システムを開発・増強するときの目標値（指標）としても利用される。情報処理システムの開発・増強に当たっては、必要とされる目標値（システム性能）を設定し、その目標値を実現するために必要となる各種性能要件やシステム構成（増強計画）を立案する。これを、**キャパシティプランニング**という。

【キャパシティプランニングの基本手順】

1) ワークロード収集

システムに求められる処理の種類や量、処理時間などの情報（特にピーク時の負荷情報が重要）を収集する。

2) システムの性能要件の決定

収集した情報を分析し、**負荷予測**やプロトタイプ（試作品）の作成などによって、システムに求められる性能要件を決定する。

3) サイジング

システムに求められる性能要件を満たすために必要となるハードウェア、ソフトウェア、周辺機器の性能要件などを見積もる。

4) 評価／改善

サイジングの結果を評価し、必要に応じて**スケールアウト**（機器の増設）や**スケールアップ**（機器の性能向上）などの改善（チューニング）を行う。なお、システム稼働後も継続的に評価／改善を繰り返し、ストレージの記憶容量やシステムの処理能力に不足がないかを管理する。

3-1-2 CPU の評価

(1) クロック周波数

クロック周波数は、クロック信号が発生する速さを表すものである。CPU の動作はこの信号に合わせて実行されるため、クロック周波数が大きいほど基本的には処理能力が高いことになる。ただし、CPU によって **CPI** (Cycles Per Instruction ; 1 命令の実行に必要なとなるクロック数) が異なることなどもあるため、クロック周波数だけでは CPU の性能を比較 (評価) できない。

(2) MIPS (Million Instructions Per Second)

MIPS は、1 秒間に実行できる命令数を百万命令の単位で表したものである。1 MIPS の CPU とは、1 秒間に 100 万命令実行できる CPU ということである。基本的には、実行できる命令数が多いほど評価は高くなる。ただし、実行する命令の内容が違う異機種の CPU を比較するのに、この指標をそのまま使用しても意味がない。

MIPS は、1 命令の実行時間又は命令を受け付けてから次の命令を受け付けることができるまでの**サイクルタイム**により決定する。ただし、命令は種類により実行時間が異なるため、1 命令を実行するための平均実行時間を求めるには命令の出現頻度により加重平均を求める方法が使用される。このとき、よく使用される命令とあまり使用されない命令があるため、業務などの処理に応じた標準的な命令の出現頻度として**命令ミックス**が使用される。代表的な命令ミックスとしては、事務処理計算で頻繁に使用される転送命令を中心に**したコマーシャルミックス**や、科学技術計算で頻繁に使用される演算命令を中心に**したギブソンミックス**などがある。また、1 命令の平均実行時間は、CPI とクロック周波数から求めることもできる。

例. 500MHz のクロック周波数で動作する CPU が 1 命令を実行するのに平均 5 CPI を必要とするとき、この CPU の性能は何 MIPS か。

- ① 1 命令の平均実行時間
= $\text{CPI} \times \text{クロック時間 (基本動作時間)}$
= $5 \text{ クロック} / \text{命令} \times (1 \text{ 秒} \div 500,000,000 \text{ クロック})$
= $0.00000001 \text{ 秒} / \text{命令}$
- ② 1 秒間に実行できる命令数
= $1 \text{ 秒} \div 1 \text{ 命令の平均実行時間}$
= $1 \text{ 秒} \div 0.00000001 \text{ 秒} / \text{命令}$
= $100,000,000 \text{ 命令} \rightarrow 100\text{MIPS}$

また、MIPS という単位を展開すると、1 MIPS の CPU は 1 マイクロ秒に 1 命令を実行できることがわかる。このことから、MIPS の計算は、次のようにまとめることができる。

$$\begin{aligned} 1\text{MIPS} &= 1 \times 10^6 \text{ 命令 / 秒} \\ &= \frac{1 \times 10^6 \text{ 命令}}{1 \text{ 秒}} = \frac{1 \times 10^6 \text{ 命令}}{1 \times 10^6 \text{ マイクロ秒}} = \frac{1 \text{ 命令}}{1 \text{ マイクロ秒}} \\ &= 1 \text{ 命令 / マイクロ秒} \end{aligned}$$

- ・ x (MIPS) の CPU の 1 命令の平均実行時間は、 $1/x$ (マイクロ秒) である。
- ・ 1 命令の平均実行時間が y (マイクロ秒) の CPU の性能は、 $1/y$ (MIPS) である。

例. ある CPU の命令ミックスが表に示す値のとき、この CPU の性能は何 MIPS か。

命令の種類	命令実行時間	出現頻度
レジスタ間演算	0.1 マイクロ秒	40%
メモリ・レジスタ参照	0.3 マイクロ秒	50%
無条件分岐	0.6 マイクロ秒	10%

- ① 1 命令の平均実行時間

$$\begin{aligned} &= \Sigma (\text{命令実行時間} \times \text{出現頻度}) \\ &= 0.1 \text{ マイクロ秒} \times 0.4 + 0.3 \text{ マイクロ秒} \times 0.5 + 0.6 \text{ マイクロ秒} \times 0.1 \\ &= 0.04 \text{ マイクロ秒} + 0.15 \text{ マイクロ秒} + 0.06 \text{ マイクロ秒} \\ &= 0.25 \text{ マイクロ秒} \end{aligned}$$
- ② CPU の性能 (MIPS)

$$\begin{aligned} &= 1 \div 1 \text{ 命令の平均実行時間 (マイクロ秒)} \\ &= 1 \div 0.25 \\ &= 4 \text{ (MIPS)} \end{aligned}$$

なお、現在では、1 秒間に実行できる命令数を十億 (10^9) 命令の単位で表した **GIPS** (Giga Instructions Per Second) が用いられることもある。

(3) FLOPS (FLoating-point Operations Per Second)

FLOPS は、1 秒間に実行できる浮動小数点数演算の命令数を表す単位である。1 FLOPS の CPU とは、1 秒間に 1 回の浮動小数点数演算が実行できる CPU という意味である。現在はスーパーコンピュータなどの性能評価指標として、1 秒間に実行できる浮動小数点数演算の命令数を 1 兆 (10^{12}) 命令の単位で表す **TFLOPS** (Tera FLOPS) が使用されている。

3-1-3 性能測定の技法

(1) ベンチマークテスト

ベンチマークテストとは、使用目的に合わせた評価用標準プログラム（ソフトウェア）を実行させて、処理性能（所要時間など）を測定する方式である。測定すること（又は測定基準）を**ベンチマーク**、利用するプログラムのことを**ベンチマークプログラム**という。それぞれのベンチマークテストは測定対象が異なるため、複数のベンチマークテストを実行することでシステムの特徴を統合的に理解する必要がある。

- **TPC ベンチマーク**

OLTP システムを評価するため、その標準化を進めているトランザクション処理性能評議会（TPC）が定めたベンチマークで、業務に応じて TPC-A～D が定められている。評価指標として、TPS や単位性能当たりのシステム価格（cost/TPS）を用いる。現在使用されている TPC-C（業務処理 OLTP 用）では、1 分間を基準とした単位（tpmC, Price/tpmC）が用いられている。

- **SPEC ベンチマーク**

CPU の性能を評価するため、業界標準化団体であるシステム性能評議会（SPEC）が定めたベンチマークで、UNIX 上でベンチマークテストを実行できる。整数型演算を中心とした **SPECint**、浮動小数点数型演算を中心とした **SPECfp** などがある。

- その他のベンチマーク

- Dhrystone：固定小数点数演算評価用
- Whetstone：浮動小数点数演算評価用
- Livermore Fortran Kernel：ベクトルコンピュータ／コンパイラ評価用
- Linpack：浮動小数点数演算評価用（連立一次方程式解法プログラム）

(2) その他の性能測定の技法

- ① **カーネルプログラムテスト**

基本的な計算問題などの簡単なプログラムを実行し、実行時間を測定して、CPU の性能を比較する方式である。

- ② **モニタリング**

測定の対象となる情報処理システムに、あらかじめ計測用の仕組みを組み込んでおき、実際の稼働状況を測定する方式である。

- **ソフトウェアモニタリング**：測定用プログラムを組み込む方式
- **ハードウェアモニタリング**：測定用機器を組み込む方式

- ③ **静的積算**

プログラムを実際には実行せずに、机上でシステムの処理を解析し、個々の命令の出現回数や実行回数の予測値から処理時間を推定し、性能を評価する方式である。

3-2 信頼性の評価

3-2-1 信頼性の考え方

現代社会には、さまざまな情報処理システムが運用されている。逆にいえば、情報処理システムなくして現代社会は成立しないともいえる。このように重要な役割を担っている情報処理システムには、当然高い信頼性が要求される。信頼性の高いシステムを構築するための考え方は、次のように分類される。

(1) 故障に対する考え方

① フォールトトレラント（耐故障技術）

人が作製した機器やシステムである以上、故障は必ず発生すると考えて、故障の影響を最小限に抑えるための仕組みを作るという考え方である。この考え方で構築されたシステムを、**フォールトトレラントシステム**という。

・フェールソフト

フォールトトレラントの一種で、故障の影響を一部に限定して運転を継続する考え方である。病院が停電した場合に、照明を最小限にして生命維持装置などに自家発電の電力を集中する**縮退運転**（いくつかの機能を犠牲にしても、主となる重要な機能を停止させない）などが該当する。

② フォールトアボイダンス（故障排除技術）

システムに故障を絶対に発生させないという考え方である。現実には故障しないということはありません。故障が発生する確率を限りなく0に近づけていく考え方（技術）といえる。

(2) 安全に対する考え方

① フェールセーフ

障害が発生した場合に、安全性を重視して他の部分に影響を与えないようにシステムを停止させる考え方である。故障が発生した場合に、すべての信号を赤信号にして事故を防ぐ、交通信号機の制御システムなどが該当する。

② フールプルーフ

システムを操作する人間の誤りがシステム全体に影響を及ぼさないように、操作者の誤りを未然に防ぐ考え方である。ふたを閉めないで動かない洗濯機や、入力データをチェックして誤りがある場合には再入力をさせるという考え方が該当する。

3-2-2 信頼性の指標

情報処理システムの信頼性などを評価する指標として、次の五つの項目の頭文字を並べた **RASIS** 又は上から 3 項目の頭文字を並べた **RAS** がある。

指 標	意 味
Reliability (信頼性)	システムが正しく動作しているか。
Availability (可用性)	使いたいときにシステムが稼働状態になっているか。
Serviceability (保守性)	障害が発生したときに対処（回復）しやすいか。
Integrity (完全性)	データの正当性が保証されているか。
Security (安全性)	システムやデータに対する機密保護が行われているか。

この五つの指標のうち、R (Reliability ; 信頼性), A (Availability ; 可用性), S (Serviceability ; 保守性) に関しては、次の尺度を使用して評価する。

- ① **MTBF** (Mean Time Between Failures ; **平均故障間動作時間** 又は平均故障間隔)
システム（装置）の故障が修復されてから、次に故障するまでの平均動作時間（システムが正常に稼働している時間の平均）のことで、R (Reliability ; 信頼性) を表すために使用される。
- ② **MTTR** (Mean Time To Repair ; **平均修理時間**)
システム（装置）が故障してから修復されるまでにかかる時間の平均で、S (Serviceability ; 保守性) を表すために使用される。
- ③ **稼働率**
システム（装置）が稼働している確率のことで、A (Availability ; 可用性) を表すために使用される。稼働率は、単位時間当たりにシステムが正常に稼働している割合を示すものであり、MTBF と MTTR を用いて計算できる。

$$\text{稼働率} = \frac{\text{MTBF}}{\text{MTBF} + \text{MTTR}}$$

なお、システム（装置）が稼働している確率（稼働率）に対して、システム（装置）が稼働していない確率のことを**非稼働率**という。非稼働率は、単位時間当たりにシステムが正常に稼働していない（故障して修理されている）割合を示すものであり、次のように計算できる。

$$\text{非稼働率} = 1 - \text{稼働率} = \frac{\text{MTTR}}{\text{MTBF} + \text{MTTR}}$$

例. ある装置を 600 時間稼働させたときの動作状況が表のとおりであった。この装置の MTBF, MTTR, 稼働率を求めよ。

時間帯	経過時間数	状態
0～120 時間	120 時間	正常動作
120～140 時間	20 時間	修理
140～290 時間	150 時間	正常動作
290～300 時間	10 時間	修理
300～480 時間	180 時間	正常動作
480～510 時間	30 時間	修理
510～600 時間	90 時間	正常動作

① MTBF (平均故障間動作時間)

= 全稼働時間 ÷ 故障回数

= (120 時間 + 150 時間 + 180 時間 + 90 時間) ÷ 3

= 180 時間

② MTTR (平均修理時間)

= 全修理時間 ÷ 故障回数

= (20 時間 + 10 時間 + 30 時間) ÷ 3

= 20 時間

③ 稼働率

$$= \frac{\text{MTBF}}{\text{MTBF} + \text{MTTR}}$$

$$= \frac{180 \text{ 時間}}{180 \text{ 時間} + 20 \text{ 時間}}$$

= 0.9

※この例では、図 a の動作状況が、図 b のように平均化されている。

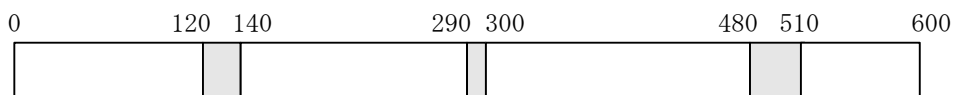


図 a ある装置の動作状況

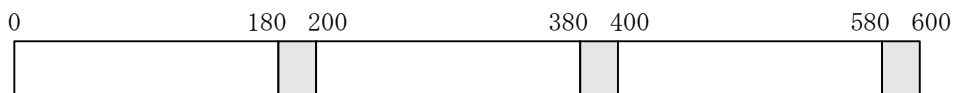


図 b 平均化された状況

3-2-3 システムの稼働率と故障率

システムは複数の装置で構成されるのが一般的である。このとき、各装置の稼働率からシステム全体の稼働率を求めることができる。

(1) 直列システムの稼働率

直列システムの場合、すべての装置が稼働していないと、システム全体として稼働していないことになる。図 2-13 の直列システムについて考えると、各装置の動作状況は表のようにまとめられる。



図 2-13 直列システムの例

	装置 1	装置 2
①	○稼働	○稼働
②	○稼働	×故障
③	×故障	○稼働
④	×故障	×故障

この表でシステム全体として稼働しているのは、①の二つの装置が稼働状態にある場合だけである。したがって、直列システムの稼働率は、次の式で求められる。

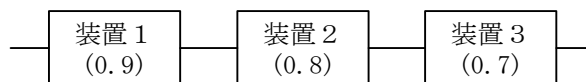
直列システムの稼働率＝装置 1 の稼働率×装置 2 の稼働率

同様に、装置が 3 台、4 台と増えても、全装置が稼働していなければ直列システムは稼働しない。そのため、直列システムの稼働率は、構成する全装置の稼働率を乗算した結果として求められる。

n 台の装置を接続した直列システムの稼働率
＝装置 1 の稼働率×装置 2 の稼働率×・・・×装置 n の稼働率

なお、稼働率は 1 未満なので、乗算を繰り返すほど値は小さくなる。つまり、直列システムでは装置を多く接続するほど、稼働率は低くなっていくことになる。

例. 下図のシステムの稼働率を求めよ。なお、() 内の数値は各装置の稼働率である。



直列システムの稼働率
＝装置 1 の稼働率×装置 2 の稼働率×装置 3 の稼働率
＝0.9×0.8×0.7
＝0.504

(2) 並列システムの稼働率

並列システムの場合は、すべての装置が稼働していなくても、システム全体として稼働している場合がある。図 2-14 のような並列システムについて考えると、各装置の動作状況は表のようにまとめられる。

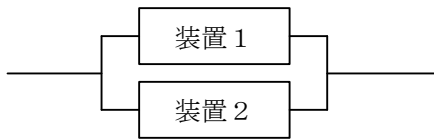


図 2-14 並列システムの例

	装置 1	装置 2
①	○稼働	○稼働
②	○稼働	×故障
③	×故障	○稼働
④	×故障	×故障

このとき、どちらか一方の装置が稼働していればシステムの機能が保たれる（システム全体として稼働する）とき、システム全体が稼働しているのは①～③の場合になる。したがって、システム全体の稼働率は、①～③の確率を加算して求められる。ただし、それでは計算が面倒になるので、全動作状況の中で④の場合だけシステムが稼働しないと考え、全動作状況の確率の合計（1）から④の確率を減算して、①～③の確率の合計を求める。以上から、並列システムの稼働率は、次の式で求められる。

$$\begin{aligned}\text{並列システムの稼働率} &= 1 - \text{装置 1 の非稼働率} \times \text{装置 2 の非稼働率} \\ &= 1 - (1 - \text{装置 1 の稼働率}) \times (1 - \text{装置 2 の稼働率})\end{aligned}$$

同様に、図 2-15 のように装置が 3 台以上並列に接続されている場合を考える。



図 2-15 3台の並列システムの例

	装置 1	装置 2	装置 3
①	○稼働	○稼働	○稼働
②	○稼働	○稼働	×故障
③	○稼働	×故障	○稼働
④	×故障	○稼働	○稼働
⑤	○稼働	×故障	×故障
⑥	×故障	○稼働	×故障
⑦	×故障	×故障	○稼働
⑧	×故障	×故障	×故障

この場合は、装置が何台以上稼働していればシステム全体が稼働しているかという条件に応じて、次のように考えていけばよい。

【1 台以上が稼働していれば、システム全体が稼働するとき】

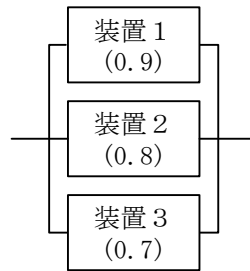
システム全体の稼働率 = 1 - ⑧の確率

【2 台以上が稼働していれば、システム全体が稼働するとき】

システム全体の稼働率 = ①の確率 + ②の確率 + ③の確率 + ④の確率

なお、並列システムでは装置を多く接続していけば、それだけ稼働率は高くなる。

例. 装置が 2 台以上稼働していればシステムが稼働する, 下図のシステムの稼働率を求めよ。なお, () 内の数値は各装置の稼働率である。



並列システムの稼働率 (2 台以上稼働していればシステムが稼働する場合)

= 3 台とも稼働している確率 + 2 台だけ稼働している確率

= $(0.9 \times 0.8 \times 0.7)$... ①の確率

+ $0.9 \times 0.8 \times (1 - 0.7)$... ②の確率

+ $0.9 \times (1 - 0.8) \times 0.7$... ③の確率

+ $(1 - 0.9) \times 0.8 \times 0.7$... ④の確率

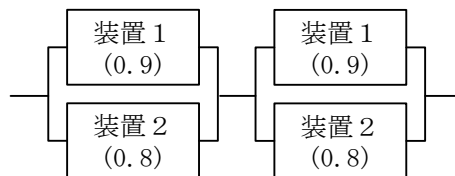
= $0.504 + 0.216 + 0.126 + 0.056$

= 0.902

(3) 直列／並列混合システムの稼働率

直列システムと並列システムが混在しているシステムの稼働率は, それぞれ部分的に稼働率を求めてから, システム全体の稼働率を計算する。

例. 下図のシステムの稼働率を求めよ。なお, () 内の数値は各装置の稼働率であり, 並列部分についてはどちらか一方が稼働していればよいものとする。



直列／並列混合システムの稼働率

= 第 1 並列部 (左側) の稼働率 $\times$ 第 2 並列部 (右側) の稼働率

= $\{1 - (1 - 0.9) \times (1 - 0.8)\} \times \{1 - (1 - 0.9) \times (1 - 0.8)\}$

= 0.98×0.98

= 0.9604

(4) システムの故障率

故障率とは、装置又はシステムに故障（障害）が発生する確率のことである。装置は一定時間稼働した後に故障が1回発生するため、装置単体の故障率はMTBFの逆数となる。

$$\text{故障率} = \frac{1}{\text{MTBF}}$$

また、複数の装置（部品）が組み合わされた場合、全体の故障率は各装置（部品）の故障率の総和となる。

$$\text{システムの故障率} = \Sigma \text{装置単体の故障率} \quad (\text{※} \Sigma \text{は総和を表す})$$

例. 故障率 10^{-8} の部品 100,000 個で構成された CPU の MTBF を求めよ。

① CPU の故障率

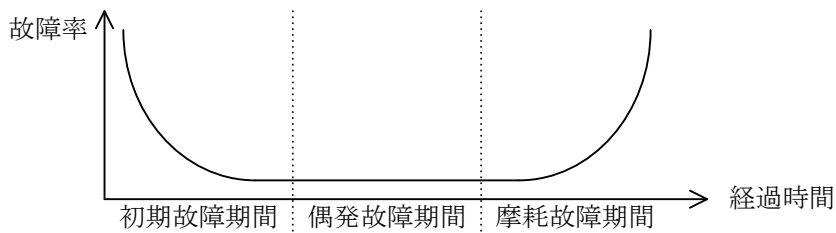
$$\begin{aligned} &= \text{部品 1 の故障率} + \text{部品 2 の故障率} + \cdots + \text{部品 100,000 の故障率} \\ &= 10^{-8} \times 100,000 \\ &= 10^{-3} \end{aligned}$$

② CPU の MTBF

$$\begin{aligned} &= 1 \div \text{CPU の故障率} \\ &= 1 \div 10^{-3} \\ &= 1,000 \end{aligned}$$

なお、ハードウェアの故障率の変化を、経過時間を横軸、故障率（あるいは故障の発生頻度）を縦軸にしたグラフで表すと、**バスタブ曲線**となることが多い。

〔バスタブ曲線〕



- ・ 初期故障期間：導入時の初期故障が発生する期間
- ・ 偶発故障期間：偶発的な操作ミスなどによる故障が発生する期間
- ・ 摩耗故障期間：耐用年数を過ぎた機械の劣化に伴う故障が発生する期間

3 - 3 経済性の評価

情報処理システムの経済性を評価する指標としては、**TCO** (Total Cost of Ownership ; 総所有コスト) がある。TCO は、システム導入時に発生する費用から、システム導入後に発生する費用までを含んだ総コストである。

(1) 初期コスト（イニシャルコスト）

初期コスト（イニシャルコスト）とは、システム導入時に発生する一時的なコスト（費用）である。代表的な初期コストには、次のようなものがある。

- ・ **ハードウェアの購入費用**

システムを構成するハードウェア（機器）を購入する費用である。

- ・ **ソフトウェアの購入費用**

システムで市販されているソフトウェア（アプリケーション）を利用するとき、そのソフトウェアを購入する費用である。

- ・ **ソフトウェアの開発費用**

システムで自社専用のソフトウェアを利用するとき、そのソフトウェアの開発にかかる費用である。自社で開発する場合には開発要員の人件費、他社に開発を依頼する場合には外注費、などの費用として発生する。

(2) 運用コスト（ランニングコスト）

運用コスト（ランニングコスト）とは、システム導入後の運用時に発生する定期的・継続的なコスト（費用）である。代表的な運用コストには、次のようなものがある。

- ・ **ハードウェアのレンタル／リース費用**

システムを構成するハードウェア（機器）を借用する場合の費用である。

- ・ **オペレータ費用**

システムを運用するオペレータに支払う人件費などの費用である。

- ・ **設備維持費用**

設備（コンピュータセンタ、ネットワーク環境など）を維持・管理するための保守費用などの費用である。

ハードウェアやソフトウェアの購入費用のように、対象と直接的な関係にあるコストを**直接コスト**という。一方、対象と間接的な関係にあるコストを**間接コスト**という。この分類は、絶対的ではなく、相対的なものである。例えば、オペレータの人件費は、対象システムの専任オペレータである場合は、直接コストに分類される。しかし、対象システムを含む社内システム全般のオペレータである場合は、間接コストに分類すべきである。

4 ヒューマンインタフェース

ヒューマンインタフェースとは、人間とシステム（コンピュータ）／サービスの接点のことである。特に、人間とコンピュータの接点であるユーザインタフェースは、対話型処理システムなどで重要視されている。人間が利用しやすいシステム／サービスを作成・提供するためには、ヒューマンインタフェースについて十分に理解しておく必要がある。

4-1 ヒューマンインタフェース技術

ヒューマンインタフェースでは、利用者にとっての“使い勝手の良さ”である**ユーザビリティ**と、利用者にとっての“アクセスのしやすさ”である**アクセシビリティ**を考慮する必要がある。

JIS Z 8521 では、ユーザビリティ（usability）の概念を、“特定のユーザが特定の利用状況において、システム、製品又はサービスを利用する際に、効果（effectiveness；ユーザが特定の目標を達成する際の正確性及び完全性）、効率 efficiency；達成された結果に関連して費やした資源）及び満足（satisfaction；システム、製品又はサービスの利用に起因するユーザのニーズ及び期待が満たされている程度に関するユーザの身体的、認知的及び感情的な受け止め方）を伴って特定の目標を達成する度合い”と定義している。

一方、JIS X 8341 では、アクセシビリティ（accessibility）として、“様々な能力をもつ最も幅広い層の人々に対する製品、サービス、環境又は施設（のインタラクティブシステム）のユーザビリティ”と定義している。

(1) インフォメーションアーキテクチャ（情報アーキテクチャ）

インフォメーションアーキテクチャ（情報アーキテクチャ）とは、情報を探しやすく整理して、わかりやすく伝えるための表現技術である。情報を整理する一般的な手順は、次のとおりである。

1) 情報の組織化

伝えたい情報を、五十音順やカテゴリ順などに分類・整理する。分類した情報のまとまりを**チャンク**、情報のまとまりに付けた名前を**ラベル**という。

2) 情報の構造化

組織化した情報を、階層型に分類・整理する。この処理を“タグ付け”ということもある。

(2) インタラクティブシステム

インタラクティブシステムとは、双方向（インタラクティブ）で情報のやり取りが行えるシステムのことである。

ただし、**JIS Z 8530**では、インタラクティブシステムを、“ユーザが遂行する仕事の能率を上げるように、ユーザからの情報入力を受け取り、ユーザに出力の伝達を行う、ハードウェア及びソフトウェアの組合せ”と規定している。

この規定を受けた **JIS Z 8520**では、人間工学的に望ましいインタラクティブシステムの設計に必要な七つの対話の原則について規定している。この七つの対話の原則は、ユーザインタフェースにおける使いにくさの問題に直面する事態を回避する指針となる。

【七つの対話の原則【JIS Z 8520】】

① 仕事への適合性	⑤ 可制御性
② 自己記述性	⑥ 誤りに対しての許容度
③ ユーザの期待への一致	⑦ 個人化への適合性
④ 学習への適合性	※①～⑦は優先度を示すものではない。

また、インタラクティブシステムにおけるユーザの情報入力では、キーボードやマウスなどのほかに、音声や映像を利用したインタフェースが利用される。

① 自然言語インタフェース

人間が普通に使用している言葉（自然言語）を利用して情報入力を行うインタフェースである。**音声認識**を利用した **VUI**（Voice User Interface）により、会話形式で処理を進めていくシステムなどが該当する。

② ノンバーバルインタフェース

言葉以外の、表情や動きなどを利用して情報入力を行うインタフェースである。**画像認識**や**動画認識**を利用して、目の動きや手の動きで情報を入力するシステムなどが該当する。なお、画像認識や動画認識では、画像の特徴を表すパターンを抽出する**特徴抽出**などが行われる。

ヒューマンインタフェースを考えるにあたっては、身体的適合性、頭脳の適合性、時間的適合性、環境的適合性、運用的適合性という「ヒューマン・マシン・インタフェースの5側面」を基準として、ユーザ操作の分析などを行うことも効果的である。

特に、**選択的知覚**（人間は関心のある情報や有利な情報だけを選択して受け入れる）を考慮して情報の表示方法を考えることや、作業時間の短縮につながる**学習機能**（同じ操作を素早く再現できる機能）を取り入れることなどは、ヒューマン・マシン・インタフェースに大きな効果をもたらすことが期待できる。

(3) GUI (Graphical User Interface)

GUI は、グラフィックス（絵）を利用することで、システムやサービスの使いやすさ、アクセスのしやすさを向上させる技術である。図 2-1（P. 106）の **ウィンドウ** や **アイコン** も GUI ツールの一つである。

- **ウィンドウ**

ディスプレイ上にウィンドウ（窓）をいくつも開き、それぞれのウィンドウを独立した一つのディスプレイのように使用できる。

- **アイコン**

資源やコマンド（アプリケーションへのショートカット）などを、わかりやすい図形やパターンで示したものである。画面上の矢印をマウスで動かしてアイコンを選択するだけで利用できるため、初心者にも操作しやすい。

- **ラジオボタン（ラジオボックス）**

選択項目（ボタン）の中から、一つだけ選択する。

- **チェックボックス**

選択項目（ボタン）の中から、いくつか（複数）を同時に選択する。

- **リストボックス**

選択項目（リスト）の中から、一つだけ選択する。

- **テキストボックス**

文字を入力する。

- **メニューバー**

選択可能な項目（メニュー）の一覧を、横に並べて表示する。

- **プルダウンメニュー**

メニューバーなどのメニューを選択したときに、さらに詳細なメニューを縦に並べて表示する。

- **ポップアップメニュー**

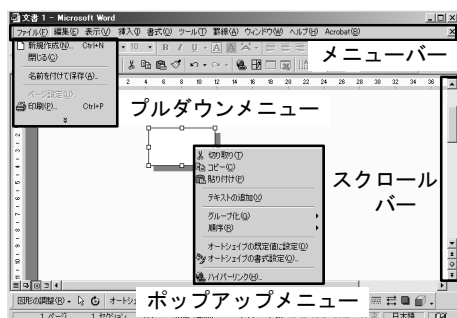
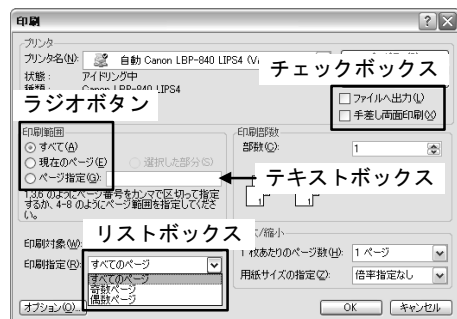
特定の位置で右クリックすることなどにより、メニューを浮かび上がらせる。

- **スクロールバー**

画面の表示領域を移動する。

- **プログレスバー**

処理の進行状況を、グラフのメモリ（棒の長さ）で表示する。



4-2 インタフェース設計

4-2-1 画面設計・帳票設計

(1) 画面設計

画面設計とは、画面に表示される内容やレイアウトなどを設計することである。画面のインタフェース設計では、「シュナイダーマンの8原則」が用いられることが多い。

【シュナイダーマンの8原則】

- ① 一貫性をもたせること
- ② 頻繁に使うユーザには近道を用意すること
- ③ 有益なフィードバックを提供すること
- ④ 段階的な達成感を与える対話を実現すること
- ⑤ エラーの処理を簡単にさせること
- ⑥ 逆操作を許すこと
- ⑦ 主体的な制御権を与えること
- ⑧ 短期記憶領域の負担を少なくすること

「シュナイダーマンの8原則」以外にも、画面設計を行うためには画面構成や画面に表示される情報の関係性に留意する必要がある。

【画面設計の留意点】

- ・画面上のタイトルやメッセージの表示位置を統一し、画面レイアウトの標準化を図る（ファンクションキーの利用方法なども標準化する）。・・・①
- ・エラーメッセージの表示場所は標準化するが、内容については標準化の対象としない（エラーの原因と対処方法がわかるように工夫する）。・・・⑤
- ・参照項目は、左から右へ、上から下に配置する。また、入力原票の形式にも配慮し、関連項目などは隣接して配置するようにする。
- ・画面上の入力項目は や [] で囲むなどして、入力場所であることを強調するように工夫する。
- ・画面上での入力には、最適な GUI ツールの利用を検討する。なお、メニューなどによる初心者向きの段階的選択以外にも、熟練者向けの**ショートカットキー**などを利用した直接選択も可能にする。・・・②
- ・処理中にはプログレスバーなどで進行状況を表示して、利用者に正常に稼働していることを知らせる。・・・④
- ・**Undo**（元に戻す）機能により、一つ前の状態に戻せるようにする。・・・⑥

また、入力画面の設計においては、**フルプルーフ**に配慮してデータの**入力チェック**の方法についても検討する。代表的な入力チェックの方式は、次のとおりである。

- ・ **ニューメリックチェック**（数字検査）
数値項目に数値以外のデータが入っていないかどうかを検査する。
- ・ **リミットチェック**（限度検査）
指定された限定値以上に達していないかどうかを検査する。
- ・ **フォーマットチェック**（書式検査）
データが指定された書式に従っているかどうかを検査する。
- ・ **重複チェック**
同じデータが重複していないかどうかを検査する。
- ・ **照合チェック**
データとマスタファイルなどを照合して正しいデータかどうかを検査する。
- ・ **バランスチェック**
別々に集計した合計が一致しているかどうかを検査する。
- ・ **論理チェック**（妥当性検査）
データが論理的に正しいかどうかを検査する。
- ・ **シーケンスチェック**
番号が順番に並んでいるか（欠番がないか）どうかを検査する。
- ・ **組合せチェック**
入力された複数の関連項目の組合せに矛盾がないかを検査する。
- ・ **チェックディジットチェック**
検査文字（**チェックキャラクタ**）を利用して入力データを検査する。

(2) 帳票設計

帳票設計は、システムから出力される帳票（報告書など）の項目やレイアウトなどを設計する。帳票設計の基本的な手順は、次のとおりである。

- 1) 総合的な検討
出力される帳票の出力目的、出力周期（時期）、出力タイミング、配付先、出力量（発生量）などを、総合的に検討する。
- 2) 出力方式、出力媒体の決定
総合的な検討をもとにして、最適な出力方式と出力媒体を決定する。出力媒体としては、紙以外にも CD への出力や画面表示などがある。
- 3) 出力項目、帳票レイアウトの作成
必要な出力項目を検討し、帳票上にわかりやすく配置する。このとき、文書の雛形（枠線や罫線など）をプリンタに登録しておき、データを重ね合わせて印字する**フォームオーバーレイ**の利用なども検討する。

4-2-2 コード設計

コード設計では、システム内で利用されるデータ（項目）のうち、コード化するデータを選定し、対象ごとにコード表を作成する。

〔コードに要求される四つの機能〕

- ・識別機能 : データを区別することができる。
- ・分類機能 : データ进行分类することができる。
- ・配列機能 : データの順序を決定することができる。
- ・チェック機能 : データが正しいかどうかをチェックすることができる。
(コードに検査用の冗長ビットを付けた場合)

コードは、原則として数字、英字を主体にして構成する。漢字などは、コードの意味をわかりやすくするには効果的であるが、維持・管理面からは適切でない。
代表的なコードの種類は、次のとおりである。

・ 順番コード

先頭項目から順番に割り当てるコードである。

例. 都道府県コード {01 : 北海道, 02 : 青森県, 03 : 岩手県, ...}

・ 区分コード (分類コード)

複数のブロックに分割し、各ブロック内で連番を割り当てるコードである。

例. 大学コード 0** 国立大学 {001 : 東京大学, 002 : 一橋大学, ...}
1** 公立大学 {101 : 都立大学, 102 : 横浜市大, ...}
2** 私立大学 {201 : 明治大学, 202 : 慶応大学, ...}

・ 桁別コード

各桁に、大分類、中分類、小分類...という意味をもたせるコードである。

例. 会員番号 1 桁目 : 種 別 (1 : 個人会員, 2 : 法人会員, ...)
2 桁目 : コース (1 : 全日, 2 : 昼間, 3 : 夜間, ...)
3 ~ 5 桁目 : 登録順序番号
※ “21006” — 全日コースの法人会員 (登録順は 6 番目)

・ 表意コード

コード化するデータの名称や略称を利用するコードである。

例. 商品コード JP20CTV : 日本製の 20 インチカラーテレビ
FR16MTV : フランス製の 16 インチ白黒テレビ

・ 合成コード

複数の種類のコードを組み合わせるコードである。

コード設計では、次のような点に注意する必要がある。

・ **コードの適用範囲**

対象システム以外でも利用される可能性がある場合、対象システムだけを考慮してコード設計をしてはいけない。コードは基本的に、共通して利用できるものが望ましい。既存コードや標準コードがある場合は、流用すべきである。

・ **コードの使用期間**

コードの使用期間によって、必要なコードの数が非常に多くなる場合もある。例えば、商品コードが順番コードで設計されており、現在の商品種類数が800点のとき、数字3桁で1,000種類の商品をコード化できるので、問題はないように思える。しかし、新商品が毎年50種類も開発される場合、4年後にはコードが不足することになる。

・ **コードの保守**

コードの保守方法についても、コード設計の段階で考慮する。特にコード表に関しては、データに矛盾が発生しないように、作成担当者や作成のタイミングなどを考慮して決定する。

・ **コードの重要性**

重要なデータのコードに関しては、オペレータの入力ミスなどを防ぐために、**チェックディジット**などを付ける必要がある。

【**チェックディジット**】

コードに付けられるチェックディジット（検査数字）を求める方式としては、**モジュラス 10**などが代表的である。

＜チェックディジットの求め方の一例＞

- ① コード“3612”の各桁に重み（コードの左から順に、5, 4, 3, 2）を乗算し、各乗算の結果を加算する。

$$3 \times 5 + 6 \times 4 + 1 \times 3 + 2 \times 2 = 46$$

- ② 加算した結果を10で除算して、余りをチェックディジットとする。
（余りを、さらに10から減算する方法もある。）

$$46 \div 10 = 4 \text{ 余り } 6 \rightarrow \text{コード“36126”}$$

※このコードを“36026”と間違えて入力すると、

$$(3 \times 5 + 6 \times 4 + 0 \times 3 + 2 \times 2 - 6) \div 10 = 3 \text{ 余り } 7$$

と、余りが0にならないのでコードに誤りがあることがわかる。

4-2-3 ヒューマンインタフェースの技法

(1) Web デザイン

Web デザインとは、インターネットの情報発信の仕組みである WWW (World Wide Web) で利用される Web ページ/Web サイトを設計 (デザイン) することである。

① Web ユーザビリティ

Web ページ/Web サイトの“使い勝手の良さ”である。Web ユーザビリティについては、ヤコブ・ニールセン博士のコラム“Alertbox”が参考にされることがある。この中では、Web ユーザビリティの質的な構成要素として、次の五つをあげている。

- ・学習容易性：どれくらい容易に利用できるようになるか？
- ・効率性：どれくらい迅速に利用できるようになるか？
- ・記憶性：どれくらい容易に操作を思い出せるようになるか？
- ・エラー：どれくらい容易にエラーから回復できるか？
- ・満足度：どれくらい楽しく利用できるか？

② Web アクセシビリティ

Web ページ/Web サイトの“アクセスのしやすさ”である。Web アクセシビリティについては、WWW の規格化を行っている **W3C** (WWW Consortium) が、次の **WCAG 2.0** (Web Content Accessibility Guidelines 2.0) を勧告している。

【WCAG 2.0 (抜粋)】

原則 1：知覚可能 (情報およびユーザインタフェースの構成要素は、ユーザが知覚できる方法でユーザに提示可能でなければならない)

- 1.1ー代替テキスト
- 1.2ー時間経過で変化するメディア
- 1.3ー適応可能
- 1.4ー識別可能

原則 2：操作可能 (ユーザインタフェースの構成要素およびナビゲーションは操作可能でなければならない)

- 2.1ーキーボード操作可能
- 2.2ー十分な時間
- 2.3ー発作の防止
- 2.4ーナビゲーション可能

原則 3：理解可能 (情報およびユーザインタフェースの操作は理解可能でなければならない)

- 3.1ー読みやすさ
- 3.2ー予測可能
- 3.3ー入力支援

原則 4：ロバスト性 (コンテンツは、支援技術を含む様々なユーザエージェントが確実に解釈できるように十分に堅牢でなければならない)

- 4.1ーロバスト性

③ スタイルシート

Web ページの文書構造や文書の装飾情報（**フレーム**、文字サイズや行間など）を定義したものである。Web ページは、HTML（HyperText Markup Language ; Web ページ用の記述言語）などで作成される。この HTML などと切り離して、Web ページのレイアウトを定義できるのが、**CSS**（Cascading Style Sheets）と呼ばれるスタイルシートである。スタイルシートを利用することで、Web ページに載せたい情報を所定の手順で与えるだけで、統一のとれた Web サイトを作成することができる。なお、CSS はブラウザ側で解釈されるが、同じコードでもブラウザによって動作が異なることがある。そのため、**クロスブラウザ**や**プログレッシブエンハンスメント**などが利用される。

- ・ **フレーム**

表示する Web ページを複数に分割する表現技法である。フレームごとに付ける飾り線や罫線などの装飾デザインを表すこともある。

- ・ **クロスブラウザ**

Web サイトや Web アプリケーションなどが正常に動作するように、ブラウザの仕様による動作の違いをなくす技術である。ブラウザごとに異なる処理をすることで、同じコードが違う動作をすることを回避させる。

- ・ **プログレッシブエンハンスメント**

どのような PC 環境でも基本的な情報を伝え、しかも高機能なブラウザを用いているユーザにはより豊かな体験を提供しようという、Web デザインの考え方である。“テキストコンテンツが確実に伝わるような論理構造”，“視覚的なデザインの制御”，“サイトの振舞いの制御”という3階層の構造によって、どのような閲覧環境（ブラウザ）でも情報が正確に伝わることを重視している。

④ ナビゲーション

Web サイトにおけるナビゲーションとは、サイト内にある目的の情報（Web ページ）まで素早く正確に到達できるようにする仕組みのことである。Web サイトで利用される代表的なナビゲーションには、次のような種類がある。

- ・ **サイトマップ**

Web ページの情報がわかるように付けたラベルを、階層的に表示することで、目的の情報をすばやく検索できるようにした仕組みである。

- ・ **サイト内検索**

検索するキーワードを入力すると、該当する Web ページや Web ページ内の位置などを表示する機能である。すべての文章を対象とする全文検索型、登録してあるキーワードを検索するインデックス型などがある。

- ・ **パンくずリスト**

トップページから、閲覧している Web ページまでの経路情報である。

(2) 人間中心設計

人間中心設計とは、システムを使いやすくすることに主眼をおいたインタラクティブシステム開発のアプローチである。単なるユーザビリティの向上ではなく、製品やサービスを利用したときのユーザ体験（**UX**：User Experience）の観点から、ユーザにとって望ましいシステムを設計する **UX デザイン**の発展である。**JIS Z 8530 (ISO 9241-210)**「人間工学—インタラクティブシステムの人間中心設計」として規格化されている。

【人間中心設計の原則】

- a) ユーザの積極的な参加、及びユーザ並びに仕事の要求の明解な理解
- b) ユーザと技術に対する適切な機能配分
- c) 設計による解決の繰返し
- d) 多様な職種に基づいた設計

(3) ユニバーサルデザイン

ユニバーサルデザインとは、年齢、文化、能力の違いや障がいの有無にかかわらず、すべての人々に、快適で利用しやすい環境やサービスを提供する設計のことである。ユニバーサルデザインでは、視力に障がいのある人のために文字サイズ・色を変えられる仕組みや、手の障がいでマウスを使えない人のために音声入力などの仕組みを用意する。

【ユニバーサルデザインの7原則】

- ① 公平性：どんな人でも公平に使えること
- ② 自由度：使う上で自由度が高いこと
- ③ 単純性：使い方が簡単で直感的にすぐ分かること
- ④ 明解さ：必要な情報がすぐに分かること
- ⑤ 安全性：うっかりミスや危険なことにつながらないこと
- ⑥ 低負担：無理な姿勢を取ることなく、少ない力で利用できること
- ⑦ 空間性：十分な大きさと空間を確保すること

また、W3Cの内部組織である **WAI** (Web Accessibility Initiative) が、特に障がいのある人にとって Web コンテンツをアクセシブル（アクセシビリティの高い状態）にするためのガイドラインとしてまとめた **WCAG 1.0** (1995 年) や **WCAG 2.0** (2008 年)、映像や音声などを利用する動的な Web コンテンツでのアクセシビリティに対する仕様書 **WAI-ARIA** (WAI-Accessible Rich Internet Applications) など指針とされる。

さらに、高齢者や障がい者に対する障壁（バリア）を排除して障壁のない（フリー）情報通信環境を提供する **情報バリアフリー**という考え方もある。情報バリアフリーについては、**JIS X 8341**（高齢者・障害者等配慮設計指針—情報通信における機器、ソフトウェア及びサービス—）で規定されている。

4-2-4 ユーザビリティ評価

ユーザビリティ評価とは、ヒューマンインタフェース設計によって作成（開発）されたシステムやWebサイトのユーザビリティ（使い勝手の良さ）を評価することである。

(1) 定性的評価

ユーザビリティ評価における定性的評価とは、システムやWebサイトのインタフェースなどを技術的に評価することである。主に、ユーザビリティの“有効さ”や“効率”を評価する目的で実施する。

【定性的評価の手法】

- ・ **ユーザビリティテスト**（ユーザテスト）

ユーザビリティエンジニアが、被験者（ユーザ）に課題（タスク）を与え、課題を実行する過程における、被験者の行動、発話などから、ユーザインタフェースの問題点を発見する手法である。なお、Webサイトを評価する場合などは、アクセスログ（Web ページアクセス情報の記録）を解析して、どのようにWeb ページを遷移したかを分析する**ログデータ分析法**（アクセスログ分析）などが併用されることもある。

- ・ **ヒューリスティック評価**

ユーザビリティエンジニア／デザイナーなど、技術・経験が豊富なメンバを招集し、既知の経験則（ヒューリスティック）を踏まえながらユーザインタフェースを評価し、問題点を明らかにする手法である。ユーザの意見が反映されない、**エキスパートレビュー**（専門家によるレビュー）の一つである。

(2) 定量的評価

ユーザビリティ評価における定量的評価とは、システムやWebサイトを他と比較して、相対的に評価する手法である。主に、“利用者の満足度”を評価する目的で実施する。

【定量的評価の手法】

- ・ **アンケート調査／インタビュー法**

システムやWebサイトを利用したユーザの意見を、不特定多数に対するアンケートや、対面式のインタビューで収集する方式である。他のシステムやWebサイトと比較した、利用者の不満などを収集しやすい。

5 マルチメディア

現代社会において、情報処理システムはさまざまな分野で利用されている。その中で、最近よく利用されているのがマルチメディアシステムである。マルチメディアシステムとは、動画・静止画・音声・テキストなどのマルチメディアデータをコンピュータで統合利用するシステムであり、さまざまな表現が可能になる。

5-1 マルチメディア技術

5-1-1 マルチメディア

マルチメディアとは、文字、画像、映像、音声などの情報をデジタル化（符号化）して情報メディアを統合した、**インタラクティブ性**（双方向性）のあるメディアである。

私たちが利用しているインターネット上のさまざまな情報は、Web サーバに登録されている（これを **Web コンテンツ** という）。ほとんどの Web コンテンツは、文字、画像、音声などを組み合わせたマルチメディアコンテンツである。これらの Web コンテンツを保存・管理し、Web サイトを構築するソフトウェアを **CMS**（Contents Management System；**コンテンツマネジメントシステム**）という。代表的な CMS として、ブラウザから Web ページの編集が簡単に行える **Wiki** などがある。インターネットの辞典といわれる Wikipedia は、Wiki の代表的な利用例である。

ほとんどの Web コンテンツにはリンク（関連ページへのアクセス情報）が埋め込まれており、リンクをたどることでさまざまな情報を検索することができる。文字情報のリンクによって関連付ける Web コンテンツを **ハイパテキスト**、画像などのリンクによって結びつける Web コンテンツ（マルチメディアコンテンツ）を **ハイパメディア** と呼ぶ。

マルチメディアコンテンツを作成するためのオーサリング環境では、主に **マルチメディアオーサリングツール** というソフトウェアを利用する。このソフトウェアでは、文書や画像／映像、音声などの素材を、マウス操作などによって簡単に統合できる（統合したファイル形式を **コンテナフォーマット** という）。このとき、Adobe Systems 社が開発した文書ファイル形式 **PDF**（Portable Document Format）が用いられることがある。PDF は、印刷イメージをそのまま PDF ファイルに変換することで、異なるアプリケーションでもほぼ同じように表示できる。PDF ファイルに変換すると、元のデータファイルよりもファイルサイズは小さくなるが、元のアプリケーションで編集できなくなる。ただし、PDF ファイルの特性は、データの変更や二次利用を防止できる利点にもなる。

なお、マルチメディアのデータファイルは、その性質上、文字情報だけのデータファイルよりもサイズが大きくなる。そのため、インターネットなどでは、映像や音楽のデータファイルをダウンロードしながら再生する **ストリーミング** という技術が利用される。

5-1-2 音声処理

音声処理とは、アナログの音声をデジタルデータとして扱う処理である。アナログ信号をデジタル信号に符号化する **A/D 変換** の技術には、**PCM** (Pulse Code Modulation; **パルス符号変調**) がある。PCM による符号化 (A/D 変換) の手順は、次のとおりである。

【PCM による符号化手順】

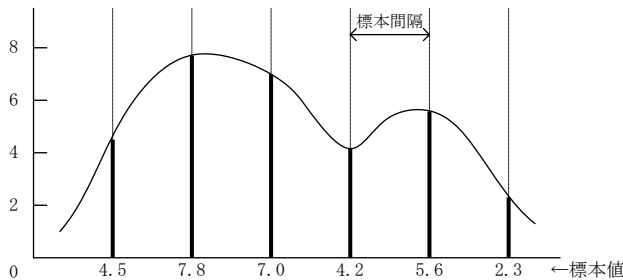
1) 標本化

符号化するアナログ信号を、一定間隔ごとにサンプリングする。このとき、標本間隔 (サンプリング間隔) を示す **標本化周波数** は、シャノンの標本化定理により決定する。

・シャノンの標本化定理

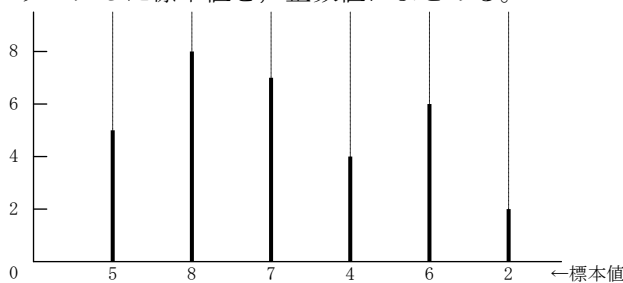
: 対象とするアナログ信号の最高周波数が f の場合、 $2f$ 以上の周波数でサンプリングすれば元のアナログ信号を復元できる。

※周波数・・・1 秒間の振動数。単位はヘルツ (Hz)



2) 量子化

サンプリングした標本値を、整数値にまとめる。



3) 符号化

量子化した整数値を、2進数で表現することで符号化する (符号として使用するビット数を **量子化ビット数** という)。

量子化した値	5	8	7	4	6	2
	↓	↓	↓	↓	↓	↓
符号化した値	0101	1000	0111	0100	0110	0010

例. 最高周波数 2,000Hz の音声データを量子化ビット数 8 ビットの PCM 方式で符号化するとき、1 秒間の音声データは全部で何ビットのデジタルデータに符号化されるか。ここで、標本化周波数は、シャノンの標本化定理によって求められる最低限必要な周波数とする。

① 標本化周波数

= アナログ信号（音声データ）の最高周波数 $\times 2$

= 2,000Hz $\times 2$

= 4,000Hz …… 1 秒間に 4,000 回サンプリングする

② 1 秒間の音声データを符号化したビット数

= 1 秒間のサンプリング回数 $\times$ 量子化ビット数

= 4,000 回/秒 $\times$ 8 ビット/回

= 32,000 ビット/秒

上記の例からわかるように、符号化後のデジタルデータのデータ量は、標本化周波数（標本間隔）と量子化ビット数により決定する。

また、デジタル信号（符号）をアナログ信号（音声データ）に復元する **D/A 変換**は、符号化手順を逆に行うことで実現できる。このとき、元のアナログ信号にできるだけ近い品質で復元できるようにするには、標本化周波数を高く（標本間隔を短く）して標本値の数を増やすか、量子化ビット数を大きくして標本値を詳細に段階化する。

代表的な音声ファイルの形式（規格）は、次のとおりである。

・ **MIDI** (Musical Instruments Digital Interface)

電子楽器や PC などを接続し、音楽データをやり取りするファイル形式である。音そのものではなく、音を再現するための演奏情報を取り扱う。

・ **MP3**

ISO により規格化された、MPEG（動画圧縮・伸張形式）の音声技術を利用した高品質の音声圧縮・伸張形式（ファイル形式）である。インターネット上での音楽配信やポータブルプレーヤでも利用されている。

・ **WAV** (RIFF Waveform Audio Format)

主に Windows 系で利用される音声データ形式である。コンテナ形式で ADPCM や WMA など、さまざまな形式のデータを格納することができる。

・ **ADPCM** (Adaptive Differential Pulse Code Modulation)

PCM の技術を応用した音声信号圧縮方式である。

・ **WMA** (Windows Media Audio)

元々はマイクロソフト社の開発した音楽配信などのストリーミング技術である。最近では、音声圧縮技術として利用されている。

5-1-3 静止画処理

静止画処理とは、動きのない画像を扱う処理である。画像表現の基本的な仕組みには、色の表現方法と画像品質がある。

- ・色の表現方法

光の3原色（RGB）と**色の3原色（CMY）**を利用する。一般的には、ディスプレイの色表現にRGB、プリンタの色表現にCMYを利用する。ただし、色の3原色では黒色（Key）を加えたCMYKの4色を利用する場合が多い。

- ・画像品質

解像度と**階調**で表現する。解像度は、画像を構成する**画素（ピクセル、ドット）**の密集度を、1インチ当たりの画素（ドット）数（**dpi** : dots per inch）で表す。階調は、画素ごとに表現できる色の段階を表す。

代表的な静止画ファイルの形式（規格）は、次のとおりである。

- ・**BMP** (Bit MaP ; **ビットマップ**)

点（ドット）の集合として、文字や画像を表現する形式である。一般に、他の形式に比べて情報量が多くなるという欠点がある。

- ・**TIFF** (Tagged Image File Format)

タグ（識別子）を付けることで、さまざまな形式のビットマップ画像を扱える形式である。

- ・**JPEG** (Joint Photographic Experts Group)

ISOの一機関であるJPEGが定めた、フルカラー対応の国際規格である。圧縮量を加減して画質とデータ量を調節できる、静止画圧縮・伸張形式である。非常に高い圧縮率をもつため、他の静止画ファイル形式よりも利用頻度は高い。JPEGの圧縮形式には、元のデータが完全に復元される**可逆圧縮**と、元のデータが完全には復元されない**非可逆圧縮**があるが、一般的には非可逆圧縮として使用される。

- ・**GIF** (Graphics Interchange Format)

256色カラー対応の静止画圧縮・伸張形式である。可逆圧縮であるが、データ量が比較的小さく、WWW上での画像処理にも使われる。

- ・**PNG** (Portable Network Graphics)

GIFを拡張した、フルカラー対応の静止画圧縮・伸張形式である。可逆圧縮であり、データ量はJPEGよりも大きくなる。

- ・**Exif** (Exchangeable image file format)

デジタルカメラで撮影した画像データに、撮影時などの情報（メタデータ）を追加して保存できる形式である。

- ・**HEIF** (High Efficiency Image File Format)

JPEGよりも圧縮率の高い画像フォーマット（コンテナ形式）である。

5-1-4 動画処理

動画処理とは、動きのある画像を扱う処理である。コンピュータの動画処理では、1枚の画像を**フレーム**（コマ）として、次々に表示する。このとき、1秒間に表示するフレーム数を**フレームレート**といい、**fps**（frame per second）という単位で表される。

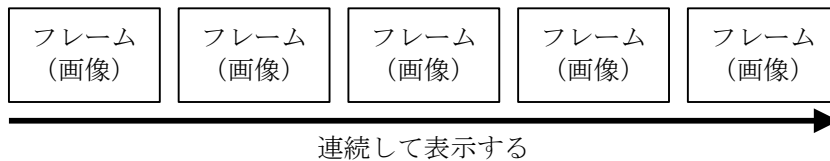


図 2-16 動画処理のイメージ

代表的な動画ファイルの形式（規格）は、次のとおりである。

- **MPEG** (Moving Picture Experts Group)

ISO の一機関である MPEG が定めた、動画データの国際規格である。圧縮対象の画像の品質などに応じて、複数の規格（MPEG-1, MPEG-2, MPEG-4, MPEG-7 など）が定められた動画圧縮・伸張形式である。インターネットなどで用いられる動画データ圧縮符号化方式の **H. 264/MPEG-4 AVC** は MPEG-4 の一種（改良版）である。

- **MP4**

主に、MPEG-4 の動画圧縮データを格納するファイル形式である。

- **MOV**

主に、QuickTime で用いられる動画圧縮データのファイル形式である。JPEG 画像を連続して再生する **モーション JPEG** が採用されている。

- **VRML** (Virtual Reality Modeling Language)

3次元グラフィックデータ（映像）を表示するためのファイル形式である。

- **AVI** (Audio Video Interleaving)

主に、Windows 系で利用される **WMV** (Windows Media Video) の動画圧縮データを格納するファイル形式で、マルチメディアファイル形式である **RIFF** (Resource Interchange File Format) を利用する。

【CSV 形式 (CSV : Comma Separated Value)】

CSV 形式は、表計算ソフトやデータベースソフトで利用されるファイル形式である。個々のフィールドのデータをカンマで区切り、各レコードは改行コードで区切られる形式である。

5-1-5 情報の圧縮・伸張

マルチメディアのデータファイルは、その性質上、文字情報だけのデータファイルよりもサイズ（データ量）が大きくなる。このファイルをサイズの大きいまま利用すると、データを記憶するための記録領域も多く必要となり、ネットワークを通じてやり取りするにも負荷が大きくなる。そこで、データ保存の効率化とネットワーク負荷を軽減するために、一定のルール（手順）に従って情報のサイズ（データ量）を小さくする。情報のサイズを小さくする処理を**圧縮**、圧縮された情報を復元する処理を**伸張**という。

圧縮の考え方（方式）には、大きく分けて次の2種類がある。

- **可逆圧縮**

圧縮したデータから、完全に元のデータに戻すことができる圧縮である。

- **非可逆圧縮**

圧縮したデータから、完全には元のデータに戻せない圧縮である。画像などの場合、復元が多少不完全でも人間には違いがほとんどわからないため、大きな問題となることはない。可逆圧縮に比べて、データ量の圧縮効果を示す**圧縮率**を高く（データ量を少なく）することができる。

代表的な情報圧縮・伸張の形式（規格）は、次のとおりである。

- **JPEG** (Joint Photographic Experts Group)

フルカラー対応の静止画圧縮・伸張形式の国際規格である。

- **MPEG** (Moving Picture Experts Group)

動画圧縮・伸張形式の国際規格である。

- **HEVC** (High Efficiency Video Coding)

H. 264 の約 2 倍の圧縮率を誇る後継規格（動画コーデック）であり、国際規格上は **H. 265** と表記される（又は H. 265/HEVC と併記される）。

- **ZIP**

WinZip などのソフトウェア製品のほか、数多くのフリーソフトウェアが対応しているファイル圧縮形式である。ファイル単体の圧縮だけではなく、複数のファイルを一つにまとめて圧縮する**アーカイブ**にも利用される。国際的には、ファイル圧縮形式のデファクトスタンダード（業界標準）になっている。

- **GZIP** (GNU ZIP)

主に、UNIX 系で利用されるファイル圧縮形式である。名称として ZIP が付いているが、ZIP との互換性はなく、アーカイブ機能もない。

- **MR** (Modified Read) / **MMR** (Modified Modified Read)

主に、ファクシミリで利用される情報圧縮・伸張形式（符号化方式）である。

5 - 2 マルチメディア応用

(1) CG (Computer Graphics ; コンピュータグラフィックス)

CG (コンピュータグラフィックス) とは、コンピュータを利用して画像 (グラフィックス) を描いたり、編集したりする技術である。

【CG 用のソフトウェア】

・ペイント系ソフトウェア

点 (ドット) 単位で画像を描くグラフィックスソフトウェアである。作成された画像をラスタグラフィックスという。マウスなどで辿った道筋上に、点 (ドット) で線を引いて画像を描くので、初心者でも容易に利用できる。

・ドロー系ソフトウェア

線の方向や長さ (ベクトルデータ) を計算して画像を描くグラフィックスソフトウェアである。作成された画像をベクタグラフィックスという。描いた画像を拡大しても、線のギザギザ (ジャギー) が目立たない。

【CG の技術】

・アンチエイリアシング

斜線や曲線に生じるギザギザを目立たなくする技術

・テクスチャマッピング

モデリングされた物体の表面に柄や模様を貼り付けて、質感を出す技術

・ブレンディング

透明度の情報を重ね合わせて、半透明画像を表現する技術

・レイトレーシング

視点に届いた光線を逆にたどって描画 (レンダリング) する技術

・クリッピング

画像全体から、表示する部分だけを取り出す技術

・シェーディング

立体感を出すため、物体の表面に影をつける技術 (ピクセル単位で処理する方式を **ピクセルシェーダ** ともいう)

・モーフィング

節点などを変化させて、ある形状から別の形状に滑らかに変化させる技術

・レンダリング

指定された視点から見える部分だけを描く **陰線消去** や **陰面消去** などにより、物体のデータを表示できるように映像化 (2次元化・3次元化) する技術

・ポリゴン

3次元CGで、閉じた立体や曲面を近似するために用いられる多角形

・キーフレーム法

基準となるフレーム間を補間して動画を作成する技術

(2) 3次元映像 (3D)

3次元映像 (3D) とは、奥行きや立体感のある映像（及び映像の作成技術）である。3次元コンピュータグラフィックス (3DCG) や**モーションキャプチャ**（人体の各関節にセンサを取り付け、動作時にセンサが感知した情報（座標や加速度）をコンピュータに取り込んでデジタルデータに変換する技術）などで作成される。また、**バーチャルサ라운드**（5.1ch 音声を二つのスピーカや普通のヘッドホンで仮想的に再現する技術）と組み合わせて、臨場感あふれるコンテンツを作成することもできる。

(3) VR (Virtual Reality)／AR (Augmented Reality)

VR (バーチャルリアリティ)；仮想現実) とは、CG などを用いて生成した仮想の世界を、現実存在する世界のように体験できることである。一方、**AR (拡張現実感)** とは、現実世界と VR を組み合わせることで、現実世界を拡張することである。AR では、現実存在する物事に対して、VR によって情報を付加したり、強調したりする。なお、過去に録画された映像を視聴することによって、その時代・その場所にいたかのような感覚が得られる **SR (Substitutional Reality)**；代替現実) という技術もある。

(4) その他の応用例

動画・静止画・音声・テキストなどのマルチメディアデータを、コンピュータで統合して利用する**マルチメディアシステム**には、さまざまな応用例がある。

- ・ **インターネット放送**
インターネットを媒体とする放送である。
- ・ **ノンリニア画像編集システム**
コンピュータ上で、画像をデジタルデータとして編集する方式である。
- ・ **CAD (Computer Aided Design)**
CG などを利用した、製品の設計を支援するコンピュータシステムである。
- ・ **シミュレータ**
現実にかかる複雑な科学現象などを、模擬的に実験するためのプログラムや仕組みである。台風の進路予測や風洞実験などをコンピュータで再現する。
- ・ **テレビゲーム**
コンピュータで遊ぶことができるゲームである。
- ・ **ビデオ・オンデマンド**
利用者の要求に応じて(**オンデマンド**)、ビデオ映像を配信するサービスである。
- ・ **バーチャルモール (サイバーモール)**
インターネット上に構築される仮想的な商店街である。商店街は存在しないが、実際に商品を購入したりすることはできる。

第2部 演習問題

問1

3層クライアントサーバシステム構成で実現したWebシステムの特徴として、適切なものはどれか。

- ア HTMLで記述されたプログラムをサーバ側で動作させ、クライアントソフトはその結果を画面に表示する。
- イ 業務処理を変更するたびに、Webシステムを動作させるための業務処理用アプリケーションをクライアント端末に送信し、インストールする必要がある。
- ウ 業務処理はサーバ側で実行し、クライアントソフトはHTMLの記述に従って、受け取った結果を画面に表示する。
- エ クライアント端末には、サーバ側からのHTTP要求を待ち受けるサービスを常駐させておく必要がある。

問2

一方のコンピュータが正常に機能しているときに、他方のコンピュータが待機状態にあるシステムはどれか。

- ア デュアルシステム
- イ デュプレックスシステム
- ウ マルチプロセッサシステム
- エ ロードシェアリングシステム

問3

フォールトトレラントシステムの構成要素に該当するものはどれか。

- ア RAID0
- イ 磁気ディスクの二重化
- ウ スケジュールバックアップ
- エ データ暗号化

問4

システムが単位時間内にジョブを処理する能力の評価尺度はどれか。

- ア MIPS 値
- イ スループット
- ウ ターンアラウンドタイム
- エ レスポンスタイム

問5

クロック時間が3ナノ秒の処理装置で、命令の実行に必要なクロック数とその命令の出現比率が表に示す値である場合、この処理装置の性能は平均約何 MIPS か。

命令の種別	命令実行に必要なクロック数	出現比率
レジスタ間演算	4	40%
メモリ・レジスタ間演算	8	50%
無条件分岐	10	10%

ア 5

イ 30

ウ 50

エ 100

問6

システムを安全かつ安定的に運用するための指標として、RASIS がある。稼働率は、RASIS のどれに含まれるか。

ア Availability

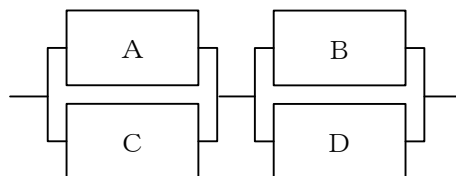
イ Integrity

ウ Reliability

エ Security

問7

図で示されるシステム構成全体の稼働率として、最も近いものはどれか。ここでA, B, C, Dは装置を示し、その稼働率は、A, Cは0.9, B, Dは0.8とする。



ア 0.72

イ 0.92

ウ 0.93

エ 0.95

問8

ヒューマンインタフェース設計において、操作の一貫性向上を目的とするものはどれか。

- ア Undo（元に戻す）機能によって、一つ前の操作状態に戻せるようにする。
- イ ショートカットキーによって操作できるようにする。
- ウ どの画面においても操作ボタンの表示位置や形を同じにする。
- エ 利用者の操作に対応した処理の進行状況を表示する。

問9

顧客コードにチェックディジット（検査数字）を付加する目的として、適切なものはどれか。

- ア 顧客コードの入力誤りを発見する。
- イ 顧客名簿を作るときに、獲得した順に顧客を配列する。
- ウ 顧客を地区別などのグループに分類できるようにする。
- エ 特定の顧客を類推できるようにする。

問10

ブラウザを用いた社員情報サービスを始めるに当たり、社員のフルカラーの顔写真を掲載することを検討している。社内ネットワークの負荷を小さくするために最も適切な画像圧縮形式はどれか。

- ア GIF
- イ JPEG
- ウ MIDI
- エ MPEG

問11

バーチャルリアリティの説明として、最も適切なものはどれか。

- ア CGなどの技術を用いることによって、コンピュータ内に作る世界を、実際の世界であるかのように表現することである。
- イ GUIの改善を目的として、画像を上から順次表示するのではなく、モザイク状の粗い画像をまず表示して、徐々に鮮明に表示することである。
- ウ 自動車や飛行機的设计に使われている風洞実験などを、コンピュータを使用して模擬実験し、想定した結果が得られるかどうかを試験することである。
- エ 製品の設計を、CG技術などを利用してコンピュータで支援することである。

第3部

ソフトウェア

1 ソフトウェアの分類

ソフトウェアとは、コンピュータを効率的に利用できるように、人間とコンピュータの間を取りもつもので、ハードウェアに対して作られた言葉である。広い意味のソフトウェアには、コンピュータを利用するための文書（マニュアル）なども含まれるが、一般的にはプログラム（命令の集まり）のことを表す。

1-1 ソフトウェアの体系による分類

ソフトウェアを体系的に分類すると、**システムソフトウェア**と**応用ソフトウェア**（**アプリケーションソフトウェア**又はアプリケーション）に大別される。システムソフトウェアは、コンピュータを構成する機器（ハードウェア）の機能を効果的に利用するためのプログラム群であり、応用ソフトウェアは利用者の目的に応じた機能を提供するプログラム群である。つまり、システムソフトウェアはハードウェア寄りのソフトウェアで、応用ソフトウェアは人間寄りのソフトウェアといえる。

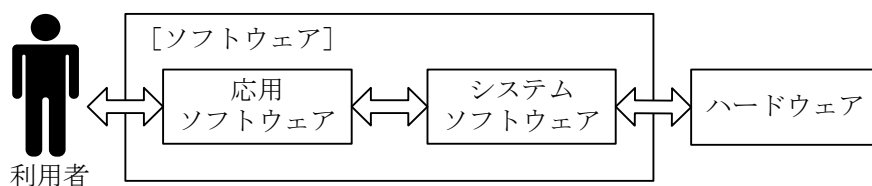


図 3-1 ソフトウェアの働き

また、ハードウェア資源などを管理してコンピュータを制御する基本ソフトウェアを、**OS** (Operating System ; **オペレーティングシステム**) と呼ぶ。この“コンピュータを制御する”という意味の解釈によって、OS は“広義の OS”と“狭義の OS”に分類される。

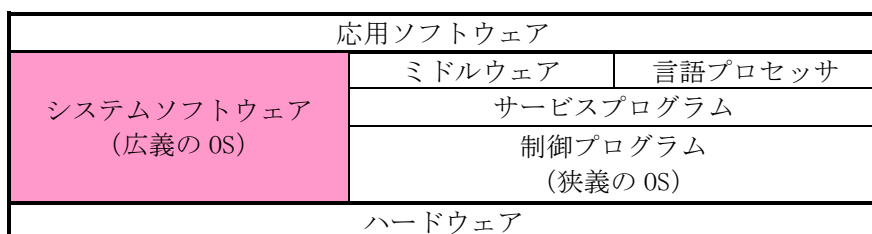


図 3-2 ソフトウェアの体系図

なお、この体系による分類は、今日では非常に曖昧なものとなっている。あくまでも、図 3-2 はソフトウェアの体系の一例（イメージ）と理解してもらいたい。

(1) 制御プログラム

制御プログラムは、ハードウェアなどの資源を効率的に管理／運用し、効率の良い動作環境を提供するソフトウェアである（狭義の OS とも呼ばれる）。制御プログラムの中核となるプログラムを**カーネル**という。通常のソフトウェアは、CPU の使用に制限がある**ユーザモード**で動作する。しかし、**割込み**（応用ソフトウェアからの処理要求など）によってカーネルが主体となって動作する場合は、CPU の使用に制限がない**カーネルモード**（**スーパーバイザモード**）に移行する。カーネルは、その機能により、次の二つに分類される。

- ・ **マイクロカーネル**

メモリ管理やプロセス管理など、必要最小限の機能だけ実装したカーネル

- ・ **モノリシックカーネル**

入出力管理やファイル管理など、多くの機能を実装したカーネル

コンピュータに電源を入れると、初期プログラム（**IPL** : Initial Program Loader）が実行されて、制御プログラム（OS）が起動する。この一連の処理を**ブート**といい、ブートに使用されるプログラムを**ブートストラップ**（ブートローダ）と呼ぶ。このブートストラップはコンピュータの電源が切れているときでも保存しておく必要があるため、初期のコンピュータでは ROM に記録されていた。しかし、現在はフラッシュメモリに記録しておく**フラッシュブートローダ**を利用していることが多い。また、複数の候補から起動する制御プログラム（OS）を選択できる**マルチブート**や、ネットワーク経由でブートを行える**ネットワークブート**なども利用されている。

(2) サービスプログラム（ユーティリティプログラム）

サービスプログラム（ユーティリティプログラム）は、コンピュータを利用するうえでの、利用者を支援するための定型的な機能を提供するソフトウェアである。代表的なものとして、複数のファイルを一つにまとめたり、元に戻したりする**アーカイブ**機能を提供するデータ管理ユーティリティ（**アーカイバ**）や、磁気ディスクを有効利用する機能を提供するディスク管理ユーティリティ（Windows のディスクデフラグ）などがある。

(3) 言語プロセッサ

言語プロセッサは、人間にわかりやすい言葉（**プログラム言語**）で作成されたプログラムを、コンピュータにわかる言葉（**機械語**）に翻訳するために利用されるソフトウェアである。代表的なものとして、元のプログラム（**ソースプログラム**）を機械語のプログラム（**オブジェクトプログラム**）に一括翻訳する**コンパイラ**などがある。なお、プログラムを実行するには、サービスプログラムの一つである**リンカージェディタ**などを利用して、実行プログラム（**ロードモジュール**）を組み立てる必要がある。

(4) ミドルウェア

ミドルウェアは、基本ソフトウェア（OS）と応用ソフトウェアの中間に位置して、多様な利用目的に共通する基本機能を提供するソフトウェアである。統一的なインタフェースで基本機能を提供することで、OS やハードウェアに依存しない応用ソフトウェアの開発を容易にする。ミドルウェアには、OS の一部として組み込まれているものと利用者が購入して追加するものがあるため、厳密に区別することは難しい。一般的に、ミドルウェアに分類されるソフトウェアには、次のような種類がある。

- ・ **応用プログラム間連携ソフトウェア**
応用プログラム間でのデータ互換などを可能にするソフトウェア
- ・ **データベース管理システム（DBMS：DataBase Management System）**
データを統合的に扱うデータベースを管理するソフトウェア
- ・ **通信管理システム**
コンピュータやサーバ間で行われる通信を管理するソフトウェア
- ・ **ソフトウェア開発支援ツール**
プログラム言語などによるソフトウェア開発を支援するソフトウェア
- ・ **運用管理ツール**
コンピュータ（サーバ）の運用を支援／管理するソフトウェア
- ・ **TPモニタ（TP：Transaction Processing；トランザクション処理）**
トランザクション処理を管理するソフトウェア

また、次の①～⑤も、広い意味ではミドルウェアに分類される。

① シェル

ユーザの操作を、コマンド解釈プログラム（**コマンドインタプリタ**）で解釈して、OS に伝える（指示する）ソフトウェアである。OS の機能に分類されることもある。

- ・ **リダイレクト機能**
標準入力（キーボード）／標準出力（ディスプレイ）を、それぞれファイルに変更できるシェルの機能である。標準出力をファイルに変更する場合は、上書きと追記のどちらかを指定することができる。

② API（Application Program Interface）

応用ソフトウェアから、OS の用意する各種機能（コマンド、関数など）を利用するための仕組み（インタフェース）である。同じ OS と API を利用することで、プログラムの互換性を高め、移植時の工数を削減することができる。

- ・ **Web API**
Web コンテンツの制作に有用な機能を利用できる仕組みであり、一般的には Web サイト運営者などがインターネット（Web）上で提供する。

③ ライブラリ

汎用性の高い特定の機能をもつプログラムを、再利用可能な形式でひとまとめにしたファイルである（ファイルを管理・提供する仕組みを指すこともある）。基本的には、ライブラリは別のプログラムに組み込んで使用するものであり、単独で実行することはできない。

- ・ **ソースライブラリ**

ソースコード（ソースプログラム）形式のライブラリである。

- ・ **オブジェクトライブラリ**

オブジェクトコード（オブジェクトプログラム）形式のライブラリである。なお、オブジェクト指向で使用されるクラス（データと手続を一体化したもの）を集めた**クラスライブラリ**を指す場合もある。

- ・ **ロードライブラリ**

ロードモジュールを生成するときに、組み込まれるライブラリである。一般的には、リンカージェディタを利用して組み込まれるものを指す。

- ・ **動的リンクライブラリ（DLL：Dynamic Link Library）**

実行時に、OSによって組み込まれるライブラリである。ロードライブラリの一種に分類されることもある。

④ コンポーネントウェア

ある特定の機能をもっていて、プログラム開発時に部品（コンポーネント）として使用されるソフトウェアのことである。コンポーネントウェアを組み合わせるアプリケーションを開発することを、**コンポーネントベースプログラミング**という。

- ・ **JavaBeans**

プログラム言語の一つである Java で開発されたプログラムを、コンポーネントとして取り扱うための規約である。

- ・ **ActiveX**

マイクロソフト社が提供する、インターネット関連のコンポーネント、およびコンポーネント技術の総称である。

- ・ **CORBA** (COmmon Request Broker Architecture)

オブジェクト同士のメッセージ交換を可能にする標準仕様であり、CCM (CORBA Component Model) というコンポーネントに関する仕様がある。

⑤ 開発フレームワーク

アプリケーションを開発する際に、頻繁に必要となる汎用的な機能をまとめて提供し、アプリケーションの土台として機能するソフトウェアである。アプリケーション開発を標準化・効率化するための全体的な枠組みともいえる。

(5) 応用ソフトウェア（アプリケーションソフトウェア）

応用ソフトウェア（アプリケーションソフトウェア）は、利用者の目的に応じてさまざまな機能を提供するソフトウェアである。ハードウェア寄りのシステムソフトウェア以外の多くのソフトウェアが応用ソフトウェアに属することになる。

応用ソフトウェアは、独自の問題解決のために開発されるオーダーメイドの**個別応用ソフトウェア**と、共通業務（給与計算や帳簿管理など）のために開発される定型的な**共通応用ソフトウェア**に分類される。共通応用ソフトウェアの多くは、利用者の環境や要求などに合わせて一部を変更して保存できる**カスタマイズ**機能を備えている。

① ワードプロソフト（ワードプロセッサソフトウェア）

文書を作成するときに使用されるソフトウェアで、文字の編集や図／グラフの挿入もできる。ワードプロソフトなどで使用されるフォント（字体）には、文字幅が等しい**等幅フォント**と文字幅が異なる**プロポーションアルフォント**、文字をドットの組合せで表現する**ビットマップフォント**と文字の輪郭線に関数曲線の情報で表現する**アウトラインフォント**などがある。なお、ビットマップフォントの文字を拡大するとギザギザ（ジャギー）が目立つため、拡大表示にはアウトラインフォントが適している。

- ・タブ機能 : 書式設定機能の一つで、印刷や表示の書式をそろえるために、カーソルを同じ行の指定位置まで移動させる機能
- ・マクロ機能 : あらかじめ手順を定義しておき、必要なときに呼び出して使うことができる機能
- ・差込み印刷機能 : 文書の一部に他のファイルのデータを挿入し、その部分だけを変えながら何枚も文書を印刷する機能
- ・禁則処理機能 : 句読点などを行頭にしない行頭禁則処理，“¥”や“(”を行末にしない行末禁則処理などの機能

② 表計算ソフト

スプレッドシートプログラムとも呼ばれ、ワークシート（スプレッドシート）上の縦横に区切った長方形のマス（セル）内にデータや計算式を入力することにより、表の作成やセル間の計算処理を行うソフトウェアである。

・プラグインソフトウェア（エクステンション、アドイン）

単独では動作せず、表計算ソフトなどのアプリケーションに組み込むことによって機能を追加するソフトウェアである。必要に応じて組み込んだり、取り外したりできる。代表的なプラグインソフトウェアには、マルチメディアデータを再生するFlash Playerなどがある。多くのプラグインソフトウェアは、インターネットなどから無料で入手できる。

③ データベースソフト

データベースを作成・利用・管理するためのソフトウェアである。専用言語を利用して、簡単なデータ処理のプログラムを作成することもできる。

④ DTP ソフト (DTP : DeskTop Publishing) / 電子編集システム

フォント調整やレイアウトなどの編集機能が充実している、商業印刷物作成のためのソフトウェアである。レイアウト作業や高精度プリンタを制御するために、**ページ記述言語** (PDL : Page Description Language) 機能を組み込んでいる (代表的なページ記述言語に **PostScript** や **ESC/Page** がある)。画面に表示されたものがそのまま印刷される **WYSIWYG** (What You See Is What You Get) 機能を特色としていたが、現在では DTP ソフト以外の応用ソフトウェアでも一般的に実現されている。

⑤ グラフィックスソフト

PC 上で図形や画像の作成・処理を行うためのソフトウェアである。**ペイント系ソフトウェア**や**ドロー系ソフトウェア**がある。現在は、CG ソフトや 3DCG ソフトという名称のほうが一般的である。

⑥ プレゼンテーションソフト

プレゼンテーション用の資料を作成したり、プレゼンテーションそのものに使用されるソフトウェアである。動画などのアニメーション機能や効果音などのサウンド機能が付いている場合がある。なお、プレゼンテーションに使用する**プロジェクタ**は、光源の明るさを表す単位である**ルーメン** (lm) の高いものを選んだほうがよい。

⑦ グループウェア

組織における共同作業をコンピュータで支援するソフトウェアである。代表的な機能として、電子メール機能、電子会議機能、スケジュール管理機能、**ワークフロー管理機能** (業務の処理手順や書類・情報の流れを規定／管理する機能) などがある。

(6) OS（オペレーティングシステム）

代表的な OS（オペレーティングシステム）としては、次のような種類がある。現在のほとんどの OS は、複数の利用者に同時（又は切替え時）に対応可能なマルチユーザ機能と、複数の処理に同時（又は切替え時）に対応可能なマルチタスク機能を備えている。

・ Windows

マイクロソフト社が開発した OS で、世界中の PC に搭載されている PC 用 OS である。シリーズとして、最新版が続々と発売されている。

- ・ 多彩なアイコンの利用などによる充実した GUI 機能
- ・ ウィンドウシステムを採用したマルチタスク機能
- ・ 接続するだけで周辺機器を使用できるプラグアンドプレイ機能
- ・ インターネットなどに対応するための充実したネットワーク機能

・ UNIX

AT&T ベル研究所でミニコン用に開発された OS であるが、現在では PC 用 OS として普及している。仕様が広く公開されているため、多くのメーカーが自社製品に取り入れやすく、さまざまな改良版も普及している。

- ・ カーネルとシェルによる構成
- ・ 分散処理を容易に実現するネットワーク機能
- ・ 入出力装置をファイルと同等に扱うことが可能
- ・ あるコマンドの標準出力を別コマンドの標準入力につなげるパイプ機能

・ Linux

UNIX の考え方を改良した PC 用 OS で、オープン OS（オープンソースソフトウェアの OS）の代表例である。

- ・ OS の核となるカーネル部をフリーで公開・配布
- ・ OS 占有モード（シングルユーザモード）によるシステム管理
- ・ { } 中の文字列を一定の法則で展開するブレース展開機能

・ macOS (OS X)

アップル社が自社 PC (Macintosh) 用に開発した PC 用 OS である。カーネルに UNIX の技術を取り入れたものも発表されている。

- ・ GUI による洗練された操作性
- ・ 充実したグラフィックスソフト／DTP ソフト

・ iOS, Android

スマートフォンやタブレット端末など、主にモバイル端末で利用されている OS である。iOS は macOS 系、Android は Linux 系をベースにしている。

・ リアルタイム OS

イベントなどに対応した処理を、定められた時間内に完了させる必要があるハードリアルタイムシステムなどに用いられる OS である。高い応答性能が要求される組込みシステムなどで利用され、数多くの製品が販売されている。

1-2 ソフトウェアライセンスによる分類

ソフトウェアライセンスとは、ソフトウェアを利用する際に守らなければならない事項（使用・改変・再配布などの可否／条件）を記述した文書のことである。一般的には、ソフトウェアの権利者と利用者が結ぶ**利用許諾契約**に含まれる。

この観点で分類したソフトウェアとしては、次のような種類がある。

- ・ **パッケージソフトウェア**
一般に市販され、使用に関して制限が設けられている。
- ・ **フリーウェア**（**フリーソフトウェア**）
無償で配布されているが、改変・再配布などに制限が設けられている。
- ・ **シェアウェア**
無料の試用期間後に有料となること以外は、フリーウェアと同じである。

また、非営利組織の **OSI**（Open Source Initiative）が発表したオープンソースの定義（**OSD**：The Open Source Definition）に基づいた、**オープンソースソフトウェア**（**OSS**：Open Source Software）という分類がある。

【OSD によるオープンソースの要件】

1. 自由な再頒布ができること
2. ソースコードを入手できること
3. 派生物が存在でき、派生物に同じライセンスを適用できること
4. 差分情報の配布を認める場合は同一性保持を要求してもかまわないこと
5. 個人やグループを差別しないこと
6. 適用領域に基づいた差別をしないこと
7. 再配布において追加ライセンスを必要としないこと
8. 特定製品に依存しないこと
9. 同じ媒体で配布される他のソフトウェアを制限しないこと
10. 技術的な中立を保っていること

【OSS の注意点】

- ・ OSS の信頼性及び安全性は、保証されているとは限らない[無保証の原則]。
- ・ OSS は、すべての OS 又はオープンソース OS で動作するとは限らない。
- ・ OSS の開発者の権利（著作権）は、放棄されているとは限らない。
- ・ ほとんどの OSS は無償で配布されるが、絶対に無償であるとは限らない。
- ・ ソースコードを入手する仕組みがあれば、インターネット上に公開されているとは限らない。同様に、OSS を改良しても再配布しない場合は、改良部分のソースコードを公開しなくても問題にならない。

代表的な OSS として、OS の核となるカーネル部（Linux カーネル）が公開・配布されている UNIX 系 OS の **Linux** がある。UNIX 系 OS のオープン化は、UNIX の標準化を目的とする団体（The Open Group）を中心に推進され、Linux 以外にも次のような OSS がある。

- **BSD 系 OS**

UNIX 互換 OS として開発された BSD (Berkeley Software Distribution) から派生した OSS である。各 OS は、次の点を重視して開発されている。

- **FreeBSD** : インテルプロセッサでのサポートを充実すること
- **NetBSD** : 多くのコンピュータで BSD 系 UNIX を動作させること
- **OpenBSD** : NetBSD の思想を引き継ぎ、セキュリティを充実すること

- **IRIX**

シリコングラフィックス社が開発・提供している OS である。UNIX 系 OS の一つである System V をベースにして開発されている。

UNIX 系 OS 以外の代表的な OSS として、次のような種類がある。また、OSS の部品をまとめてパッケージ化した **オープンソースライブラリ** として、Perl 用ライブラリの **CPAN**, PHP 用ライブラリの **PEAR**, JavaScript 用ライブラリの **jQuery** などがある。

用途	ソフトウェア名
Web サーバ	Apache HTTP Server
DNS サーバ	BIND
メールサーバ	sendmail
Web ブラウザ	Firefox, Chrome
データベース	MySQL, MariaDB, PostgreSQL
スクリプト言語	Ruby, Python, PHP, Perl
統合開発環境	Eclipse

[LAMP/LAPP]

インターネットなどのデータベース連動系の Web アプリケーション開発で使用されている代表的な OSS の組合せである。

- **LAMP** : Linux + Apache HTTP Server + MySQL / MariaDB + PHP / Perl / Python
- **LAPP** : Linux + Apache HTTP Server + PostgreSQL + PHP / Perl / Python

※**XAMPP** : LAMP の主要ソフトウェアなどを一括インストールできるパッケージ (Apache HTTP Server, MySQL / MariaDB, PHP, Perl など)

また、OSS の使用について定めた、代表的な **OSS ライセンス** は次のとおりである。OSS ライセンスでは、複写・再配布・改変などを制限しない代わりに、二次著作物にも同じライセンスを適用しなければいけない **コピーレフト (Copyleft)** の考え方が採用されることが多い。そのため、二次著作物を配布（販売）する際に、複写・再配布・改変などに制限のある別ライセンスと併用する **デュアルライセンス** が使用されることもある。

- **GPL** (GNU General Public License)／**LGPL** (GNU Lesser GPL)

フリーソフトウェア財団 (FSF : Free Software Foundation) が作成したライセンスで、自由に利用できるソフトウェアに対して適用される。主に、FSF が推進する UNIX 互換ソフトウェア開発プロジェクト (**GNU/GNU プロジェクト**) の成果物 (フリーコンパイラ **GCC** (GNU Compile Collection) など) に適用される。代表的なコピーレフトのライセンスである。

- **MPL** (Mozilla Public License)

Netscape 社がオープンソースにした Web ブラウザのために、Netscape 社と Mozilla Organization が作成した、コピーレフトのライセンスである。

- **BSDL** (**BSD ライセンス**)

著作権を表示し、免責条項 (無保証であることなど) を含めれば、自由に複製・配布・改変できるライセンスである。条件を満たせば、改変ソフトウェアのソースコードを公開せずに再配布できるなど、若干特殊な規定がある。

- **Apache ライセンス**

Apache ソフトウェア財団による BSD スタイルのライセンスであるが、そのバージョンによっては GPL と両立する。

- **Apache Tomcat** : Java サーブレット/JSP を処理するサーバ用 OSS
- **Apache Hadoop** : 多数のサーバで構成された大規模な分散ファイルシステム機能を提供し、分散処理フレームワーク MapReduce による大規模データの分散処理を実現する OSS (HDFS にデータを蓄積し、一括して分散バッチ処理を行う)
- **Apache Kafka** : ストリーミングデータを利用した分散メッセージ処理を実現する OSS
- **Apache Spark** : MapReduce を改善した、大規模データの高速分散処理を実現する OSS
- **Apache Storm** : データストリームのリアルタイム処理を実現する OSS

OSS については、数多くの **オープンソースコミュニティ** (OSS の開発・改善・情報交換を目的とする団体) で、世界中の有志 (技術者、利用者、ボランティアなど) が、ネットワークなどを通じて研究・開発を進めている。OSS を積極的に採用する企業も増加傾向にあることから、今後の更なる発展が期待される分野である。

2 OS（オペレーティングシステム）

OS（オペレーティングシステム）とは、ハードウェアなどの資源を管理し、有効に活用するためのソフトウェアである。

2-1 OSの機能と構成

OS（オペレーティングシステム）を利用する目的として、主に次の三つがあげられる。

1. 資源の有効利用

コンピュータに関わる資源（ハードウェア的資源、ソフトウェア的資源、人的資源）を有効に活用する。資源の代表的な有効活用方法として、複数のプログラムで資源を共有して見かけ上同時に実行する**多重プログラミング**などがあり、スループットの向上やレスポンスタイム（応答時間）の短縮といった効果が期待できる。

2. RASISの向上

コンピュータシステムの信頼性を向上させ、安全性を確保する。障害の発生した命令を再試行することにより MTBF を長くしたり、障害状況を記録しておくことで MTTR を短くしたりする。

3. 使いやすさの提供

利用者の負担を軽くするため、操作性を向上させる。プログラムを実行させるたびにオペレータが操作していたのでは手間がかかるため、ジョブ（プログラム）の連続処理ができるようにする。

これらの目的を達成するために、OS は次のような管理機能を提供する。

管理機能	役割
ジョブ管理	人間から見た仕事（ジョブ）の実行を管理する。
タスク管理	機械から見た仕事（タスク）の実行を管理する。
記憶管理	記憶領域を有効に利用するために管理する。
データ管理	補助記憶装置のデータへのアクセスを管理する。
入出力管理	入出力装置へのアクセスを管理する。
ネットワーク管理	ネットワーク制御を管理する。
運用管理	運用に関する容易性や柔軟性を管理する。
ユーザ管理	ユーザに関する情報や制御を管理する。
セキュリティ管理	セキュリティに関する情報や制御を管理する。
障害管理	障害の検出／記録／対応について管理する。

2-2 OSの管理機能

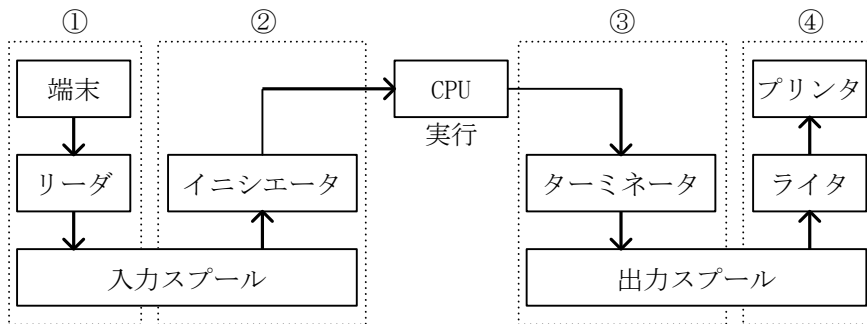
OSの提供する各種管理機能のうち、ジョブ管理、タスク管理、記憶管理をOSの三大機能と呼ぶことがある。ここでは、この三大機能を中心とした、OSの各種管理機能について説明する。

2-2-1 ジョブ管理

ジョブ管理は、人間（利用者）から見た仕事の単位である**ジョブ**の実行を管理する機能である。一般的に、ジョブは複数の**ジョブステップ**（仕事を分割した一つの単位）の集合として定義されている。ジョブ管理では、ジョブ（ジョブステップ）単位に資源の割当てを行い、ジョブの連続実行などを実現する。

ジョブ管理によるジョブ実行の一般的な流れ（手順）は、次のようになる。

【一般的なジョブ実行手順】



- ① 利用者が入力装置（端末など）からジョブ（プログラム及び実行の指示）を入力する。このとき使用されるのが、**ジョブ制御言語**（JCL: Job Control Language）である。入力されたジョブは、**リーダ**によって読み込まれ、いったん入カスプールに蓄積される。
- ② 入カスプールの実行待ちジョブの中から実行するジョブを選択し、**イニシエータ**が実行準備（資源割当てなど）をした後、ジョブを実行する。
- ③ ジョブの実行が終了した後、**ターミネータ**により後処理（資源解放など）が行われる。このとき出力結果がある場合は、出カスプールに蓄積される。
- ④ 出カスプールに蓄積されている出力結果の中から出力するものを選択し、**ライタ**が出力装置（プリンタなど）に結果を出力する。

ジョブ管理では、この一連のジョブ実行手順を実現するために、次の(1)～(3)の機能を用いている。

(1) スプーリング

スプーリングは、周辺装置（入出力装置）の動作を、処理装置の動作と独立に並行して行わせる方式である。具体的には、主記憶装置と入出力装置（低速）との間のデータ転送を、高速な補助記憶装置（**スプール**）を介して行う方法である。CPU をデータ転送待ち時間から解放して、システム全体の処理能力（スループット）を向上させる。スプーリングのようにデータを蓄える記憶領域（バッファ）を利用して、アクセス速度差のある機器間で情報をやり取りする方法を**バッファリング**と総称することもある。

(2) ジョブスケジューラ

ジョブスケジューラは、ジョブを自動的に実行するプログラムである。ジョブの制御の流れを管理するとともに、実行待ちジョブの中から実行するジョブを選択する**ジョブスケジューリング**も行う。

〔ジョブスケジューリングの方式〕

・優先順位方式（プライオリティスケジューリング）

各ジョブに割り当てられている優先順位に従って、優先順位の高いジョブから実行する方式である。対話型処理とバッチ処理が混在するようなシステムで対話型処理の優先順位を高くしておくことによって、対話型処理の応答性能を向上させるときなどに有用である。ただし、CPU を多用するジョブ（CPU バウンド）の優先順位が高いと、CPU が専有されてしまい、他のジョブの CPU 待ち時間が長くなってしまう場合もある。

・タイムスライス方式

ある一定時間ごとに、ジョブに CPU の使用权を割り当てて実行する方式である。OS によってジョブを強制的に切り替え、連続して実行することができるので、CPU の空き時間がなくなり、スループットが向上する。

・FCFS 方式（FCFS : First-Come First-Served）

ジョブを到着順に処理する方式である。ジョブ間に公平に CPU を割り当てるが、処理時間の長いジョブが先に投入された場合は、後続ジョブが待たされてスループットやレスポンスタイム（応答時間）が低下することもある。先入れ先出し法（FIFO : First-In First-Out）ともいう。

(3) マスタスケジューラ

マスタスケジューラは、主として利用者とコンピュータの間のインタフェースを管理するプログラムである。利用者とコンピュータの間での、ジョブ実行に関する指示（JCL など）や各種メッセージのやり取りを実現する。

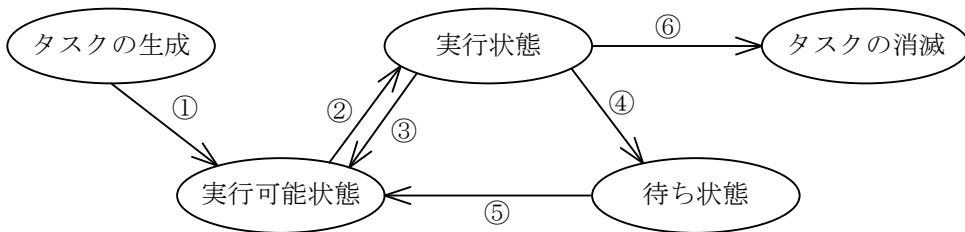
2-2-2 タスク管理

タスク管理とは、機械（コンピュータ）から見た仕事の単位である**タスク**の実行を管理する機能である。なお、OSによってはタスクのことを**プロセス**と呼ぶため、**プロセス管理**といわれる場合もある。

タスクは、CPU を割り当てることにより実行可能となる。そのため、タスク管理によるタスク実行の流れ（手順）とは、タスクに無駄なく CPU を割り当てる手順といえる。このとき、複数のタスクを並行して管理（実行）することを**マルチタスク**という。人間（利用者）から見て複数のプログラムが同時に実行されているように見える**多重プログラミング（マルチプログラミング）**は、このマルチタスクにより実現されている。

タスク管理によるタスク実行の一般的な流れ（手順）は、次のように**状態遷移図**（時間の変化や行動などに応じて変化する状況を表す図）にまとめられる。

【タスクの状態遷移図】



- ① 実行すべきジョブ（ジョブステップ）からタスクを生成する。生成したタスクにはCPU以外の資源が割り当てられ、CPUが割り当てられれば実行できる**実行可能状態**となる。
- ② タスクにCPUが割り当てられると、タスクは**実行状態**となる。
- ③ 実行状態のタスクより優先度の高いタスクが実行可能状態になったり、実行中のタスクがCPUの割当て時間（タイムクォンタム）を使い切った場合、実行中のタスクを一時中断して実行可能状態に戻し、CPUの割当てを待つことになる。
- ④ 実行中のタスクが資源（例えば、入出力命令の実行）を要求すると、要求した資源がそろうまでタスクの実行は中断されて**待ち状態**となる。
- ⑤ 待ち状態にあるタスクの要求資源がそろった時点で、タスクは実行可能状態となる。
- ⑥ 何度かのCPU割当てにより実行を終了したタスクは、消滅する。

なお、実行中のタスクを一時的に停止させ、管理用のプログラムを実行させるためには**割込み**が発生する。

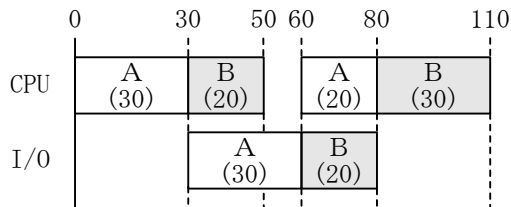
例. 二つのタスク A, B を同時に実行した場合, 二つのタスクが処理を終了するまでにかかる時間は何ミリ秒か。なお, タスク A の優先度のほうが高いものとする。

タスク A	CPU (30)	I/O (30)	CPU (20)

タスク B	CPU (20)	I/O (20)	CPU (30)

()内は処理時間 (ミリ秒)

<タスクスケジューリングの結果>



- ① タスク A, B が生成されて, 実行可能状態になる。
- ② 二つのタスクのうち, 優先度の高いタスク A に CPU が割り当てられ, タスク A が実行状態になる。
- ③ タスク A の CPU 処理が終了して I/O 処理を要求すると, SVC 割込みが発生してタスク A は待ち状態になる。I/O は使用されていないので, タスク A の I/O 処理が開始される。
- ④ タスク B に CPU が割り当てられ, タスク B が実行状態になる。
- ⑤ タスク B の CPU 処理が終了して I/O 処理を要求すると, SVC 割込みが発生して, タスク B は待ち状態になる。なお, I/O はタスク A が使用しているので, タスク B は I/O が空くのを待つことになる。
- ⑥ タスク A の I/O 処理が終了すると, 入出力割込み (入出力完了割込み) が発生して, タスク A は実行可能状態になる。この結果, I/O が空くのでタスク B の I/O 処理が開始される。
- ⑦ タスク A に CPU が割り当てられ, タスク A が実行状態になる。
- ⑧ タスク A の CPU 処理が終了すると, タスク A の実行が完了するので, タスク A は消滅する。
- ⑨ タスク B の I/O 処理が終了すると, 入出力割込み (入出力完了割込み) が発生して, タスク B は実行可能状態になる。
- ⑩ タスク B に CPU が割り当てられ, タスク B が実行状態になる。
- ⑪ タスク B の CPU 処理が終了すると, タスク B の実行が完了するので, タスク B は消滅する。

タスク管理では、この一連のタスク実行手順（マルチタスク）を実現するために、次の(1)～(3)の機能を用いている。

(1) ディスパッチャ

ディスパッチャは、実行可能状態にある複数のタスクの中から、実行するタスクを選択（**ディスパッチ**）するタスク管理プログラムである。実行するタスクを選択する基準は、タスクスケジューリングの方法により決定する。

〔タスクスケジューリングの方法〕

- ・ **プリエンプティブ方式**（**プリエンプション方式**）

実行中のタスクが、OS の制御により一時的に中断される（プリエンプションが発生する）方式である。

- ・ **優先度順方式**

各タスクに割り当てられている優先度に従って、優先度の高いタスクから CPU を割り当てる方式である。割り当てた優先度を変えない**静的優先度順方式**と、優先度が低いタスクがいつまでも実行されない状態（**スタベーション**）が発生しているとき、優先度を上げる**エージング**を可能とする**動的優先度順方式**がある。優先度順方式のように、事象（イベント）の発生をトリガ（きっかけ）とする方式を**イベントドリブン方式**という。

- ・ **処理時間順方式**

：処理予定時間の短いタスクに高い優先度を与え、早めに処理をしていく方式である。処理予定時間の長いタスクが CPU 割当てを待ち続ける可能性が高くなる。

- ・ **ラウンドロビン方式**

実行可能状態のタスクを到着順に待ち行列で管理し、一定時間（**タイムクウォンタム**）ごとに待ち行列の先頭にあるタスクに CPU を割り当てる方式である。一定時間を使い切っても処理が終了しないタスクは、待ち行列の末尾に回される。ラウンドロビン方式のように、一定の微小時間をタスクに割り当てる方式を**タイムスライス方式**という。

- ・ **ノンプリエンプティブ方式**（**ノンプリエンプション方式**）

実行中のタスクが、OS の制御によって中断されない方式である。

- ・ **到着順方式**

実行可能状態にあるタスクを到着順に待ち行列で管理して、実行中のタスクが終了するごとに待ち行列の先頭にあるタスクに CPU を割り当てる方式である。

(2) セマフォ

セマフォは、タスクに資源を割り当てるにあたり、タスク間で同期をとって共有資源を管理する相互制御（排他制御）に用いられる。実際には、資源要求時の P 操作と資源解放時の V 操作により、タスク間の制御を行っている。

〔P 操作／V 操作〕

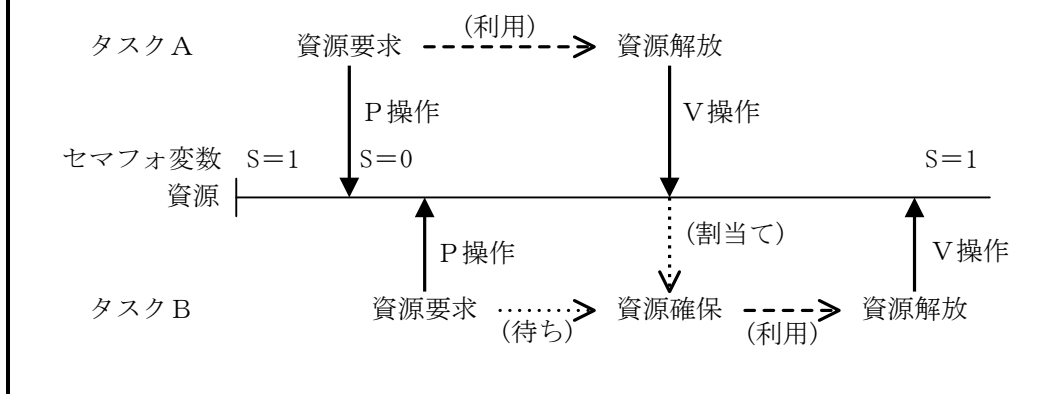
資源ごとのセマフォ変数の値により、資源の確保と解放を行う操作である。資源を単一にしか割り当てられないため、セマフォ変数の値が 0 と 1 しかとれないセマフォは、特に **2 進セマフォ**と呼ばれる。

・ P 操作

資源要求が発生した場合の操作である。セマフォ変数の値が 1 以上であれば資源を確保し、セマフォ変数の値から 1 を減算する。セマフォ変数の値が 0 であれば、資源が解放されるまで待ち状態になる。

・ V 操作

資源を解放する場合の操作である。該当資源待ちのタスクがあれば資源を割り当て、なければセマフォ変数の値に 1 を加算する。



(3) スレッド

スレッドは、タスクを細分化したもので、CPU 資源のみを割り当てる単位である。軽量プロセス（軽量タスク）とも呼ばれる。メモリなどは、細分化する前のタスクに割り当てられたものを使用する。スレッドは、すべての OS がもっている機能ではない。

なお、ソフトウェア制御には、割込み以外に、機器や資源の状態を監視（問合せ）するポーリング制御という考え方もある。

2-2-3 記憶管理

記憶管理は、コンピュータの記憶領域を有効に利用するために管理する機能である。記憶管理は、**実記憶管理**と**仮想記憶管理**に分類される。

(1) 実記憶管理

実記憶管理は、主記憶装置の記憶領域を有効に利用するための管理機能である。実行するプログラムは主記憶装置内に記録されている必要があるが、記憶できる容量には限界があるため補助記憶装置を利用する方式である。

① 区画方式

主記憶装置を区画に区切って管理する方式である。区画方式は、**固定区画方式**と**可変区画方式**に分類される。

a : 固定区画方式

主記憶装置の使用可能な記憶領域を、決められた大きさに区切って利用する方式である。プログラムやデータを必要なときに補助記憶装置から区画に読み込み（**ロールイン**）、必要なくなったら補助記憶装置に書き出す（**ロールアウト**）。

・単一固定区画方式

主記憶装置の使用可能な記憶領域を、単一の固定区画として使用する方式である。制御は簡単であるが、主記憶装置を一つのプログラムだけで使用するため未使用領域が多くなる。

・多重固定区画方式

主記憶装置の使用可能な記憶領域を、複数の固定区画として使用する方式である。区画ごとにプログラムを記録できるが、区画の容量より大きなプログラムは記録できない。

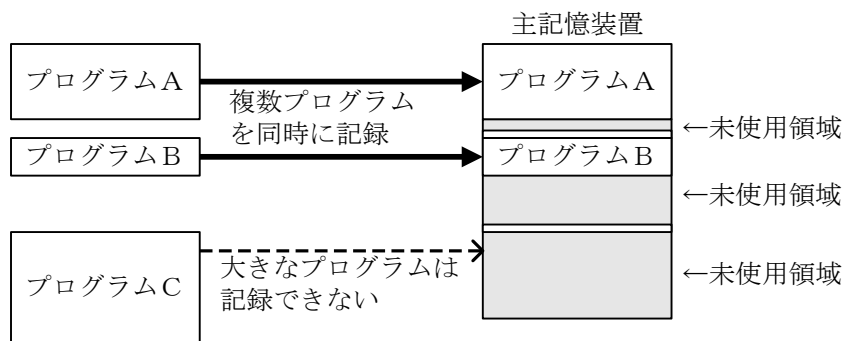


図 3-3 固定区画方式（多重固定区画方式）

b : 可変区画方式

主記憶装置の使用可能な記憶領域を、プログラムの容量に合わせて区画として区切る方式である。

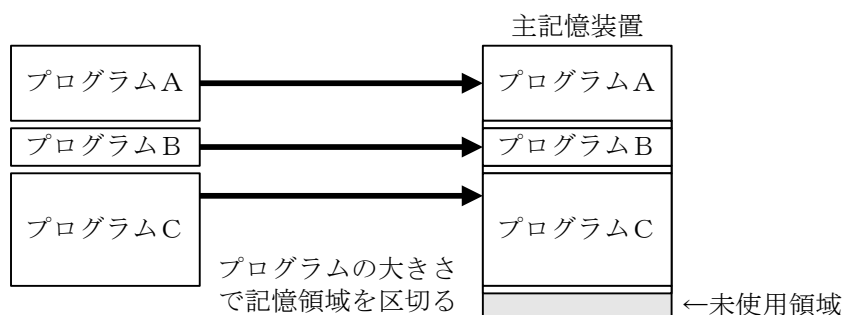


図 3-4 可変区画方式

可変区画方式では、プログラムの実行後も区画は残るため、細切れの未使用領域が多発する。これを**フラグメンテーション**（記憶領域の断片化）という。この場合、細切れの未使用領域を一つにまとめる**メモリコンパクション**を実行する。

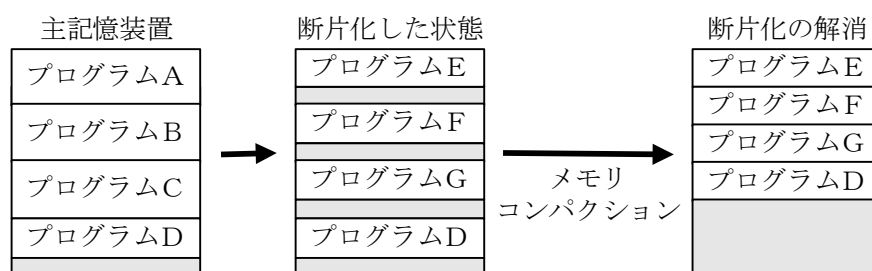


図 3-5 フラグメンテーションとメモリコンパクション

また、主記憶装置（メモリ）を利用したプログラム（OS、アプリケーションなど）のバグなどが原因となり、動作中に確保した領域が解放されずにメモリ使用量が増加し続ける現象（**メモリリーク**）が発生する可能性もある。メモリリークが発生すると主記憶装置の使用可能な記憶領域が減少して、深刻な問題となる。そこで、不要となった領域を自動的に解放する**ガーベジコレクション**を実行し、空き領域として再利用できるようにする。

【メモリプール】

プログラム（OS、アプリケーションなど）がメモリ領域を一括して確保してから、必要に応じて確保したメモリ領域を割り当てて利用する方式である。メモリプール管理でも、同じ大きさのメモリブロックを割り当てる固定長方式と、必要な大きさのメモリブロックを割り当てる可変長方式が用いられる。

② オーバレイ方式（セグメント方式）

主記憶装置の使用可能な記憶領域以上の大きさのプログラムを、実行するための方式である。プログラムを、同時に実行されることのない複数の部分（セグメント）に分割し、実行時にセグメントを交互に補助記憶装置から転送する方式である。

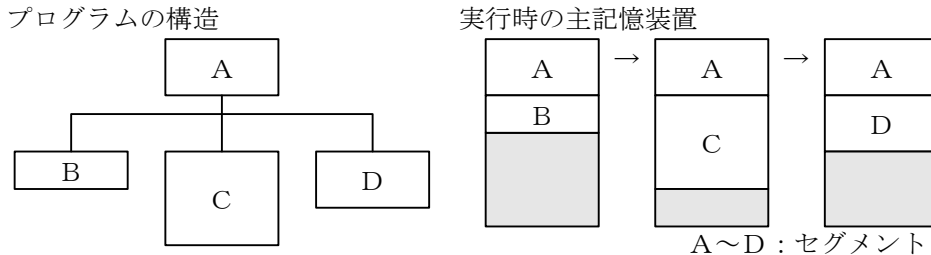


図 3-6 オーバレイ方式

③ スワッピング

多重プログラミングなどにおいて、プログラムの実行中に優先度の高いプログラムなどを実行するために、現在のプログラムの実行を一時的に停止して補助記憶装置に退避させることである。主記憶装置に余裕ができた時点又は実行を再開する時点で、退避したプログラムを補助記憶装置から主記憶装置に戻して中断時点からの処理を再開する。主記憶装置から補助記憶装置に退避する作業を**スワップアウト**、戻す作業を**スワップイン**という。スワッピングが多発すると、プログラム処理に関係ない時間（オーバーヘッド）が多くなるため、処理効率が急激に低下する。

(2) 仮想記憶管理

仮想記憶管理とは、主記憶装置よりも大きな論理的記憶空間（仮想記憶空間又は論理アドレス空間）を仮定し、そこに記録されているプログラムを実行する仮想記憶方式という考え方を実現する機能である（実際には、主記憶装置上のプログラムしか実行できないので、実行する部分を補助記憶装置から主記憶装置にロード（転送）して実行する）。

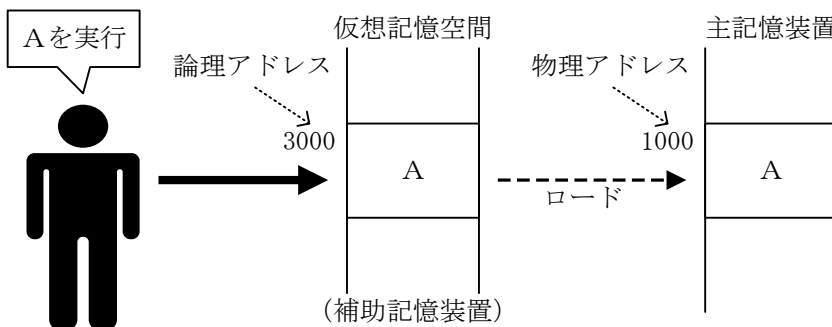


図 3-7 仮想記憶方式のイメージ

仮想記憶管理は、記憶領域の管理方法の違いによって、次のように分類される。

- ・ **ページング方式**

主記憶装置や補助記憶装置の記憶領域を一定の大きさ（ページ）に区切って管理する方式である。記憶装置間のやり取りはページという固定長の単位で行われるため、記憶領域に無駄が発生しにくい。補助記憶装置のページを主記憶装置にロードすることを**ページイン**、逆を**ページアウト**と呼ぶ。

- ・ **セグメント方式**

プログラムなど意味のある単位（セグメント）に分割し、セグメント単位で主記憶装置と補助記憶装置の間でやり取りを行う方式である。

- ・ **セグメントページング方式**

一定の大きさに区切ったページのうち、論理的に関連のあるページをセグメントとしてまとめて取り扱う方式である（論理的な結びつきに関係なく、複数ページをセグメントとする考え方もある）。

仮想記憶管理では、仮想記憶上の番地（論理アドレス）と主記憶上の番地（物理アドレス）を変換する必要がある。この変換のことをアドレス変換といい、**動的アドレス変換機構（DAT : Dynamic Address Translator）**というハードウェアにより行われる。

セグメント方式では、セグメントの開始位置（物理アドレス）を記録したセグメントアドレステーブルを参照する**ベースアドレス方式**でアドレス変換が行われる。一方、ページング方式では、各ページの物理アドレスが記録されているページアドレステーブルを動的アドレス変換機構が参照することでアドレス変換が行われる。

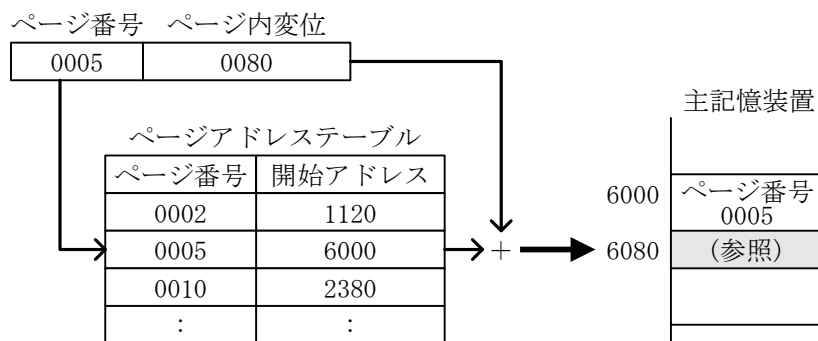


図 3-8 ページング方式におけるアドレス変換

このとき、ページアドレステーブルに登録されていない（主記憶装置に登録されていない）ページを参照すると、動的アドレス変換機構によって**ページフォールト**という割込みが発生する。ページフォールトが発生すると、参照ページが主記憶装置にロードされて、ページアドレステーブルも更新される。このとき、主記憶装置に空き領域がない場合は、主記憶装置上にあるページ（置換え対象ページ）を一つ選択してページアウトし、その空き領域に参照ページをページインする。

ページフォールトが多発すると、ページイン／ページアウトが頻発して処理効率が低下する**スラッシング**という状態になる。そのため、なるべくページフォールトが発生しないようにするため、その時点以降に参照される確率が低いページをページアウトする必要がある。このようなページを選択するための手法を**ページリプレースメントアルゴリズム**といい、代表的なものとして次の三つがある。

- ・ **FIFO 法** (FIFO : First-In First-Out ; 先入れ先出し)
最も古くから主記憶上に存在するページをページアウトする。
- ・ **LRU 法** (LRU : Least Recently Used)
最後に参照されてから最も経過時間の長いページをページアウトする。
- ・ **LFU 法** (LFU : Least Frequently Used)
参照された回数が最も少ないページをページアウトする。

例. 主記憶装置のページ容量が3ページで、ページが { 1, 2, 3, 2, 1, 4, 1, 2 } の順に参照されるとき、FIFO 法およびLRU 法ではそれぞれページインがどのように行われるか。

参照ページ	FIFO 法			LRU 法		
1	①			①		
2	1	②		1	②	
3	1	2	③	1	2	③
2	1	②	3	1	②	3
1	①	2	3	①	2	3
4	④	2	3	1	2	④
1	4	①	3	①	2	4
2	4	1	②	1	②	4

○…参照ページ, …ページイン

なお、仮想記憶方式によって、主記憶装置の見かけの記憶容量を大きくすることはできる。しかし、主記憶装置の記憶容量が小さすぎると、スラッシングが発生して処理効率が低下する。したがって、メモリ不足で動作しないアプリケーションを動かすようなとき、仮想記憶の記憶容量を大きくすることは一時的な対処にしかない。このような場合には、メモリの増設や必要のないアプリケーションの使用を停止するなど、利用できる記憶領域を増やすことを検討する必要がある。

2-2-4 その他の管理機能

(1) データ管理

データ管理は、補助記憶装置に記録されたデータへのアクセス方法を、補助記憶装置に依存しないインタフェースで提供する機能である。データを利用するには、補助記憶装置の特性やデータの記録方法など、物理的な点について考慮する必要がある。しかし、このようなことを利用者が考えていては、コンピュータを簡単に利用することはできない。そこで、利用者が意識せずにデータを扱えるようにするのがデータ管理の役割となる。データ管理では、**ファイル編成方式**やファイルの**アクセス方式**について管理する（ファイル編成方式やファイルのアクセス方式の詳細については、第4章で説明する）。

(2) 入出力管理

入出力管理は、コンピュータで処理するデータを入力したり、処理結果を出力するために入出力装置を制御する機能である。入出力要求（**SVC 割込み**）が発生したとき、**入出力制御方式**に応じた処理（例えば、チャンネル制御方式であれば入出力チャンネルにチャンネルプログラムを送るなど）を行い、入出力完了時には**入出力割込み**によって入出力が完了したことを通知する。また、入出力装置の概念を拡張して、デバイスドライバなどを利用した周辺装置とのインタフェースを提供する機能も入出力管理の役割に含めることがある。

(3) ネットワーク管理

ネットワーク管理は、通信に関するネットワーク制御を行う機能である。**ネットワーク制御プログラム**を利用して、**LAN**（Local Area Network）や**WAN**（Wide Area Network）と接続するインタフェースを提供し、**通信プロトコル**（**OSI 基本参照モデル**、**TCP/IP**など）を用いた**プロトコル制御**や、データのやり取りに関する順番を管理する**メッセージ制御**などを行う（ネットワークの詳細については第5部で説明する）。

(4) 運用管理

運用管理は、システム始動処理（OSの初期化）からシステム終了処理までのコンピュータシステムの運用状態について管理する機能である。運用の**スケジューリング**や、さまざまな情報（システム資源の使用状況、課金情報など）を**モニタリング**によって記録することなど、システムを効率的に運用するための補助的なサポートを行い、運用に関する容易性や柔軟性を実現する。なお、運用管理には、後述する**ユーザ管理**の機能が含まれる場合もある。

(5) ユーザ管理

ユーザ管理は、利用者（ユーザ）ごとにユーザ ID などを発行し、ユーザ情報やシステム環境を保存する**プロファイル機能**、**アカウント機能**などを提供する機能である。

アカウント機能におけるアカウントとは“権利”のことである。システムを利用するための**システム利用権**、データを利用するための**アクセス権**（特にファイルに関する権利は**ファイルアクセス権**という）、端末を利用するための**端末利用権**など、コンピュータ（又はシステム）を利用するためにはさまざまな“権利”が必要となる。このような権利を、利用者（ユーザ）ごとに設定／変更／削除することをアカウント管理という。このとき、ユーザ識別に利用するユーザ ID（およびパスワードなど）を**ユーザアカウント**（利用者に与えられた権利）と呼ぶこともある。ユーザアカウントには、一般的な利用者のほかに、コンピュータ又はシステムに対する管理者権限（すべての権限）をもつ**スーパーユーザ**（Windows 系 OS では **Administrator**、UNIX 系 OS では **root**）、一時的な利用者を意味する**ゲスト**などの種類がある。また、システムに接続された全端末でユーザアカウントを共有するために、**ディレクトリサービス**を用いることがある。ディレクトリサービスは、ネットワーク経由で資源情報を一元管理するためのデータベースサービスであり、アクセスにはプロトコル **LDAP**（Lightweight Directory Access Protocol）が一般的に用いられる。

(6) セキュリティ管理

セキュリティ管理は、コンピュータの安全性を保つために、情報の機密性、完全性、可用性を維持する機能である。ユーザ管理のアカウント機能などを利用して資源への不正なアクセスを禁止する**アクセス制御**、記録されたデータを暗号化して情報漏えいを防止する**暗号化制御**、外部からの不正侵入を検出・防御する制御などが該当する。これらのセキュリティ制御が有効に機能しているかを確認するため、利用状況の履歴（ログ）を記録する**ロギング機能**や、ログイン／ログアウトなどのイベントを検査（監査）する**オーディット機能**などにより、**アカウントビリティ**（責任追跡性）が確保されていることを**信頼プロセス**として検証する（セキュリティの詳細については第6部で説明する）。

(7) 障害管理

障害管理は、ハードウェア、ソフトウェアを監視し、障害発生時に対策処理を行う機能である。対象とする障害は、ハードウェアを原因とする**ハードウェア障害**、ソフトウェアを原因とする**ソフトウェア障害**の両方である。障害の状況に応じて、障害箇所を切り離してシステムを再構築する、システムを再起動（リスタート）する、などの適切な対策を実施する。また、誤動作が生じたときの障害状況・原因・対策を記録しておくことで、同種の障害が発生したときの修復時間（MTTR）短縮に役立つ。

3 プログラム言語と言語プロセッサ

プログラム言語は、コンピュータに仕事を指示するプログラムを作成するための言語である。このプログラム言語で記述されたプログラムを、コンピュータの理解できる言葉に翻訳するのが言語プロセッサである。

3-1 プログラム言語の分類

コンピュータ内部では、情報はすべて電氣的に記録される。すなわち、コンピュータに理解できるのは、0と1の二つの状態しかない。そこで、最初のプログラム言語として、0と1の組合せでコンピュータに仕事の指示を与える**機械語**が利用されていた。

しかし、機械語（0，1）でプログラムを書くことは、人間にとっては非常にやりにくく間違いも起こりやすい。そこで、いくつかの0と1の組合せを記号に対応させるプログラム言語が考えられた。これを、**アセンブラ言語**又は**シンボリック言語**という。

しかし、アセンブラ言語も結局は0と1の組合せを記号化したものである。そのため、アセンブラ言語でプログラムを作成するには、コンピュータ内部の仕組み（ハードウェアなど）を詳しく知っている必要があった。そこで、人間の考える処理を、人間にわかりやすく表記できるプログラム言語が開発された。これを、**高水準言語**と分類し、それまでに使用していた機械語やアセンブラ言語を**低水準言語**として区別した。

【高水準言語の分類】

- ・ **手続型言語**

演算命令やデータ操作命令などの実行（手続）によって処理手順（アルゴリズム）を表現する、手続型プログラムを記述する言語である。

- ・ **関数型言語**

関数を入力データに作用させることによって出力データを得る、関数型プログラムを記述する言語である。

- ・ **論理型言語**

問題に内在する関係や条件などを論理式によって記述する、論理型プログラムを記述する言語である。

- ・ **オブジェクト指向言語**

データと処理（手続）を一体化したオブジェクトという単位を中心にした、オブジェクト指向プログラムを記述する言語である。

- ・ **スクリプト言語**

簡易プログラム（スクリプト）を記述する言語である。

現在，一般的に使用されているプログラム言語として，代表的なものを説明する。

3-1-1 アセンブラ言語

アセンブラ言語は，機械語に最も近い言語で，アセンブラ言語と機械語の命令は1対1に対応する。アセンブラ言語の体系・文法は，対応するコンピュータによって異なる。

- **CASL II**

基本情報技術者試験で使われているプログラム言語で，仮想計算機 COMET II のもとで動作するアセンブラ言語である。

3-1-2 手続型言語

手続型言語は，手続を基本単位として，処理手順（アルゴリズム）を記述するプログラム言語である。ほとんどの手続型言語は，記述したプログラム（原始プログラム）を機械語のプログラム（目的プログラム）に一括翻訳してから実行する**コンパイル方式**を採用している。

手続型言語は，コンピュータの歴史でいえば，ほとんどが初期の時代に開発されているため，現在では使用されていない（又は用途が限定される）ものが多い。ただし，以降のプログラム言語に大きな影響を与えた手続型言語も多く，また論理的な思考を身に付ける意味からも，技術者として一つは習得しておくことを推奨したい。

- **C**

AT&T ベル研究所で開発された UNIX OS を記述するためのプログラム言語である。移植性が高く，現在も PC 上のシステム開発などで利用されている。

- **COBOL**

事務処理向きのプログラム言語である。グラフィックスを活用したプログラム開発などには不向きであるが，金融系システムなどで今も利用されている。

- Fortran

科学技術計算向きのプログラム言語である。

- PL/I

Fortran と COBOL の特性を兼ね備えたプログラム言語である。

- Pascal

プログラミング教育用に開発されたプログラム言語である。

- BASIC

初心者向きの会話型プログラム言語である。

3-1-3 オブジェクト指向言語

オブジェクト指向言語は、データとデータの処理（手続、メソッド）を一体化したオブジェクトという単位でとらえ、そのオブジェクト単位でプログラムを作成していくプログラム言語である。**オブジェクト指向**は、処理（手続）に注目するのではなく、処理の対象となるデータに注目した考え方である。

- **C++**

C にオブジェクト指向の考え方と機能を取り入れたオブジェクト指向言語である。C よりも、さらに汎用性、再利用性が高い。

- **Java**

アメリカの Sun Microsystems 社が、C++をもとに開発したオブジェクト指向言語である。Java で開発したプログラム（**Java アプリケーション**）は、**Java 仮想マシン**（JVM: Java Virtual Machine）を実装した環境であれば、異なるハードウェアや異なる OS 上でも稼働させられる**マルチプラットフォーム**対応である。コンピュータの機種や OS に依存しないので、インターネットで動作する Web アプリケーションや分散システム環境で利用される。現在は、**実行時コンパイラ**（JIT コンパイラ）を利用したコンパイル方式が主流である。

- **Java Servlet**（Java サーブレット）

クライアントからの要求に応じて、Web サーバで実行される Java プログラムである（**サーブレット**とは、サーバ側で実行されるプログラムのことである）。

- **Java Applet**（Java アプレット）

クライアントが、Web サーバからダウンロードして実行する Java プログラムである（**アプレット**とは、クライアント側にダウンロードして実行されるプログラムのことである）。

- **JavaBeans**

Java で開発されたプログラムを、アプリケーションの部品（コンポーネント）として扱うための規約である。

- **VB**（Visual BASIC）

マイクロソフト社が開発したオブジェクト指向言語で、ウィンドウやツールボックスを使ったビジュアルな環境でのプログラムの作成が可能である。VB をもとにアプリケーションとの連携を可能にした **VBA**（VB for Application）と呼ばれるマクロ言語は、Excel, Word, Access などの製品と対応している。

- **マクロ**

アプリケーションで行われる特定の操作手順を、自動的に連続実行できる仕組みである。操作手順を記録する方法と、マクロ言語で処理手順を記述する方法がある。

3-1-4 スクリプト言語

スクリプト言語は、一連の命令を集めて、簡単に実行できるようにした簡易プログラム（スクリプト）を、利用者（エンドユーザ）でも開発することができる簡易プログラム言語である。マウスなどの GUI 環境を利用して、さまざまな指定をすることでプログラムを作成できる。プログラムの事前コンパイル（一括翻訳）が不要な**インタプリタ方式**で逐次解釈して実行されるものが多いが、コンパイル方式で実行されるものもある。

- **JavaScript**

Web ブラウザで動作するプログラムを記述するためのオブジェクト指向スクリプト言語である。Java と似ている部分もあるが全くの別物で、Web ページのコード中に命令を直接書き込むことでプログラミングできる。ホームページ作成などに利用される。

- **ECMAScript**

Ecma International（旧 ECMA；ヨーロッパ電子計算機工業会）によって規格化された JavaScript の標準規格である。

- **VBScript**

Web ブラウザで動作するプログラムを記述するためのオブジェクト指向スクリプト言語である。VB の機能の一部を、インターネットのホームページ上で使える。

- **PostScript**

Adobe Systems 社が開発したページ記述言語で、テキストデータや図形などをページ単位で記述する。主に、プリンタへの出力を意識した DTP ソフトで採用されている。

- **Ruby/Python/PHP/Perl**

Web サーバがブラウザからの処理要求に応じてアプリケーションプログラムを起動する仕組みを、**CGI**（Common Gateway Interface）という。この仕組みにより起動されるアプリケーション（CGI プログラム）を記述する言語である。それぞれの言語ごとに特徴はあるが、テキスト処理能力に優れている。

【共通言語基盤（CLI：Common Language Infrastructure）】

共通言語基盤（CLI）は、異なるシステム環境で、それらの環境の独特な性質を考慮に入れた書換えを行うことなく、複数の高水準言語で書かれたアプリケーションを動作させることを可能とする実行環境のことである。元々は、マイクロソフト社の .NET Framework 用仮想マシンの仕様であったが、さまざまな高水準言語で記述したプログラムを共通のプラットフォームで実行できる環境（仕様）として、**JIS X 3016（ISO/IEC 23271）**などで標準化されている。

3-1-5 マークアップ言語

マークアップ言語は、印刷や画面表示を目的としたプログラム言語の一種である。テキスト（文字列）を**タグ**と呼ばれるマークで囲み、属性情報を付加する。文書の構成要素（文・段落・タイトル・図表など）と関係を、簡単に定義することができる。

① **SGML** (Standard Generalized Markup Language ; 標準一般化マーク付け言語)

ISO で標準化された汎用マークアップ言語である。文書の論理構造や意味構造、属性を記述することができるため、文書管理やデータ交換が容易に行える。SGML では、**DTD** (Document Type Definition ; 文書型定義) によって文書構造を定義している。

② **HTML** (HyperText Markup Language)

SGML を拡張して、文章以外の図形や静止画、動画なども取り扱えるようにしたハイパertext記述言語である（ハイパertextは、文字情報のリンクによって別のコンテンツと関連付けるコンテンツである）。現在は、HTML Living Standard を標準とする方向で進められている。

HTML は**開始タグ**と**終了タグ**でテキストを囲むことにより、文字の大きさや色、貼り込む画像ファイル、リンク先の Web ページなどを指定する。仕様は規格化されているが、Web ブラウザを提供しているメーカーが機能を拡張した独自仕様としているため、特定メーカーのブラウザでしか使用できないタグもある。

また、HTML と切り離して Web ページのレイアウトだけを定義するスタイルシートとして **CSS** (Cascading Style Sheets ; 段階スタイルシート) が利用できる。

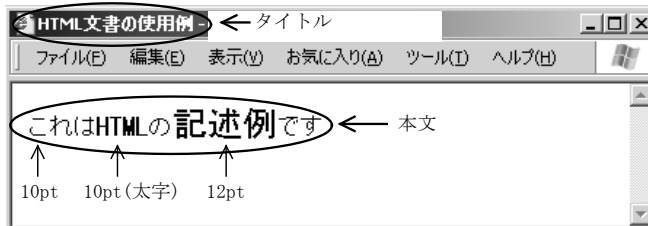
【HTML 文書の表示例】

<HTML 文書>

<TITLE>HTML 文書の使用例</TITLE>

<BODY>これはHTMLの記述例です</BODY>

<表示例（本文の標準フォントサイズを 10pt とした場合）>



③ **DHTML** (Dynamic HTML)

HTML の拡張仕様で、HTML のページに JavaScript や VBScript でスクリプトを埋め込むことによって、ユーザの操作に合わせてブラウザで動的に Web ページを作成したり、ユーザの入力データをチェックしたりすることができる。

④ XML (Extensible Markup Language)

SGML と HTML から派生した、双方向リンク可能なハイパテキスト記述言語である。インターネット環境との親和性が高いため、ネットワークを介した企業間取引などで利用されることが多い。DTD (文書型定義) を用いることで、ユーザ独自のタグを定義できる (現在は、DTD の代わりに XML Schema と呼ばれる XML 専用のスキーマ言語を利用することが多い)。HTML では終了タグを省略することができる場合もあるが、XML では空要素であっても開始タグと終了タグで囲まなければいけない。

また、XML ではスタイルシートとして、XSL (Extensible Stylesheet Language ; 拡張可能なスタイルシート言語) が利用できる。

【XML 関連技術】

• XML パーサ

XML 文書の構成要素をアプリケーションから利用するためのソフトウェアである。XML 文書をツリーで管理する DOM (Document Object Model), XML 文書を先頭から順に読み込んで解釈する SAX (Simple API for XML) などの API が利用できる。

• SOAP (Simple Object Access Protocol)

XML をベースにしたデータやサービスを呼び出すための、メッセージ交換用プロトコル (通信規約) である。

• SVG (Scalable Vector Graphics)

XML をベースにした 2 次元ベクタグラフィックス画像の形式である。

• SDL (Service Description Language)

Web サービスの機能を定義する XML ベースの言語である。現在は、WSDL (Web SDL) に吸収されている。

• Ajax (Asynchronous JavaScript + XML)

ブラウザと Web サーバが同期をとらずに XML 文書を送受信して、動的に画面を再描画する仕組みである。

⑤ XHTML (Extensible HyperText Markup Language)

SGML を拡張した HTML を、XML で再定義したマークアップ言語である。“タグの省略ができない” など HTML より仕様が厳密になっているが、現在は廃止されている。また、携帯電話や携帯端末での利用を目的とした XHTML Basic は、XHTML のモジュール化仕様によって定義されたモジュールを組み合わせたサブセットである。

【YAML (YAML Ain't Markup Language)】

YAML は、構造化されたデータの表現形式 (フォーマット) である。“YAML はマークアップ言語ではない” という再帰的な名称が付けられているが、XML が同様の目的に使われることから軽量マークアップ言語と呼ばれる場合もある。

3-2 言語プロセッサ

言語プロセッサは、機械語以外のプログラム言語で書かれたプログラムを機械語又は**中間言語**（高水準言語と機械語の間に位置する言語）に翻訳するソフトウェアである。翻訳前のプログラムと翻訳後のプログラムは、同じ仕事を処理するプログラムであっても内容が異なるため、次のように区別される。

- ・ **原始プログラム**（ソースプログラム）

機械語以外のプログラム言語で作成されたプログラムのことである。

- ・ **目的プログラム**（オブジェクトプログラム）

機械語又は中間言語に翻訳されたプログラムのことである。

3-2-1 言語プロセッサの種類

(1) アセンブラ

アセンブラは、アセンブラ言語で書かれた原始プログラムを、目的プログラムに翻訳する言語プロセッサである。アセンブラにより翻訳することを、**アセンブル**という。

(2) コンパイラ

コンパイラは、高水準言語で書かれた原始プログラムを、目的プログラムに翻訳する言語プロセッサである。コンパイラにより翻訳することを、**コンパイル**という。コンパイラで翻訳される高水準言語のことをコンパイラ言語といい、代表的なものとしてCやCOBOLなどがある。なお、高水準言語の命令は複数の機械語に翻訳されるため、言語ごとに専用のコンパイラが必要となる。

【コンパイラの翻訳手順】

- 1) **字句解析**

文字列からなる原始プログラムを字句（トークン）単位に分割する。

- 2) **構文解析**

プログラム言語の文法に従って字句構成を解析し、構文木を生成する。

- 3) **意味解析**

構文木の意味を解析し、中間コードを生成する。

- 4) **最適化**

実行時の処理効率を高めて実行時間を短縮できるように、中間コードにおける式の簡略化や構成の変更などを行う。

- 5) **コード生成**

機械語又は中間言語のコード（オブジェクトコード）に変換する。

① 字句解析

一般的なプログラム言語は、**文脈自由文法**による**形式言語**で定義されている。そのため、**正規表現**を解釈する**有限オートマトン**などを使用して字句（トークン）単位に分割する。一般的なプログラム言語の場合、識別子、定数、キーワード、演算子などを、字句（トークン）として分割する。

- ・ **文脈自由文法**

前後の文脈に依存せずに、記号の置換ができる文法である。例えば、文脈自由文法で“A”を“xy”又は“yz”と定義すると、“xyz”を“Az”又は“xA”のどちらに置換してもよいことになる。

- ・ **形式言語**

プログラム言語のように、厳密な文法で定義された言語のことである。これに対して、日常会話で使われる言語を自然言語という場合もある。

- ・ **正規表現**

文字列の形式（パターン）を定義する際に使用される表記法である。

【正規表現の表記例】

[A-Z]*[0-9]+

[A-Z]：英字 1 文字を表す

[0-9]：数字 1 文字を表す

*：直前の正規表現の 0 回以上の繰返しを表す

+：直前の正規表現の 1 回以上の繰返しを表す

※この例で定義される文字列とは、英字が 0 回以上繰り返された後に数字が 1 回以上繰り返されるものである。つまり、“AB1”や“123”は該当するが、“ABC”や“12A”は該当しない。

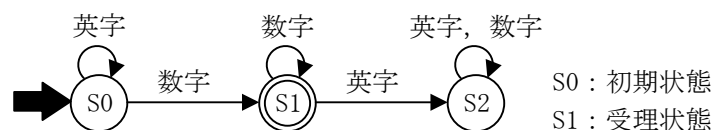
- ・ **有限オートマトン**

オートマトンとは、入力に対する処理が内部の状態によって異なることをモデル化する表記法である。有限オートマトンは、有限個の状態と遷移で構成されるオートマトンであり、**状態遷移表**や**状態遷移図**などを使用して表現されることが多い。

【状態遷移表による表現】

	S0	S1	S2	
英字 (A~Z)	S0	S2	S2	S0：初期状態 S1：受理状態
数字 (0~9)	S1	S1	S2	

【状態遷移図による表現】



② 構文解析

プログラム言語の文法に従って字句（トークン）の構成を解釈して、**構文木**を生成する。文脈自由文法では構文を **BNF 記法** 及び **構文図式** で定義することが多いため、プログラムの字句が正しい意味を構成していることを確認する作業ともいえる。

• BNF 記法 (BNF : Backus Naur Form)

プログラム言語の文法などを定義する **メタ言語**（言語を定義するための言語）である。XML などの規定にも利用されている。

【BNF 記法の表記例】

〈式〉 ::= 〈数値〉 | 〈式〉〈演算子〉〈数値〉

〈数値〉 ::= 〈数字〉 | 〈符号〉〈数字〉 | 〈数値〉〈数字〉

〈演算子〉 ::= 〈符号〉 | * | /

〈符号〉 ::= + | -

〈数字〉 ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

※この例の場合，“+4-12*6”は、次のように解釈される。

+4-12*6

→ 〈符号〉〈数字〉〈符号〉〈数字〉〈数字〉〈演算子〉〈数字〉

→ 〈数値〉〈符号〉〈数値〉〈数字〉〈演算子〉〈数値〉

→ 〈数値〉〈符号〉〈数値〉〈演算子〉〈数値〉

→ 〈数値〉〈演算子〉〈数値〉〈演算子〉〈数値〉

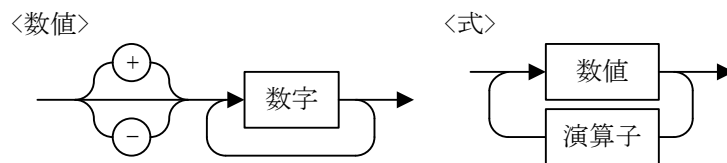
→ 〈式〉〈演算子〉〈数値〉〈演算子〉〈数値〉

→ 〈式〉〈演算子〉〈数値〉

→ 〈式〉 … 正しい〈式〉と解釈される

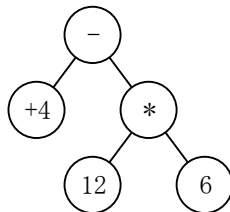
• 構文図式

BNF 記法の定義を、視覚的にわかりやすく図式化したものである。



• 構文木

構文解析の結果を木構造で図式化したものである。構文木のうち、意味解析に必要な情報だけで構成されたものを、特に**抽象構文木**ともいう。



③ 意味解析

構文解析で生成した構文木を、プログラム言語の意味的に解析して中間コードを生成する。意味的な解析には、プログラム言語の文法だけでは解析できない部分（変数名の妥当性、外部関数の呼出しなど）を解釈することも含まれる。中間コードでは、数式の表現として**四つ組み形式**や**逆ポーランド記法**を利用する場合もある。

- **四つ組み形式**（三番地文）

式を（演算子，被演算子1，被演算子2，結果）のように、四つの要素（三つの変数の番地）で表現する方式である。

例えば，“s=4-12*6”という式は、次のように表現される。

(*, 12, 6, T1)

(-, 4, T1, s)

- **逆ポーランド記法**

演算子を、演算対象の変数（式）や数値の後に書く数式の記述方法である。演算子の出現順序が演算の実行順序となるため、コンピュータ内部の数式表現として用いられる（構文木の代わりに用いられる場合もある）。

なお、演算子を前に書く**ポーランド記法**という記述方法もある。

例えば，“s=4-12*6”という式は、次のように表現される。

逆ポーランド記法：(s 4 12 6 * - =)

ポーランド記法：(= s - 4 * 12 6)

④ 最適化

中間コードを解析して、プログラム構造の変更（式の簡略化や構成変更など）を行い、実行時の処理効率を高めたコードに最適化する。

最適化の方法	内容
関数のインライン展開	関数を呼び出す場所に、呼び出す関数を展開する。
共通部分式の削除	複数の箇所に存在する同じ式の値を作業用変数に求めて、式を作業用変数に置き換える。
定数の畳込み	定数同士の計算式を、計算結果に置き換える。
定数伝播	変数を定数で置き換える。
無用命令の削除	実行結果に影響しない文を削除する。
ループ内不変式の移動	ループ内で値の変化しない式を外に移動する。
ループのアンローリング	ループ中の繰り返し処理を展開する。

⑤ コード生成

最適化した中間コードを、目的とするオブジェクトコードに変換する。基本は機械語に変換するが、いったん中間言語のコード（**Java バイトコード**や共通言語基盤の**共通中間言語**（CIL：Common Intermediate Language）によるバイトコードなど）に変換しておき、実行時に実行環境に合わせた機械語に変換する場合も多くなっている。

また、コンパイラには、その利用目的などによって、次のような種類がある。

- ・ **プリコンパイラ（プリプロセッサ）**

コンパイルの前に高水準言語のプログラムに含まれる付加的な機能などを解釈して、高水準言語のプログラムに変換するコンパイラである。

- ・ **最適化コンパイラ**

処理過程の最適化に重点を置き、コンパイル後の目的プログラムの大きさや実行速度をできるだけ小さくするコンパイラである。

- ・ **クロスコンパイラ**

機械語が異なるコンピュータの一方で、もう一方のコンピュータで動作する目的プログラムを生成するコンパイラである。同じ用途で用いられるアセンブラのことを、**クロスアセンブラ**という。

- ・ **実行時コンパイラ（Just-In-Time compiler ; JIT コンパイラ）**

原始プログラム（ソースコード）又は中間言語で記述された中間コードを、実行直前に機械語に翻訳するコンパイラである。実行開始時に時間が若干かかるが、実行環境に依存しないでプログラムを実行できるメリットがある。

(3) インタプリタ

インタプリタは、高水準言語で書かれた原始プログラムを、1 命令ずつ翻訳・実行する言語プロセッサである。コンパイラとインタプリタの違いは、次のようになる。

	コンパイラ	インタプリタ
目的プログラム	生成する	生成しない
翻訳単位	全文一括（プログラムが完成していないと翻訳できない）	1 命令（プログラムが未完成でも翻訳／実行できる）
実行速度	速い	遅い
適応業務	バッチ処理	対話型処理

(4) ジェネレータ

ジェネレータは、入力データや出力データの様式、処理の条件などをパラメタとして与えることにより、目的プログラムを生成する言語プロセッサである。処理手順を意識しないで目的プログラムを生成できるため、**非手続型言語**と分類されることもある。

(5) トランスレータ

言語処理系の場合、**トランスレータ**は、プログラムを別のプログラム言語のプログラムに変換するソフトウェアである。コンパイラやアセンブラは、原始プログラムを目的プログラムに変換（翻訳）するトランスレータといえる（インタプリタは含まれない）。

3-2-2 サービスプログラム

言語プロセッサにより作成された目的プログラムは、そのままでは実行できない。目的プログラムを実行するためには、さまざまな**サービスプログラム（ユーティリティプログラム）**が必要となる。また、プログラムの実行には直接関係ないが、プログラムの作成や実行に補助的な役割を担うサービスプログラムもある。

ここでは、代表的なサービスプログラムについて説明する。

(1) リンカ（リンケーজেディタ，関係編集プログラム）

リンカ（リンケーজেディタ）は、複数の目的プログラム（目的モジュール）を一つの実行プログラム（**ロードモジュール**）に組み立てる（リンクする）サービスプログラムである。コンパイラにより機械語に翻訳される原始プログラムは、もともと一つのプログラムを分割したモジュール単位であることが多い。この場合、コンパイラは別のモジュール（外部関数など）を未解決アドレスとして処理する。この未解決アドレスを解決して、分割・翻訳した目的プログラムを統合するのがリンカの役割である。このとき、すでに**プログラムライブラリ（ロードライブラリ）**に登録されている部品（**ライブラリモジュール**）を取り出して組み込む機能もある。

なお、実行直前にOSによって**動的リンクライブラリ（DLL：Dynamic Link Library）**の部品（モジュール）を組み込む**動的リンク**と区別するために、リンカによるリンクを**静的リンク**と呼ぶこともある。

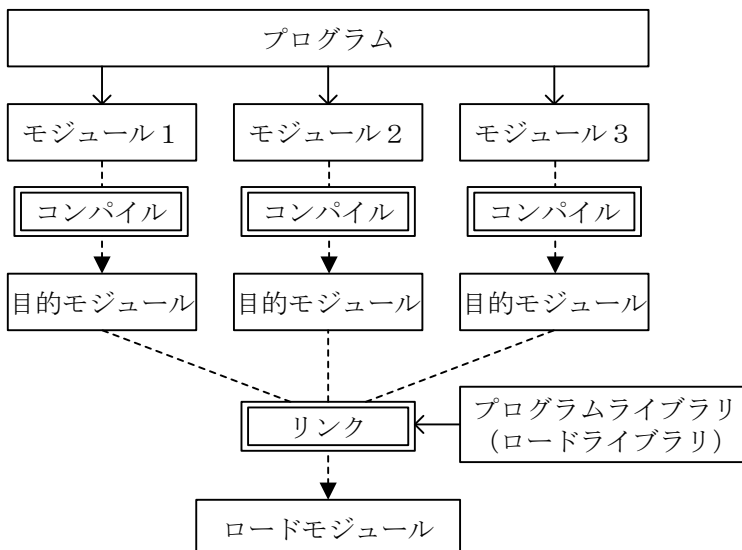


図 3-9 リンカ（リンケーজেディタ）の働き

(2) ローダ

ローダは、ロードモジュールを主記憶装置に格納して、実行させるプログラムである。また、目的モジュールを実行可能なロードモジュールにして主記憶装置に格納する、**関係ローダ**というプログラムもある。

(3) エディタ

エディタは、プログラムの入力や修正などに対して、補助的な役割を担うソフトウェアである。一般には、文字入力に使用する**テキストエディタ**のことを指すが、図形や絵などを入力する**図形エディタ**や、特定のプログラム言語の構造に合わせて、入力効率を高めた**構造エディタ**などもある。

(4) エミュレータ

エミュレータは、異なる OS などの環境を擬似的に作成して、他機種のプログラムを自機種で実行するマイクロプログラムである。

- ・ **インサーキットエミュレータ (ICE : In-Circuit Emulator)**

マイクロプログラムの開発などにおいて、対象マイクロプロセッサの機能をハードウェアにより模倣（エミュレート）するエミュレータである。開発用のテストツール（デバッグツール）として利用される。

- ・ **VM (Virtual Machine ; 仮想マシン)**

エミュレータによって、仮想的に構築されたコンピュータである。この考え方により、1 台のコンピュータで複数の OS を動作させることなどができる。

(5) シミュレータ

シミュレータは、自然科学などの複雑な事象を模擬的に実験するためのプログラムや仕組み（システム）のことである。言語処理系の場合は、プログラムの動作を再現（シミュレート）するプログラム開発用ツールとして利用されることもある。

(6) 整列／併合プログラム

整列プログラムは、データの整列（ソート）処理を行うプログラムである。一方、**併合プログラム**は、整列された複数のデータを一つにまとめる併合（マージ）処理を行うプログラムである。整列と併合は密接な関係にあるため、共通のサービスプログラム（整列／併合プログラム）として提供される場合が多い。

3-3 プログラムの属性

コンピュータで実行されるプログラムには、いくつかの性質がある。実行するプログラムの性質により運用の仕方が変わることもあるので、注意する必要がある。

(1) 再配置可能プログラム（リロケータブルプログラム）

再配置可能プログラム（リロケータブルプログラム）は、実行するプログラムをローダによって主記憶上に格納するとき、どの位置に格納しても実行可能なプログラムである。**リロケーションプログラム**とも呼ばれる。ロード位置に対応してアドレス情報を補正することなどにより、プログラムの実行を可能にしている。また、プログラムが実行中であっても、格納位置の変更が可能なものは、特に**動的再配置可能プログラム**という。

(2) 再使用可能プログラム（リユーザブルプログラム）

再使用可能プログラム（リユーザブルプログラム）は、何度でも繰り返し使用できるプログラムである。通常、プログラムは実行ごとに、ローダにより主記憶装置に格納し直す必要がある（これをリロードという）。再使用可能プログラムは、このリロードをすることなく、一度ロードした後は何度でも繰り返し使用できる。再使用可能プログラムでは、プログラムの実行開始時に変数の初期化などを行い、処理の整合性を保っている。そのため、同時に複数のタスクで使用することはできない。

(3) 再入可能プログラム（リエントラントプログラム）

再入可能プログラム（リエントラントプログラム）は、複数のタスクで同時に使用しても正しい結果が得られるプログラムである。再入可能プログラムでは、プログラムを変数部分と手続部分に分割し、タスクごとに変数部分を割り当て、手続部分のみ共有することで同時処理を実現している。なお、同時に複数のタスクで使用するためには、リロードしなくても繰り返し使用できる再使用可能プログラムの特性が必要である。

(4) 再帰プログラム（リカーシブプログラム）

再帰プログラム（リカーシブプログラム）は、プログラム実行中に自分自身を呼び出すことができるプログラムである。実行途中の状態を LIFO（Last-In First-Out；後入れ先出し）方式で記録することにより、実行後に一つ前の状態に戻ることができる。現在は、自分自身の呼出しも、別タスクからの呼出しとして制御するのが主流であるため、再帰プログラムは再入可能であるといえる。

4 ファイル

コンピュータで管理する情報の単位として、ファイルがある。OS のデータ管理機能とは、このファイルを管理する機能ともいえる。この章では、ファイルの編成方式やアクセス方式について学習する。

4-1 ファイルとレコード

コンピュータ内部での情報の単位についてまとめると、次のようになる。

- **ビット**

コンピュータ内部での最小記憶単位である。0 又は 1 で記録される。

- **バイト**

複数のビット（8 ビット）を集めたものである。一般的に、文字コードは 8 ビットで表現されることが多いため、一つの文字を表す単位である。

- **アイテム**

複数の文字で構成された、一つの意味をもつ単位である。

- **レコード**

複数の関連するアイテムを集めて、データとして処理する単位である。

- **ファイル**

同一形式のレコードを複数集めたもので、データを管理する単位である。

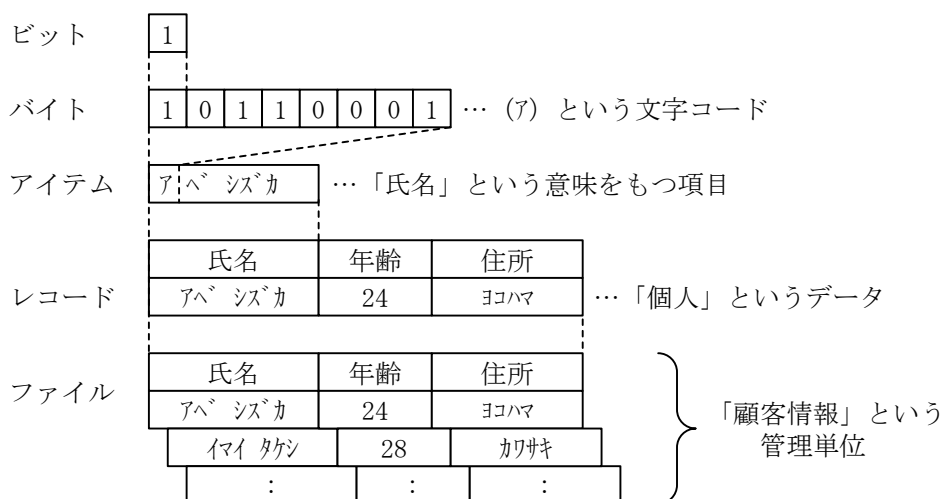


図 3-10 コンピュータ内部の情報単位

4-1-1 レコードの分類

(1) 物理レコードと論理レコード

コンピュータ内部では、データは意味のあるレコード単位で処理される。しかし、実際に補助記憶装置などでレコードを扱う場合、1レコード単位で入出力を行うと効率が悪い。ため、複数のレコードをまとめた**ブロック**という単位で取り扱う。この入出力の単位であるブロックを、物理的に取り扱うレコードという意味で**物理レコード**という。それに対して、処理単位となるレコードを**論理レコード**と呼んで区別している。

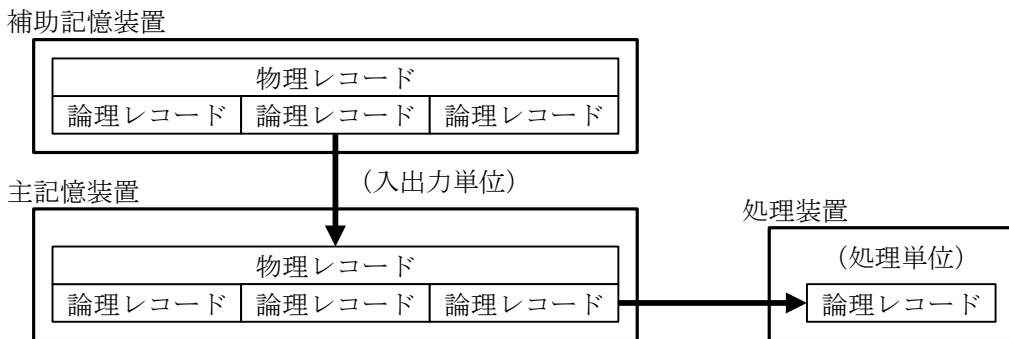


図 3-11 物理レコードと論理レコード

(2) レコードの記録形式

レコードを記録形式で分類すると、次のようになる。

- ・ **固定長レコード**

すべてのレコードの大きさ（バイト数）が、一定のレコード形式である。ブロック化が可能であり、取扱いも容易である。

- ・ **可変長レコード**

レコードの大きさが、それぞれ異なっているレコード形式である。レコードの大きさが異なっても、各レコードの先頭に大きさに関する情報が記録してあるため、ブロック化が可能である。

- ・ **不定長レコード**

レコードの大きさが、それぞれ異なっているレコード形式である。レコードの大きさが異なっている点では可変長レコードと同じであるが、各レコードの大きさに関する情報をもたないため、ブロック化ができない。

4-1-2 ファイルの分類

ファイルの分類には、使用目的による分類、保存（利用）期間による分類、利用者による分類などがある。なお、この分類は、後述するファイルのアクセス方式や編成方式とは別のものである。

(1) 使用目的による分類

- ・ **マスタファイル**（基本ファイル）
業務の中核となる情報（商品情報や顧客情報など）を記録するファイル
- ・ **トランザクションファイル**（変動ファイル）
マスタファイルに対する更新情報などを記録するファイル
- ・ **ヒストリカルファイル**
処理の履歴などを記録するファイル。ジャーナルファイルやログファイルともいい、主として障害対策に使用される。
- ・ **バックアップファイル**
重要なファイルの内容などをそのまま複写しておくファイル。主として、障害対策に使用される。
- ・ **ワークファイル**（作業ファイル）
処理を行う過程で一時的に作成される中間ファイル

(2) 保存（利用）期間による分類

- ・ **保存ファイル**
長期間にわたり使用され続けるファイルで、永久ファイルともいう。マスタファイルなどが該当する。
- ・ **一時ファイル**
一時的に、必要に応じて使用されるファイルで、原則として使用後に削除される。トランザクションファイルやワークファイルなどが該当する。

(3) 利用者による分類

- ・ **システムファイル**
OS を構成するプログラムなど、システムを運用／管理するためのファイル
- ・ **ユーザファイル**
利用者が使用するプログラムやデータを記録したファイル

4-2 ファイルのアクセス方式

ファイルの**アクセス方式**とは、ファイルに記録してあるレコードを読み込んだり、ファイルにレコードを書き込んだりする方式のことである。

(1) 順次アクセス（シーケンシャルアクセス）

順次アクセス（シーケンシャルアクセス）は、ファイルの先頭から順に、レコードにアクセスする方式である（単に順アクセスともいう）。

ファイルからレコードを読み込む場合、先頭のレコードから規則正しく順番にしか読み込むことができない。このため、全レコードを先頭から順番に処理する場合にはよいが、途中にある特定のレコードだけを処理する場合には向いていない。

ファイルにレコードを書き込む場合、すでに記録してあるレコードの後ろから順番に記録することになる（レコードが記録されていない場合は先頭から順番に記録する）ので、無駄な領域が発生しない。そのため記録効率は良いが、レコードを途中に挿入するためにはファイルを作り直す必要がある。

(2) 直接アクセス（ランダムアクセス）

直接アクセス（ランダムアクセス）は、ファイルに記録されているレコードの順番に関係なく、特定のレコードに直接アクセスする方式である。

特定のレコードを対象とした処理（読み書き）には向いているが、全レコードに対する処理には向いていない。また、特定のレコードにアクセスするためには、そのレコードの記録位置を知る必要があるため、アクセス速度は順次アクセスよりも遅くなる。

(3) 動的アクセス（ダイナミックアクセス）

動的アクセス（ダイナミックアクセス）は、直接アクセスと順次アクセスを組み合わせたアクセス方式である。

具体的に、氏名が'サ'から始まる人の一覧表を作成する場合を例に説明する。レコードの読み込み方法としては、最初に氏名が'サ'から始まるレコードに直接アクセスした後、連続するレコードを順次アクセスしていくことになる。この二つのアクセスを同時に可能にするのが、動的アクセスである。

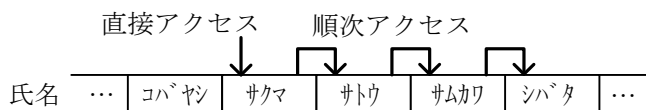


図 3-12 動的アクセス

4-3 ファイル編成方式

ファイル編成方式とは、ファイルにレコードを記録する順番や構造などを定めた方式のことである。ファイル編成方式は汎用機（メインフレーム）が主流の時代に考えられたので、現在の OS がすべてのファイル編成方式をサポートしているわけではない。

4-3-1 ファイル編成方式の種類と特徴

① 順編成ファイル

記憶媒体上に、連続してレコードを記録する順次アクセス専用の編成方式である。レコードが連続して記録されるため記憶媒体の使用効率は良いが、レコードを途中に追加することはできない。DASD（直接アクセス記憶装置）を含む、すべての記憶媒体で実現できるが、磁気テープは順編成ファイルしか実現できない。

② 相対編成ファイル

順編成ファイルのようにレコードを順番に記録し、各レコードに先頭レコードからの相対番号（相対アドレス）を付けて管理する編成方式である。記憶媒体の使用効率が良く、相対番号によるレコードの直接アクセスも可能である。

③ 索引順編成ファイル（索引編成ファイル）

データを記憶する**基本域**（プライム領域）と特定レコードを直接アクセスするための情報（索引）を記憶する**索引域**（インデックス領域）、および基本域から追い出されたレコードを格納する**あふれ域**（オーバフロー領域）で構成される編成方式である。記憶媒体の使用効率が良く、大規模なファイルを作ることができる。

④ 区分編成ファイル

ファイルを**メンバ**と呼ばれる複数の単位に分割して記録し、各メンバの情報などを**登録簿**（ディレクトリ）で管理する編成方式である。メンバは順編成ファイルをサブファイルに分割したもので、メンバ単位で更新管理が行える。プログラムライブラリとして使用されるのが一般的である。

⑤ 直接編成ファイル

アドレスを指定することで、特定のレコードにアクセスできる直接アクセス専用の編成方式である。レコードの記録位置が分散されるため、記憶媒体としての使用効率は良くない。磁気ディスク装置など、DASD（直接アクセス記憶装置）でのみ実現できる（直接編成ファイルで特定のレコードを指定するため、レコード内のキー項目を使用してアドレスを求める方法については、4-3-2 で説明する）。

⑥ VSAM 編成ファイル

仮想記憶方式を採用している OS で使用できる編成方式である。従来のファイル編成方式を仮想的に実現する編成方式ともいえる。

4-3-2 直接編成ファイルのアドレス指定方式

(1) 直接アドレス指定方式

直接アドレス指定方式は、レコードキーの値を、そのままアドレスとする方式である。アドレスを求めるための処理は必要ないが、レコードキーの値によってはアドレスにばらつきが多く、実用的とはいえない。なお、レコードキーの値とアドレスの対応表を使用する方法もあるが、レコード数分の表要素が必要となるため、用途は限定される。

(2) 間接アドレス指定方式

間接アドレス指定方式は、レコードキーの値を法則に従って変換し、アドレスを求める方式である。変換作業を**ハッシング**、使用される法則（関数）を**ハッシュ関数**という（そのため、間接アドレス指定方式の直接編成ファイルのことを、**ハッシュ編成ファイル**と呼ぶこともある）。代表的な方法には、レコードキーの値をある値で除算した余りをアドレスとする**除算法**があり、そのほかにも**重合せ法**や**基数変換法**などがある。

間接アドレス指定方式では、異なるレコードキーの値が同一のアドレスに変換される可能性がある（この現象を、**シノニム**が発生したという）。このとき、変換されたアドレスを利用できるレコードを**ホームレコード**、利用できないレコードを**シノニムレコード**という。

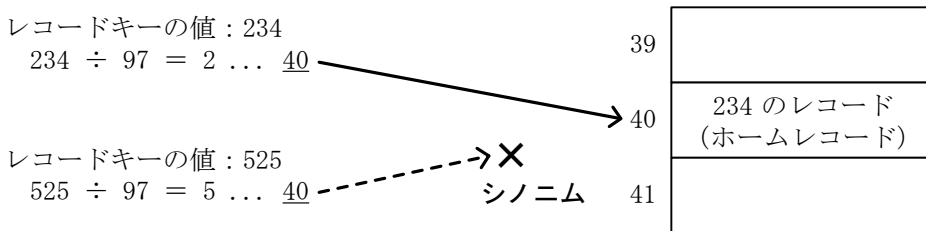


図 3-13 シノニムの発生（除算法で 97 を使用した場合）

シノニムが発生した場合の対処法には、求めたアドレスに近い空き領域にシノニムレコードを記録する**シーケンシャル法**や、別の記憶領域を確保して、そこにシノニムレコードを格納してポインタでつなぐ**チェーン法**がある。なお、どちらの方法でも、レコードキーの値から求めたアドレスでは目的のレコードにアクセスできなくなるので、アクセス効率は低下する（アクセス時間が長くなる）。したがって、レコードキーの値の分布に偏りがなく、レコードキーの値から求めたアドレス（ハッシュ値）が**一様分布**（事象の発生確率がほぼ等しい分布）に近似されることが望ましい。

4-4 小型コンピュータのファイル管理

ここまでの説明は、主として大型汎用コンピュータが扱うファイルを対象としたものである。ここからは、小型コンピュータ（PC など）の OS（Windows, UNIX）におけるファイル管理について説明する。

4-4-1 ファイルシステム

ファイルシステムとは、ハードディスク装置などの補助記憶媒体に記録されたファイルを管理する仕組みのことである。OS ごとに異なるファイルシステムが提供され、Windows 系の PC で使用される **FAT ファイルシステム**（FAT : File Allocation Tables）などが代表的である。FAT ファイルシステムでは、ハードディスクを**ボリューム**といい、ボリュームを**クラスタ**という単位で管理する。クラスタとは、ボリュームを細分化、管理する、又は磁気ディスクのセクタを複数まとめた管理単位である（ファイルの入出力はクラスタ単位で一括して行われるので、レコードやアイテムとして識別するのは各ソフトウェアの役割となる）。代表的なものとしては、クラスタ識別子として 32 ビットを使用し、ボリュームを 2^{32} 個のクラスタに細分化する **FAT32** がある。ただし、現在はハードディスク装置の記憶容量が大きくなっているため、ボリュームの最大容量が FAT32 よりも大きく設定された **NTFS**（NT File System）が多く利用されている。

Windows 系以外の OS で使用される代表的なファイルシステムは、次のとおりである。

ファイルシステム名	対応 OS	備考
UFS (Unix File System)	UNIX 系	パーティションに分割
HFS (Hierarchical File System)	Mac OS	iPod でも HFS+を利用

4-4-2 ディレクトリ管理

小型コンピュータの OS では、ファイルをつリー構造で階層的に管理する。特に、UNIX ではファイルシステム全体を単一のツリーとして管理している。このとき、複数ファイルを上位階層で一つにまとめるために使用されるのが**ディレクトリ**である。

階層構造を作成するには、まず最上位にディレクトリが必要となる（これを**ルートディレクトリ**という）。ルートディレクトリの下に複数のファイルが記録されるが、すべてのファイルを同列で扱うと管理が複雑になるので、関連のあるファイルをいくつかまとめたディレクトリをルートディレクトリの下に作成する（これを**サブディレクトリ**という）。サブディレクトリの下には、ファイルのほかにサブディレクトリを作成することもできる。このようにファイルを階層的に管理する場合の一例を図 3-14 の階層構造図で示す。

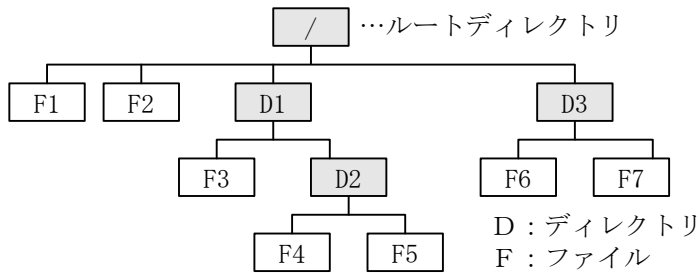


図 3-14 ファイルの階層構造図

階層構造では、利用者は現在使用しているディレクトリ（これを**カレントディレクトリ**という）の下しか見ることができない。図 3-14 の例でカレントディレクトリがルートディレクトリの場合、利用者には“F1, F2, D1, D3”しか見えない。したがって、サブディレクトリ以下のファイルを使用するには、カレントディレクトリを移動させなければいけないことになる（例えば、“F3”を使用したい場合は、カレントディレクトリを“D1”に移動させてから使用する）。

しかし、カレントディレクトリを毎回、移動させるのは大変なので、次の二つの方式で**パス**（経路）を指定してファイルを参照する（ファイルを参照するパスのことを、**参照情報**ということもある）。

- ・ **絶対パス**：ルートディレクトリからの経路を指定する。
- ・ **相対パス**：カレントディレクトリからの経路を指定する。

例. カレントディレクトリが“D2”のとき、“F3”を指定する絶対パスと相対パスを答えよ。なお、“..”は親ディレクトリを表し、“/”は、パスの先頭にある場合はルートディレクトリ、中間にある場合は区切りを表すものとする。

絶対パス：/D1/F3 （ルートディレクトリ→D1→F3）

相対パス：../F3 （カレントディレクトリ D2→親ディレクトリ D1→F3）

また、マルチユーザのコンピュータでは利用者ごとにファイルを管理する必要がある。そのためのディレクトリとして、**ホームディレクトリ**と**デスクトップディレクトリ**が使用される。OS を起動し、ログインしたときのカレントディレクトリをホームディレクトリといい、一般利用者がファイルの保存などに使う階層での最上位となる。利用者がログインすると、その利用者専用のデスクトップ画面が表示される。その管理情報を格納しているのがデスクトップディレクトリである。

4-4-3 ファイル共有

ファイル共有とは、一つのファイルを複数の利用者と共有できる仕組みである。ファイルサーバやNAS（Network Attached Storage）に記録されたファイルの**アクセス権**（ファイルを参照する権利）を設定することで、共有ファイルとして利用できるようになる。特に、NASは複数のプロトコルに対応しているファイルシステムがストレージに内蔵されているため、異なるOSや機種（サーバ）間でファイル共有が簡単に行える。なお、OSなどが提供する**ファイル検索機能**を利用すると、ファイル名の一部や最終更新日時などを指定するだけで、共有ファイルを含め、使用したいファイルが検索できる。

4-4-4 シンボリックリンク

シンボリックリンクは、UNIX系OSのファイルシステムでファイルやディレクトリに別の名前を与え、利用者やアプリケーションが別名をファイル本体と同様に扱えるようにする機能である。利用者やアプリケーションは、元のファイルを扱うのと同じ感覚で、シンボリックリンクを通してファイルやディレクトリにアクセスすることができる。同様の機能を、Windowsでは**ショートカット**、Mac OSでは**エイリアス**として提供している。

4-4-5 フォルダ

Windows系OSでは、ファイル操作もGUI環境で行っている。つまり、使用したいファイルをパスなどで指定するのではなく、アイコンを選択して指定するようになっている。

このとき、ディレクトリの代わりに**フォルダ**という名前が使用される。フォルダの中にもフォルダを作成できるため、利用者は目的のファイルが表示されるまで、フォルダを選択していく。つまり、フォルダを選択しながらウィンドウを切り替えていくことにより、カレントディレクトリを移動させていることになる。なお、フォルダも、ファイル共有と同じように共有することができる。

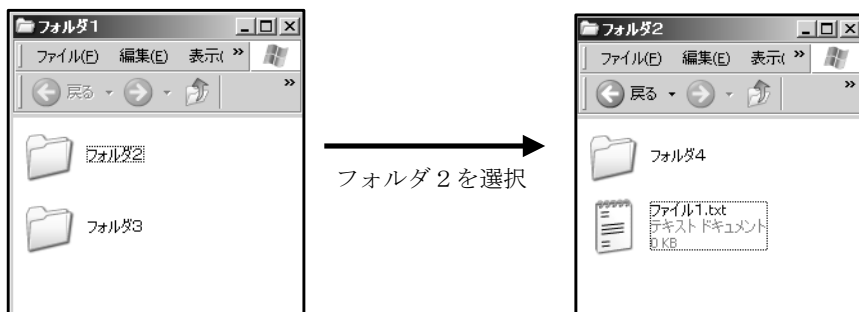


図 3-15 GUI 環境のファイル管理（フォルダ）

4-5 バックアップ

バックアップとは、万一の障害などに備えて、ファイルなどのデータを補助記憶装置に保存することである。ファイルに記録されているデータは非常に重要なものであるため、バックアップによって信頼性・可用性の向上に努める必要がある。バックアップ用の記録媒体としては、DDS (Digital Data Storage) などのストリーマを利用する場合が多い。

【バックアップの種類】

- ・ **フルバックアップ** (full backup)

すべてのデータを複写する。バックアップの作業時間は長くなるが、リストア（復元）は最新のフルバックアップデータだけで行える。

- ・ **差分バックアップ** (differential backup)

前回のフルバックアップ以降に更新されたデータを、すべて複写する。リストアには、最新のフルバックアップデータと最新の差分バックアップデータを使用するだけなので、増分バックアップより作業は少ない。

- ・ **増分バックアップ** (incremental backup)

前回のバックアップ（フルバックアップ及び増分バックアップ）以降に更新されたデータだけを複写する。バックアップの作業時間は差分バックアップより短いですが、リストアには、最新のフルバックアップデータとフルバックアップ以降のすべての増分バックアップデータを使用する。

【バックアップ運用上の注意点】

- ・ バックアップは、スケジュールを立てて定期的に実施する。特に複数の人が関係するバックアップは、業務時間外に実施できるように計画する。
- ・ 誤操作やバックアップのし忘れを防ぐために、バックアップ処理はできるだけ自動化する（手間もかからなくなる）。
- ・ 記憶容量に十分な余裕がある記録媒体を選ぶ。ただし、同じ HDD では障害時にバックアップファイルも読めないなので、別の記録媒体を使用する。
- ・ バックアップファイルの保管場所は、ファイルの重要性などを考慮して十分に検討する（元のファイルと同じ部屋に保管すると、火災などで同時に焼失する危険性がある。入退管理していない部屋に保管すると、重要な情報が第三者に持ち出されてしまう危険性がある）。
- ・ 災害、事故などからデータを保護するために、正副の二つをとる**多重バックアップ**を実施し、地理的に離れた別の場所に保管する。
- ・ バックアップファイルは、世代情報（版番号など）を付けて、できるだけ複数世代にわたって管理する。

第3部 演習問題

問1

次の文は、言語プロセッサ、サービスプログラム及び制御プログラムについて記述したものである。a, b, cの組合せとして正しいものはどれか。

- a ハードウェア資源の割振りなどにより、効率の良い動作環境を提供する。
- b プログラムの翻訳などを行う。
- c データ管理ユーティリティ、ディスク管理ユーティリティなど、一般的に必要とされるプログラムを提供する。

	a	b	c
ア	言語プロセッサ	サービスプログラム	制御プログラム
イ	サービスプログラム	言語プロセッサ	制御プログラム
ウ	制御プログラム	言語プロセッサ	サービスプログラム
エ	制御プログラム	サービスプログラム	言語プロセッサ

問2

UNIXにおいて、あるコマンドの標準出力を、別のコマンドの標準入力に直接、つなげる機能はどれか。

- ア パイプ
- イ バックグラウンドジョブ
- ウ ブレース展開
- エ リダイレクト

問3

オープンソースソフトウェアに関する記述のうち、適切なものはどれか。

- ア オープンソースソフトウェアはアプリケーションソフトウェアだけに存在する概念であり、オープンソースソフトウェアのOSは存在しない。
- イ オープンソースソフトウェアは改変が自由なソフトウェアであり、著作権は必ず放棄されていなければならない。
- ウ オープンソースソフトウェアは自由な再頒布ができるソフトウェアであり、再頒布可能な個人やグループを特定してはいけない。
- エ オープンソースソフトウェアは無償で配布されるソフトウェアであり、二次著作物であっても有償で配布してはいけない。

問4

スプーリング機能に関する記述として、適切なものはどれか。

- ア あるタスクを実行しているときに、入出力命令の実行によって CPU がアイドル状態になると、ほかのタスクに CPU を割り当てる。
- イ 細切れになった主記憶装置の領域を一つにまとめて、大きな使用可能領域として確保する。
- ウ 実行中のプログラムを一時中断して、制御プログラムに制御を移す。
- エ 主記憶装置と低速の入出力装置とのデータ転送を、補助記憶装置を介して行い、システム全体のスループットを高める。

問5

プロセスは、実行可能状態、実行状態、待ち状態を遷移しながら実行される。プロセスの状態遷移に関する記述のうち、適切なものはどれか。

- ア CPU 処理と入出力処理が交互に現れるプロセスを複数個同時に実行させると、各プロセスは実行状態と待ち状態の二つの状態間だけを遷移する。
- イ 実行可能状態とは、CPU の割当てを待っている状態をいう。実行可能状態のプロセスは一般に複数個存在し、これらは待ち行列を形成する。
- ウ マルチプログラミングシステムでは、CPU が1個でも、実行状態のプロセスは複数個存在する。
- エ ラウンドロビン方式による時分割処理を行っているシステムでは、実行状態のプロセスは、一定時間が経過すると待ち状態に遷移する。

問6

フラグメンテーションに関する記述のうち、適切なものはどれか。

- ア 可変区画方式では、いろいろな大きさのメモリ領域の獲得や返却を行ってもフラグメンテーションは発生しない。
- イ 固定区画方式は、可変区画方式よりもメモリ領域の獲得と返却を速く行えるが、フラグメンテーションが発生しやすい。
- ウ フラグメンテーションの発生によって、合計としては十分な空きメモリ領域があるのに、必要とするメモリ領域を獲得できなくなることがある。
- エ メモリ領域の獲得と返却の頻度が高いシステムでは、メモリ領域を返却するたびにメモリコンパクションを行う必要がある。

問7

仮想記憶におけるセグメント方式とページング方式に関する記述のうち、ページング方式の特徴はどれか。

- ア 仮想アドレス空間の管理単位である領域の大きさを、動的に変更できる。
- イ 実記憶領域の使用効率が高く、領域管理も容易である。
- ウ プログラムから見た論理的な単位でアクセスを行うことができる。
- エ プログラム実行中に別プログラムをプログラム単位で読み込むことができる。

問8

仮想記憶システムで使用されるページリプレースメントアルゴリズムには、FIFO方式やLRU方式がある。これらのページリプレースメントアルゴリズムの基本的な考え方として、適切なものはどれか。

- ア その時点以降に削除される可能性が最も高いページがどれかを推測する。
- イ その時点以降に削除される可能性が最も低いページがどれかを推測する。
- ウ その時点以降の最も近い将来に参照されるページがどれかを推測する。
- エ その時点以降の最も遠い将来まで参照されないページがどれかを推測する。

問9

Java Servletの説明として、適切なものはどれか。

- ア Javaで開発されたプログラムであり、Webサーバからクライアントにダウンロードして実行される。
- イ Javaで開発されたプログラムであり、クライアントの要求に応じてWebアプリケーションサーバ上で実行される。
- ウ Javaで開発されたプログラムをアプリケーションの部品として取り扱うための規約である。
- エ Javaで開発されたプログラムを実行するインタプリタ（実行環境）であり、バイトコードと呼ばれる中間コードを実行する機能をもつ。

問10

文書・章・節のタイトル、図・表などの文書構成要素、それらの間の関係などを定義するための標準文書記述言語はどれか。

- ア CSS イ HTML ウ SGML エ SOAP

問11

コンパイラにおける最適化の説明として、適切なものはどれか。

- ア 構文木をプログラム言語の意味的に解析して、中間コードを生成する。
イ コンパイルを実施するコンピュータとは異なるアーキテクチャをもったコンピュータで動作するオブジェクトコードを生成する。
ウ プログラム構造の変更などを行い、実行時の処理効率を高めたオブジェクトコードを生成する。
エ プログラムの実行時に、ソースコードや中間コードを翻訳してオブジェクトコードを生成する。

問12

動的リンクライブラリ（DLL）に関する記述として、適切なものはどれか。

- ア コンパイル時に、コンパイラによって組み込まれる。
イ コンパイルの前に、プリコンパイラによって生成される。
ウ 実行時に、OSによって関係される。
エ ロードモジュール作成時に、関係編集プログラムによって関係される。

問13

プログラムの属性に関する記述のうち、適切なものはどれか。

- ア 再帰的処理のためには、実行途中の状態を FIFO 方式で記録し、制御する必要がある。
イ 再使用可能なプログラムは、再入可能でもある。
ウ 再入可能プログラムを実現するためには、プログラムを変数部分と手続部分に分割して、変数部分をタスクごとにもつ必要がある。
エ 複数のタスクで同時に実行できるようにしたプログラムは、再帰的である。

問14

直接編成ファイルにおけるレコードキーの値を格納アドレスに変換したハッシュ値の分布として、理想的なものはどれか。

ア 一様分布

イ 幾何分布

ウ 二項分布

エ ポアソン分布

問15

ファイルシステムに関する記述のうち、適切なものはどれか。

ア 階層構造の最上位のディレクトリとして、最初にカレントディレクトリを作成する。

イ ディレクトリでファイルを管理するためには、FAT ファイルシステムを採用しなければいけない。

ウ ファイルシステム内の特定のファイルを指定するためには、ディレクトリを経由するパスを指定する。

エ ファイルを登録するディレクトリと下位のディレクトリを登録するディレクトリを別々に用意する必要がある。

問16

ファイルのバックアップに関する記述のうち、適切なものはどれか。

ア RAID0 のようにファイルのストライピングを行っているシステムでは、ファイルのバックアップは全く不要な作業である。

イ バックアップ作業を効率的に行うために、バックアップファイルは物理的に同一の装置に作成するようにする。

ウ バックアップは、ハードウェア障害や人為的なミスによって消失したファイルの復元に効果を期待できる。

エ ファイルの内容を磁気テープにバックアップする方式は安価であるが、自動化できないという欠点がある。

第4部

データベース

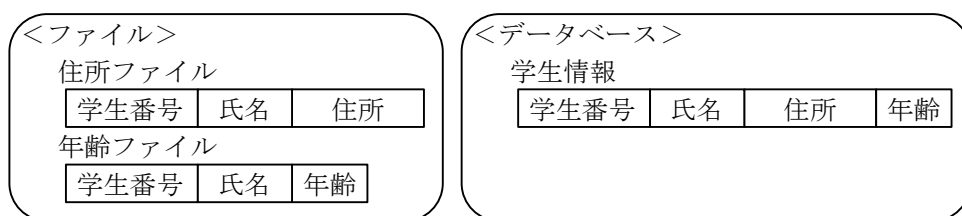
1 データベースの概要

コンピュータでデータを扱う方法として、ファイルを利用する方式のほかにデータベースを用いる方式がある。データベースとは、データを集中して一括管理する方式のことである。この章では、データベースの概念とその運用について学習する。

1-1 データベースとファイルの違い

ファイルはプログラムごとに作成されるため、データ項目の重複や、重複したデータ項目の内容がファイルごとに異なるデータの不整合という問題が生じる危険性がある。

そこで、データを一元管理して、データ項目の重複やデータの不整合が発生することを防止する**データベース**という考え方が利用されるようになった。



- ・ **データ項目の重複**：同じ学生の学生番号と氏名が、二つのファイル（住所ファイルと年齢ファイル）に重複して記録される。
- ・ **データの不整合**：住所ファイルの氏名を変更したとき、年齢ファイルの氏名を変更し忘れると、同じ学生番号の氏名が異なることになる。

図 4-1 ファイルとデータベース

データベースには、次のような機能が要求される。

- ・ **データの共有機能**
複数の利用者に同一データを提供できるようにする。
- ・ **データの独立機能**
データの修正がプログラムに影響を与えないようにする。
- ・ **データの保全機能**
データが常に正しい（矛盾が生じない）状態に保たれているようにする。
- ・ **データの障害対策機能**
データベースの障害に対して速やかに対応できるようにする。
- ・ **データの機密保護機能**
アクセス権の設定などにより、データの安全性を保証する。

1-2 データベースの設計

データベースを使用するためには、まず取り扱うデータを分析してデータベースを設計する必要がある。

1-2-1 データモデル

データモデルとは、現実世界のさまざまなデータの間を、コンピュータで扱うためにモデル化したものである。データモデルは、モデル化の視点によって分類される。

- ・ **概念データモデル**

特定のデータベースを意識せずに、データの間をモデル化したものである。どのようなデータを扱うのかを明確にする。

- ・ **論理データモデル（外部モデル）**

特定のデータベースを意識して、データの間を論理的にモデル化したものである。データベースにおけるデータ間の相互関係を明確にする。

- ・ **物理データモデル（内部モデル）**

特定のデータベース製品を意識して、データの間を物理的にモデル化したものである。データ型など、データベースの物理的な内部構造を明確にする。

このうち、特定のデータベースを意識した論理データモデルとして、次の三つのデータモデルが利用される。

- ・ **階層モデル**

データの間を、親と子が1対多の関係となる、階層構造で表したデータモデルである。**階層型データベース（HDB : Hierarchical DataBase）**として実装される。

- ・ **ネットワークモデル**

データの間を、親と子が多対多の関係となる、階層構造で表したデータモデルである。**網型データベース（NDB : Network DataBase）**として実装される。

- ・ **関係モデル（リレーショナルモデル）**

データの間を、2次元の表形式で表したデータモデルである。**関係データベース（RDB : Relational DataBase）**として実装される。

なお、階層型データベース（HDB）や網型データベース（NDB）のように、データの論理的な配置を利用者が意識するデータベースを**構造型データベース**と呼ぶ。

このテキストでは、関係モデルを実装した関係データベース（RDB）を中心に、この後の説明を進めていく。

1-2-2 関係モデル（リレーショナルモデル）

関係モデル（リレーショナルモデル）は、データの関係を2次元の表形式で表したデータモデルである。表全体を**関係（リレーション）**、表を構成する行を**組（タプル）**、列を**属性（アトリビュート、フィールド）**という。

関係	学生番号	氏 名	国語	数学	英語	組
	2001	阿部 雅之	80	90	60	
	2002	今井 美紀子	60	30	70	
	2003	上野 英二	70	100	90	
属性						

図 4-2 関係モデル（リレーショナルモデル）

図 4-2 において，“2001”，“阿部 雅之”，“80”，“90”，“60”のように、実際に記録されている値のことを**実現値**という。また、属性が取り得る実現値の集合を**定義域（ドメイン）**という（定義域には、属性のデータ型や制約条件が含まれる場合もある）。このことから，“関係は定義域の直積の部分集合”と定義される（**直積**：すべての組合せを得る集合演算，**部分集合**：ある集合に含まれる集合）。

定義域			定義域			直積			部分集合			
学生番号			国語			学生番号		国語		学生番号	国語	
2001		×	80		→	2001		80	→	2001		80
2002			60			2001		60		2002		60
						2002		80				
						2002		60				
										(関係)		

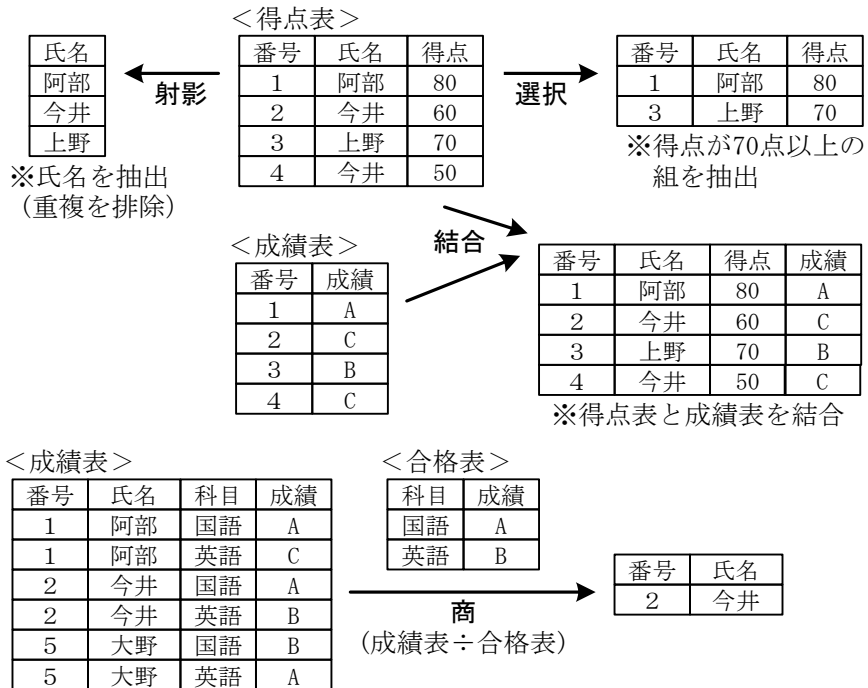
図 4-3 関係の定義

このように、関係モデルは集合とも深い関係がある。そのため、関係モデルを実装する関係データベースでは、関係データベース特有の**関係演算（関係代数）**と、集合の考え方に基づく**集合演算**を組み合わせで操作される。

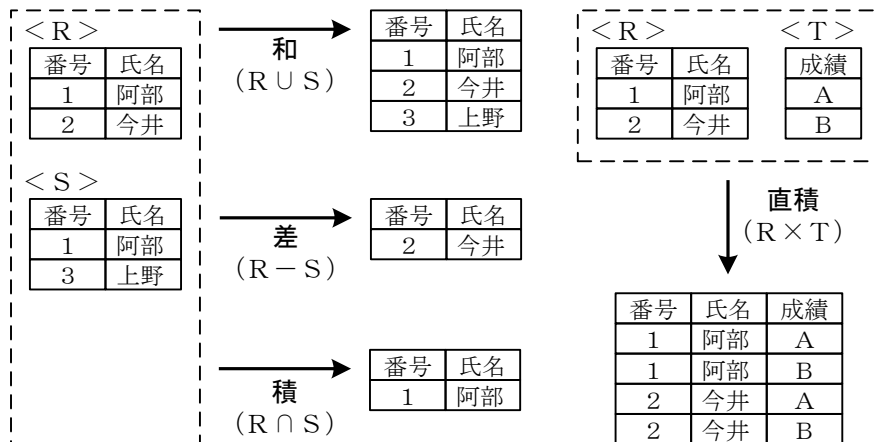
	演算名（操作名）	操作内容
関係演算	選択 (selection)	指定した条件が成立する組を抽出する。
	射影 (projection)	指定した属性を抽出する。
	結合 (join)	複数の表を一つにまとめる。
	商	別表のすべての組と等しい組を抽出する。
集合演算	和 (合併)	重複を除いて二つの表の組をすべて抽出する。
	差	別表に属する組を除いて抽出する。
	積 (共通)	二つの表に共通する組を抽出する。
	直積	二つの表の組をすべて組み合わせる。

【関係演算／集合演算の例】

・関係演算



・集合演算



このほかに、関係データベースの操作として、次のようなものがある。

- ・挿入：関係（表）に、新しいデータ（組）を追加（挿入）する操作
- ・更新：関係（表）に記録されているデータを変更（更新）する操作
- ・削除：関係（表）に記録されているデータ（組）を削除する操作

1-2-3 データベースの概念設計

データベースの概念設計とは、概念データモデルを作成する工程である。それには、最初に**データ分析**を行う必要がある。データ分析では、データベースを導入する対象業務に必要なデータを洗い出し、そのデータの意味と関連を分析・整理する。また、**命名規約**などを用いてデータ項目の標準化を行い、データの重複を排除する。データ項目名や意味・属性などの**メタデータ**は **DD/D** (Data Dictionary/Directory) にある利用者向けの**データディクショナリ**に登録し、異音同義語／同音異義語が発生しないようにする。

データ分析の結果を基に、概念データモデルを作成する。概念データモデルとしては、**E-R モデル** (E-R 図) や **UML** を利用する。

(1) E-R モデル (E-R 図)

E-R モデル (E-R 図) は、対象世界を**エンティティ** (Entity ; 実体) と**リレーションシップ** (Relationship ; 関連) の二つの概念で表すモデルである。

エンティティは、管理対象となる実体であり、長方形で表される。リレーションシップは、エンティティとエンティティの関連で、菱形で表される。さらに、エンティティやリレーションシップの特性・属性は、**アトリビュート**として楕円で表される。

また、E-R モデルでは、リレーションシップの**カーディナリティ** (多重度) を表すこともできる。カーディナリティには、“1 対 1”、“1 対多”、“多対多”の3種類がある。例えば、図 4-4 は、1 人の教師が複数の学生に対して講義を行い、1 人の学生が複数の教師の講義を受ける、“多対多”の関連を表す E-R モデルである。

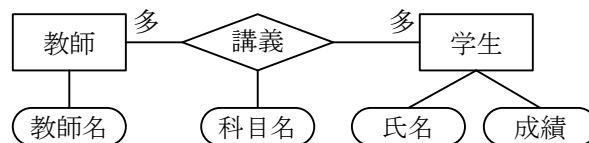


図 4-4 E-R モデル (E-R 図) の例①

なお、E-R モデルでカーディナリティを表記する際には、“多”を“*”や“M”又は“N”と表記する場合もある。また、図 4-5 のようにエンティティ同士を線で直接結ぶ表記法もある。この場合、線に矢印が付いているほうが“多”を意味する。図 4-5 は、問屋から複数種類の商品を仕入れ、商品は必ず同じ問屋から仕入れることを表している。

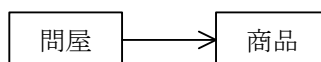


図 4-5 E-R モデル (E-R 図) の例②

(2) UML (Unified Modeling Language)

UML は、**OMG** (Object Management Group ; オブジェクト指向技術の標準化団体) によって認定された、標準的な統一モデリング言語である。オブジェクト指向開発の分析から設計、実装、テストまでの成果物（仕様書など）の表記に使用される、さまざまな図などが規定されている。

データベースの概念設計では、UML の**クラス図**を用いてデータモデルを表現する。クラス図は、管理対象となるクラス及びクラス間の関係を表現する図である。

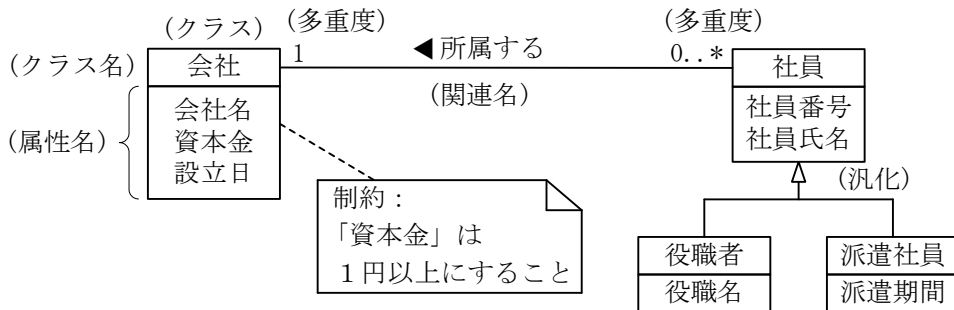


図 4-6 クラス図の例

- **クラス**は、三つの区画（クラス名、属性名、操作名を記述する区画）に分割した長方形で表される（属性名と操作名の区画は省略することができる）。図 4-6 には、クラス名が「会社」、「社員」、「役職者」、「派遣社員」という四つのクラスがある。
- **関連**は、二つのクラス間を実線でつないで表される。関連の意味を明確にするには、関連名を付ける。関連名で関連の主体と客体を明確にする場合は、関連名の先頭に、客体から主体に向けて黒塗りの三角形を付ける。図4-6 の場合、「会社」と「社員」には、“社員（客体）が会社（主体）に所属する”という関連がある。なお、関連には、関連クラス（二つのクラスを結ぶクラス）を破線でつなぐこともできる。
- **多重度**は、関連するクラス間の数的対応関係であり、“1”や“*”などで表される。例えば、“*”（0以上：一般的に“1対多”などの“多”），“0..1（0又は1）”，“0..*”（0以上），“1..*”（1以上）のように表す。図4-6 の場合、「会社」には0人以上の「社員」が所属し、「社員」は必ず一つの「会社」に所属するという多重度を表している。
- **汎化（特化）**は、上位クラス（スーパークラス）の性質を継承する性質であり、下位クラスから上位クラスに向けて、白抜きの三角形を付ける。図4-6の場合、「役職者」と「派遣社員」は、上位クラス「社員」の属性（社員番号、社員氏名）を継承する。
- **制約**は、クラスなどに付けられる条件であり、破線で結ばれたメモ（右上の角を折った長方形の紙）で表される。図4-6 の場合、「会社」には、属性“資本金”の値が1以上でなければいけないという制約が付けられている。

1-2-4 データベースの論理設計

データベースの論理設計とは、実装するデータベースを意識して論理データモデルを作成する工程である（構造型データベースはファイル／レコード／フィールド（項目）を、関係データベースはテーブル（表）／タプル／属性（アトリビュート）を明確にする）。

関係データベースの場合、概念設計で表したデータ項目を、どのテーブル（表）の属性（アトリビュート）として管理するかを決定する**テーブル設計**を行う。このとき、各テーブルの**主キー**及び**外部キー**についても検討しなければならない。

・主キー

テーブルのタプル（レコード）を一意に特定する項目である（複数の項目からなる**複合キー**でもよい）。主キーには、NULL（空値）を許可しない**非 NULL 制約**や、テーブル内での値の重複を許さない**一意性制約**などの制約が発生する。

・外部キー

別テーブルの主キーと比較してタプル（レコード）を参照するための項目である。外部キーには、参照するテーブルの主キーに同じ値をもつタプル（レコード）が存在しなければいけない**参照制約**を設定することができる。

※非 NULL 制約、一意性制約、参照制約は、データを常に正しい状態に保つための**整合性制約**と呼ばれる。

一般的なテーブル設計では、必要な項目をテーブルに列挙した**非正規形**に対し、**データの正規化**を行って無駄のないテーブルを設計する。なお、**関数従属性**とは、ある属性が決まれば他の属性が一意に決まる性質のことである。

【データの正規化手順】

- 1) **第一正規化**（成果物：第一正規形のテーブル）
繰返し項目や導出項目（計算により求められる項目）を排除する。
- 2) **第二正規化**（成果物：第二正規形のテーブル）
主キーが複合キーのとき、主キーを構成する一部の項目に従属（**部分関数従属**）している項目を別テーブルに分割する。
- 3) **第三正規化**（成果物：第三正規形のテーブル）
主キー以外の項目に従属（**推移関数従属**）している項目を別テーブルに分割する。

手順に従って第三正規化まで進めると、すべての項目が主キーに従属（**完全関数従属**）するテーブルを設計できる。ただし、第三正規形が最良でない場合もある（例えば、よく使う導出項目は残しておくほうが効率の良い場合もある）ので、運用面も含めて検討する必要がある。

例. 次の非正規形のテーブルを正規化して得られる第三正規形を答えよ。なお、下線は主キーであることを，{ }は繰返し項目であることを表している。

伝票（伝票番号，日付，顧客番号，顧客名，合計金額，
{商品番号，商品名，単価，数量，金額}）

- ① 第一正規化：繰返し項目（商品番号，商品名，単価，数量，金額），
導出項目（合計金額，金額）の排除

伝票（伝票番号，日付，顧客番号，顧客名，
商品番号，商品名，単価，数量）

- ② 第二正規化：部分関数従属（伝票番号に従属する {日付，顧客番号，顧客名}，商品番号に従属する {商品名，単価}）の分割

伝票（伝票番号，日付，顧客番号，顧客名）
明細（伝票番号，商品番号，数量）
商品（商品番号，商品名，単価）

- ③ 第三正規化：推移関数従属（顧客番号に従属する顧客名）の分割

伝票（伝票番号，日付，顧客番号）
明細（伝票番号，商品番号，数量）
商品（商品番号，商品名，単価）
顧客（顧客番号，顧客名）

1-2-5 データベースの物理設計

データベースの物理設計とは，実装するデータベース製品（データベースソフトなど）を意識して物理データモデルを作成する工程である。各属性に最適となるデータ型をデータベース製品の使用可能なデータ型から選択したり，磁気ディスク上での記憶形式や配置（マッピング）などを検討する。また，予測されるデータ量から，必要となるディスク容量を見積もったり，処理効率やアクセス効率などの性能評価も行う。このとき，アクセス効率を向上させるための**インデックス**（検索キー）を設定することについても検討する。データベースのサイズが大きい場合，検索によく利用される属性に対してインデックスを設定しておくことで検索効率を大幅に高めることができる。ただし，インデックスを記録するためのディスク容量が必要になり，インデックスの属性に対する更新処理などは，通常の属性に対する処理よりも時間がかかるようになるので十分に検討する。インデックスにはB⁺木インデックス，ハッシュインデックス，ビットマップインデックスなどがある。

1 - 3 データベース管理システム (DBMS)

データベース管理システム (DBMS : DataBase Management System) とは、データベースを効果的に活用するために、データベースを管理するソフトウェアである。なお、DBMS などを利用してデータベースを管理する人をデータベース管理者 (DBA) という。

1-3-1 データベース定義機能

データベース定義機能は、**スキーマ** (データベースの論理構造, 格納構造, 物理構造に関する定義と記述) を定義する機能である。スキーマの定義は, 次の **3層スキーマ** を利用して行われることが多い。

スキーマ名	意味	使用言語
概念スキーマ (スキーマ)	データベース全体の論理的な構造や名称を定義する。	DDL (データ定義言語)
外部スキーマ (副スキーマ)	利用者から見たデータベース (データベースの必要部分だけ) を定義する。	DDL (データ定義言語) DML (データ操作言語)
内部スキーマ (記憶スキーマ)	データベースの物理的な構造 (記憶領域や編成方式) を定義する。	DSDL (データ記憶／記述言語)

1-3-2 データベース操作機能

データベース操作機能は, データの定義や操作に使用するデータベース言語 (関係データベースの場合は **SQL**) の使用環境を提供する機能である。また, 関係データベース管理システム (RDBMS) には, **SQL** を実行する際に効率が良くと考えられるアクセス経路を選択する **オプティマイザ** 機能などもある。

【データベース操作言語 (SQL など) の実行方法】

・親言語方式

高水準言語を使用して **SQL 文** を実行する方式である。プログラム中に **SQL 文** を記述する **埋込型 SQL** や, **SQL 文** を外部モジュールとして作成してプログラムから呼び出す **モジュール言語方式** などがある。

・独立言語方式

SQL 文 を単独で実行する方式である。コマンドラインから **SQL 文** を入力して実行する **会話型 SQL** や, 登録済みの **SQL 文** をパラメタ付きのコマンドで呼び出して実行する **コマンド方式** などがある。また, 既定のフォームから処理を選択したり, 条件を入力したりするだけで, **クエリ** (データベースを操作する命令で, **SQL** に変換される) として実行されるものもある。

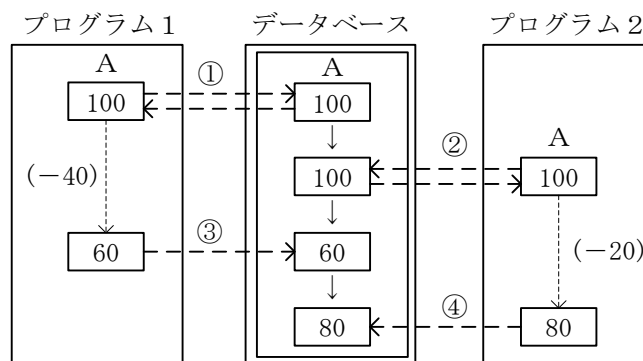
1-3-3 データベース制御機能

データベース制御機能は、データベースに記録されたデータの信頼性・安全性を向上させるために提供される機能である。

(1) 保全機能

保全機能とは、データの完全性（データインテグリティ）を保つ機能である。データの完全性を損なう（不整合が発生する）代表的な例として、次の**二重更新**がある。

【二重更新の例】



- ① プログラム 1 が、データベース中のデータ A (100) を参照する。
- ② プログラム 2 が、データベース中のデータ A (100) を参照する。
- ③ プログラム 1 で変更された内容 (60) で、データ A を更新する。
- ④ プログラム 2 で変更された内容 (80) で、データ A を更新する。
→更新内容が上書きされ、プログラム 1 の更新が無効になる。

このような問題を解決するために考えられたのが、**排他制御**である。排他制御にはいくつかの方法があるが、代表的なものとして**ロック方式**がある。

ロック方式は、参照されたデータベースにロック（鍵）をかけることにより、処理が終了するまで参照を制限する方式である。この方式を使用すると、更新処理中のデータを他のプログラムが参照する危険性がなくなるため、データの完全性は高められる。

ロック方式では、次の 2 種類のロックを使い分ける。

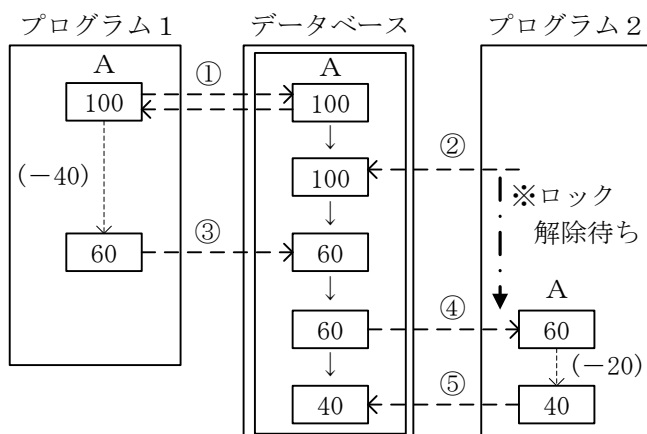
・共有ロック

主として、データの読取り時に使用するロック。ロックをかけた利用者以外に対しても読込み（共有ロック）だけは許可する。

・専有ロック（占有ロック）

主として、データの変更時に使用するロック。ロックをかけた利用者以外に対して、読込みも書込みも許可しない。

【排他制御（ロック方式）の例】



- ① プログラム 1 が、データベース中のデータ A（100）を参照する。
→データ A に専有ロックをかける。
- ② プログラム 2 が、データベース中のデータ A を参照しようとするが、ロックされているため、解除（アンロック）されるまで待ち状態になる。
- ③ プログラム 1 で変更された内容（60）で、データ A を更新する。
→データ A のロックを解除（アンロック）する。
- ④ ロック解除を確認後、プログラム 2 がデータ A（60）を参照する。
→データ A に専有ロックをかける。
- ⑤ プログラム 2 で変更された内容（40）で、データ A を更新する。
→データ A のロックを解除（アンロック）する。

ただし、ロック方式には**デッドロック**という問題が発生する可能性もある。デッドロックとは、ロックによって複数のプログラムが同時に待ち状態となり、実行が完全に停止してしまう現象である。デッドロックを完全に回避するのは難しいが、ロックをかける単位（粒度）を小さくすることで、デッドロックが起きる可能性を低くすることはできる。

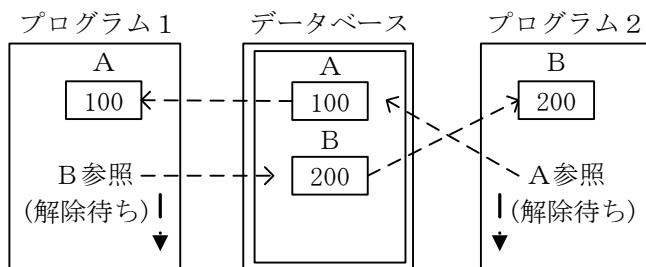


図 4-7 デッドロックのイメージ

また、排他制御のその他の方法として、データベースを資源としてセマフォで管理する**セマフォ方式**などがある。

(2) 機密保護機能

機密保護機能とは、データの安全性（データセキュリティ）を保つ機能である。データベースには、重要なデータや機密性の高いデータが格納される。そこで、データの機密保護が必要となる。DBMS が提供する機密保護機能には、次のようなものがある。

・暗号化

データベースに記録してある内容を暗号化することで、第三者にデータベースの情報が漏えいしても、意味がわからないようにするための方法である。正当な利用者にはのみ暗号解読用のソフトウェアを配付することで、比較的簡単に機密保護機能を高められる。しかし、“復号は絶対に不可能なことではない”，“暗号化／復号ソフトが漏えいするおそれがある”などの問題も考えられるため、別の手段と併用されることが多い。

・アクセス権の設定

該当データベースに対して、どのような処理（アクセス）ができるかを、あらかじめ設定する方法である。“利用者Aはデータを参照できるが更新はできない”，“利用者Bはデータの参照／更新はできるが削除はできない”というように、利用者ごとに細かく設定できる。データベースに対する各種権限と、一般的に設定できるアクセス権の関係をまとめると、次のようになる。

	読取権限	挿入権限	削除権限	更新権限
接続する権限	○	○	○	○
検索する権限	○	○	○	○
新規登録する権限		○		
削除する権限			○	
更新する権限				○

・パスワードの設定

データベースの利用者に対して、利用する権利をもっているかどうかをユーザ ID とパスワードによって確認する方式である。外部からの不正使用を防ぐために効果的である。また、アクセス権を設定する場合も、ユーザ ID を不正使用されないようにするため、パスワードの設定は必須となる。

・ジャーナルファイル（ログファイル）への記録

データベースの利用状況（日時、アカウント、利用内容）などを記録しておく、不正使用がなかったかを確認する方式である。即時的な機密保護はできないが、後日、不正利用者を特定できるため抑止効果はある。補助的な意味で利用されることが多い。

(3) 障害回復機能

障害回復機能は、障害が発生したとき、速やかに回復するための機能である。障害回復には、ある時点におけるデータベースをそのまま複製した**バックアップファイル**、データベースに対して行われる更新処理を記録した**ジャーナルファイル**（ログファイル）などが利用される。ジャーナルレコードはメモリに作成され、データベースに対するトランザクション単位の更新処理終了時に行われる**コミット**（更新処理を確定する処理）のタイミングで、ファイルに書き出される。そのため、一定時間ごと又は一定処理量ごとに**チェックポイント**を設定してログを整理し、メモリにある更新中のデータベースを**チェックポイントファイル**に記録する。これにより、障害発生時にはチェックポイント時点から復旧することができる。

障害回復のために行われるリカバリ処理には、次の2種類がある。

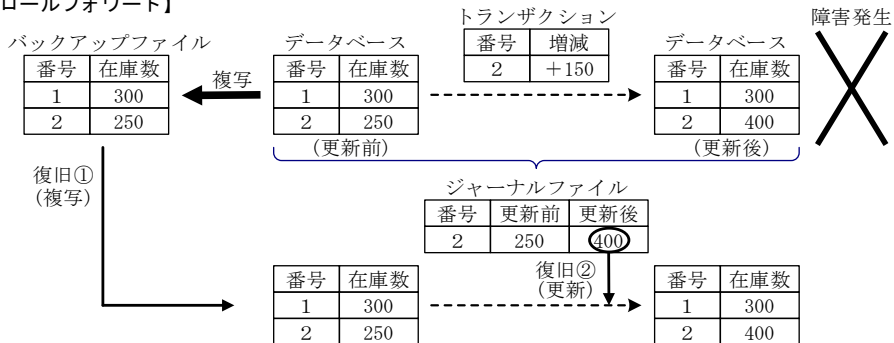
- ・ **ロールフォワード**（フォワードリカバリ、前進復帰）

主に、記憶媒体などの物理的な障害に対する回復手法である。ジャーナルファイルの更新後情報によりバックアップファイル又はチェックポイントファイルの内容を更新し、障害発生時点のデータベースとして復旧する。

- ・ **ロールバック**（バックワードリカバリ、後退復帰）

主に、トランザクションエラーなどの論理的な障害に対する回復手法である。無効処理後のデータベースを、ジャーナルファイルの更新前情報により無効処理前の状態に復旧する。なお、ロールバックは、コミットされていない更新処理を取り消すという意味で利用されることもある。

【ロールフォワード】



【ロールバック】

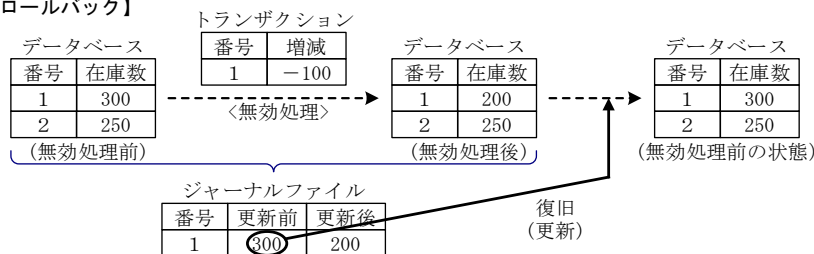


図 4-8 ロールフォワードとロールバック

リカバリ処理は、トランザクション単位でも実行される。この場合、チェックポイント後にコミットされたトランザクションは、ロールフォワードによってコミット完了後の状態に復旧する。一方、障害発生時点で実行中のトランザクションに対しては、コミットされていない処理を取り消し（ロールバック）、処理開始前の状態に戻してからトランザクションを再実行する。

図4-9の場合、チェックポイント後にコミットされたT1、T2は、チェックポイント時点からロールフォワードで回復する。一方、障害発生時点で更新処理中のT3は、チェックポイント時点からロールバックで処理開始前の状態に戻して再実行する。

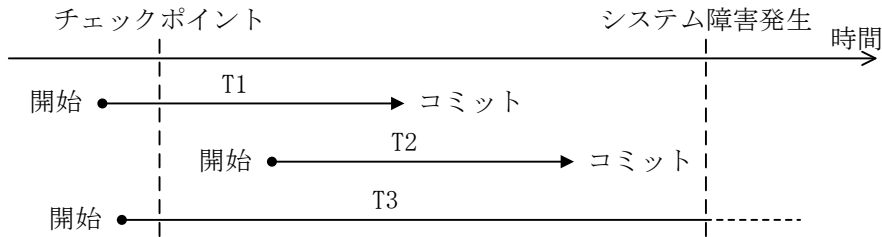


図4-9 トランザクション単位の障害回復

また、データベースを再始動する方法には、次の2種類がある。

- ・ウォームスタート方式

電源を切らないソフトウェアリセットなどによって、メモリなどの状態を保持したままで再始動する方式である。データベースの初期化などを行うことはなく、チェックポイントファイルやジャーナルファイルを利用して再始動直前の状態に復元する。

- ・コールドスタート方式（初期プログラムロード）

電源を切るハードウェアリセットなどによって、メモリなどの状態を完全にクリアしてから再始動する方式である。再始動後は直近のバックアップファイルとジャーナルファイルにより、データベースを再始動直前の状態に復元する。

なお、データベースに対する更新（追加、削除）を繰り返した結果、再利用されない記録領域が発生してアクセス効率が低下する場合がある。このようなとき、データベースを最適化する**再編成**も、障害回復機能の一つといえる。

【データベースのトランザクション管理】

データベースに対するトランザクション処理にも、**ACID 特性**は求められる。

- ・Atomicity（**原子性**）：コミット又はロールバックによって終了する。
- ・Consistency（**一貫性**）：処理の前後で矛盾（データの不整合）を生じない。
- ・Isolation（**分離性**）：複数のトランザクションは相互干渉しない。
- ・Durability（**持続性**）：コミットされた結果は、処理後も保持され続ける。

2 SQL

SQL (Structured Query Language) は、関係データベースを利用するためのデータベース言語である。SQL は、データを定義する DDL (Data Definition Language ; データ定義言語) とデータを操作する DML (Data Manipulation Language ; データ操作言語) などで構成される。

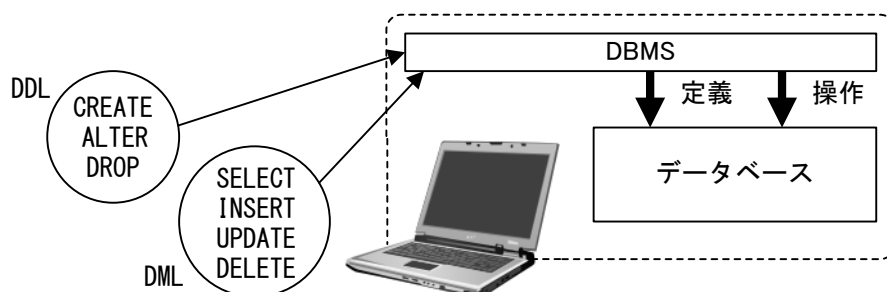


図 4-10 DDL (データ定義言語) と DML (データ操作言語)

この章では、次に示す三つのテーブル (表) を用いて、SQL の基本的な使用方法を説明する。

<学生表>

学生番号	氏名	性別
6724	山本 一希	男
6725	本山 十吾	男
6816	山田 百音	女
6817	山本 千代	女

英数字 かな漢字 かな漢字

<科目表>

科目番号	科目名
K11	英語 I
K12	英語 II
K21	数学

英数字 かな漢字

<得点表>

学生番号	科目番号	得点	受験日
6724	K11	65	20XX/10/20
6724	K21	85	20XX/10/21
6725	K21	60	20XX/10/21
6817	K11	85	20XX/10/20
6817	K12	90	20XX/10/20
6817	K21	95	20XX/10/21

英数字 英数字 整数 日付

※各テーブルの下線は、主キーを表す。

2-1 データ定義

データ定義とは、データベース、テーブル（実表）、ビュー（仮想表）などを定義することである。

2-1-1 データベースの定義

データベースの定義には、CREATE DATABASE 文を使用する。

```
CREATE DATABASE データベース名
```

ex01. データベース“成績管理 DB”を定義する。

```
CREATE DATABASE 成績管理 DB
```

また、定義したデータベースの構成（名称など）を変更する場合は ALTER DATABASE 文、削除する場合は DROP DATABASE 文を使用する。

2-1-2 テーブルの定義

テーブルの定義には、CREATE TABLE 文を使用する。

```
CREATE TABLE テーブル名
( 列名1 データ型1, 列名2 データ型2, ... )
```

- ・定義できる代表的なデータ型は、次のとおりである。

データ型	定義名	意味
文字型	CHAR	指定された長さの文字（1バイト）
	NCHAR	指定された長さの文字（2バイト）
数値型	INT	処理系で定義された精度をもつ整数
	DEC	指定された桁数の整数部と小数部をもつ数値
日付型	DATE	西暦年/月/日

- ・ **PRIMARY KEY** : 主キーの宣言（**主キー制約**：非 NULL 制約＋一意性制約）
- ・ **FOREIGN KEY** : 外部キーの宣言（**参照制約**）
- ・ **NOT NULL** : **非 NULL 制約**（実現値として NULL を許可しない）
- ・ **UNIQUE** : **一意性制約**（テーブル内での実現値の重複を許可しない）
- ・ **CHECK** : **検査制約**（実現値の条件を指定する）

※列定義と同時に行うと**列制約**，最後にまとめて行うと**表制約**と呼ばれる。

※**CREATE ASSERTION**を使用すると，テーブルへの制約（**表明**）も定義できる。

ex02. テーブル“得点表”を定義する。

```
CREATE TABLE 得点表
( 学生番号 CHAR(4),
  科目番号 CHAR(3),
  得点      INT    CHECK (得点 >= 0),
  受験日   DATE   NOT NULL,
  PRIMARY KEY (学生番号, 科目番号),
  FOREIGN KEY (学生番号) REFERENCES 学生表(学生番号),
  FOREIGN KEY (科目番号) REFERENCES 科目表(科目番号)
)
```

※得点には0以上という検査制約（列制約）を設定

※受験日には非 NULL 制約（列制約）を設定

※(学生番号, 科目番号)には主キー制約（表制約）を設定

※学生番号には、外部キーとして学生表の学生番号を参照するという参照制約（表制約）を設定

※科目番号には、外部キーとして科目表の科目番号を参照するという参照制約（表制約）を設定

また、定義したテーブルの構成を変更（再定義）する場合は ALTER TABLE 文、削除する場合は DROP TABLE 文を使用する。ALTER TABLE 文では、再定義の内容が属性の追加であれば ADD で、データ型の変更であれば MODIFY で指定する。

2-1-3 ビューの定義

ビューとは、実在するテーブルから仮想的に設定するテーブルのことで、外部スキーマに該当する（ビューを**仮想表**又は**導出表**、テーブルを**実表**又は**基底表**と呼ぶこともある）。ビューは、単一テーブルの属性や、複数テーブルから作成した新しい属性を記録できる。利用者は定義されたビューを通常のテーブルと同じように操作できるので、利用者ごとにデータベースの利用範囲を限定してデータの機密保護／保全に役立てることができる。

ビューの定義には、CREATE VIEW 文を使用する。

```
CREATE VIEW ビュー名 AS SELECT ～
```

※下線部には、データを抽出する SELECT 文（詳細は 2-2 で説明）を記述する。

ex03. “学生表”の学生番号と氏名を取り出して、ビュー“氏名表”を定義する。

```
CREATE VIEW 氏名表 AS SELECT 学生番号, 氏名 FROM 学生表
```

※下線部は、学生表から学生番号と氏名を抽出する SELECT 文

ビューは、あくまでも仮想テーブルの定義であり、新たにテーブルを作成しているわけではない（図 4-11 の赤紫色の部分を見えないようにしたと考えればよい）。したがって、ビューを更新すると、元のテーブルにも更新結果は反映される。そのため、タプルが元のテーブルと 1 対 1 に対応しないビュー（後に説明する集合関数を利用したビューなど）に対しては更新できないという制約もある。

学生番号	氏名	性別
6724	山本 一希	男
6725	本山 十吾	男
6816	山田 百音	女
6817	山本 千代	女

図 4-11 ビューの概念

2-1-4 アクセス権の定義

アクセス権とは、利用者ごとに割り当てる、データベースを利用する権利（**処理権限**）のことである。アクセス権の設定は、データの機密保護などに効果的である。

アクセス権の定義には、GRANT 文を使用する。

GRANT 権限名 1, 権限名 2, … **ON** テーブル名 **TO** 識別子(ロール名)

- ・定義できるアクセス権は、次のとおりである。

権限名	意味
SELECT	データベースを参照する権限
INSERT	データベースにデータを追加（挿入）する権限
DELETE	データベースのデータを削除する権限
UPDATE	データベースのデータを更新する権限
REFERENCES	データベースを再定義する権限
ALL PRIVILEGES	データベースに関するすべての権限

- ・識別子(ロール名)は、権限が与えられる対象者(役割)を一意に示すものである。

ex04. “学生表”に関する参照と更新の権限を“講師”（ロール名）に設定する。

GRANT SELECT, UPDATE ON 学生表 TO 講師

なお、定義（付与）した権限を取り消すには、REVOKE 文を使用する。

REVOKE 権限名 1, 権限名 2, … **ON** テーブル名 **FROM** 識別子(ロール名)

2-1-5 データの格納

データの格納は、テーブルにデータを挿入する処理である（本来、データ操作に含まれるものであるが、データベースを利用する準備としてデータ定義の中で説明する）。データをテーブルに格納するには、次の二つの方法がある。一般的に、少量のデータなら①の方法でよいが、大量のデータの場合には②の方法を利用するほうが効率的である。

① タプル（レコード）単位で格納する

タプル（レコード）単位でのデータ格納には、INSERT 文を使用する。

```
INSERT INTO テーブル名 VALUES (データ 1, データ 2, ...)
```

ex05. “科目表” にタプル (K11, 英語 I) を格納する。

```
INSERT INTO 科目表 VALUES ('K11', '英語 I')
```

② データ格納用のプログラムを利用する

親言語方式であらかじめデータ格納用のプログラムを準備したり、ユーティリティプログラムのデータ格納用プログラムを利用する。

[NoSQL]

NoSQL (Not only SQL) は、データ構造が単純で、膨大な量のデータを瞬時に検索する機能をもつデータベース管理システムである。SQL を利用する RDBMS の高度な機能を排除し、シンプルで柔軟な機能を提供する。

- ・ **キーバリュ型**（キーバリュストア）

任意の保存したいデータ (value) と、そのデータを一意に識別できる値 (key) を組みとして管理する。指定したキーに対応するデータを高速に検索できるが、あいまい検索はできない。

- ・ **ドキュメント型**（ドキュメントストア）／ドキュメント指向型

1 件分のデータ (XML 形式など) をドキュメントとして管理する。個々のドキュメントのデータ構造は自由であり、データを追加する都度変えることができる。

- ・ **グラフ型**／グラフ指向型

ノード、リレーションシップ、プロパティの 3 要素によって、ノード間の関係性を表現する。グラフ理論に基づき、要素間での任意の数の関係性を表すことができる。

- ・ **カラム型**（カラムストア）／列指向型

データをカラム（列）単位で管理する。列単位の処理（集計など）に適している。

2-2 データ操作

データ操作でもっとも多く使用されるのが、データベースからのデータ参照（抽出）である。SQL では、データの参照に SELECT 文を使用する。ここでは、SELECT 文の使用方法を中心に説明する。

2-2-1 条件指定のない参照

条件指定のない参照は、指定した属性を抽出する関係演算“射影”に該当する。条件指定のない参照では、次の SELECT 文を使用する。

```
SELECT 列名 1, 列名 2, ... FROM テーブル名
```

- ・ ***** : 列名として指定すると、すべての列が抽出される。
- ・ **DISTINCT** : 列名に指定すると重複が排除される（同じ結果が表示されない）。

ex06. “学生表” から、すべての列を抽出する。

```
SELECT * FROM 学生表
```

<ex06 の実行結果>

学生番号	氏名	性別
6724	山本 一希	男
6725	本山 十吾	男
6816	山田 百音	女
6817	山本 千代	女

ex07. “学生表” から、性別を抽出する（射影）。

```
SELECT 性別 FROM 学生表
```

ex08. “学生表” から、重複を排除して性別を抽出する（射影）。

```
SELECT DISTINCT 性別 FROM 学生表
```

<ex07 の実行結果>

性別
男
男
女
女

<ex08 の実行結果>

性別
男
女

2-2-2 条件指定のある参照

条件指定のある参照は、指定した条件が成立する組を抽出する関係演算“選択”に該当する。条件指定のある参照では、次の SELECT 文を使用する。

SELECT 列名 1, 列名 2, … **FROM** テーブル名 **WHERE** 抽出条件

- ・ WHERE 句に記述した条件を満たすタプルが抽出される。
- ・ 抽出条件に使用する代表的な演算子や表記は、次のとおりである。

<比較演算子>

表記	使用例	意味
=	A = B	AとBは等しい
<>	A <> B	AとBは等しくない (A ≠ B)
<	A < B	AはBより小さい
<=	A <= B	AはB以下 (A ≤ B)
>	A > B	AはBより大きい
>=	A >= B	AはB以上 (A ≥ B)

<論理演算子>

表記	意味
AND	二つの条件が両方とも成立すれば真
OR	二つの条件のどちらかが成立すれば真
NOT	元の条件が真なら偽、偽なら真 (否定)

<その他 (特殊な条件指定)>

表記	意味
BETWEEN ~ AND ~	値の範囲指定 (○以上○以下)
LIKE ~	文字列のパターン指定 (ワイルドカード指定) (% : 0文字以上の文字, _ : 1文字)
IS NULL	NULL の判定
IN (~)	()内のいずれかと一致

ex09. “学生表” から、性別が ‘女’ のタプルを抽出する (選択)。

SELECT * FROM 学生表 **WHERE** 性別 = '女'

ex10. “科目表” から、科目名が ‘数学’ 以外の科目名を抽出する (選択, 射影)。

SELECT 科目名 **FROM** 科目表 **WHERE** 科目名 <> '数学'

<ex09の実行結果>

学生番号	氏名	性別
6816	山田 百音	女
6817	山本 千代	女

<ex10の実行結果>

科目名
英語 I
英語 II

ex11. “得点表” から、科目番号に '2' が含まれていて、得点が 80 以上 90 以下の学生番号、科目番号、得点を抽出する。

```
SELECT 学生番号, 科目番号, 得点 FROM 得点表
WHERE (科目番号 LIKE '%2%')
      AND (得点 BETWEEN 80 AND 90)
```

ex12. “得点表” から、科目番号に '2' が含まれているか、得点が 80 以上 90 以下の学生番号、科目番号、得点を抽出する。

```
SELECT 学生番号, 科目番号, 得点 FROM 得点表
WHERE (科目番号 LIKE '%2%')
      OR (得点 BETWEEN 80 AND 90)
```

<ex11 の実行結果>

学生番号	科目番号	得点
6724	K21	85
6817	K12	90

<ex12 の実行結果>

学生番号	科目番号	得点
6724	K21	85
6725	K21	60
6817	K11	85
6817	K12	90
6817	K21	95

ex13. “得点表” から、得点が 60 より大きいタブルのうち、学生番号の左端から数えて 2 桁目が '7' か、受験日が '20XX/10/21' のタブルを抽出する。

```
SELECT * FROM 得点表
WHERE 得点 > 60
      AND (学生番号 LIKE '_7__' OR 受験日 = '20XX/10/21')
```

<ex13 の実行結果>

学生番号	科目番号	得点	受験日
6724	K11	65	20XX/10/20
6724	K21	85	20XX/10/21
6817	K21	95	20XX/10/21

※() を付けないと AND が先に評価されて、次のような誤った結果が抽出される。

学生番号	科目番号	得点	受験日
6724	K11	65	20XX/10/20
6724	K21	85	20XX/10/21
6725	K21	60	20XX/10/21
6817	K21	95	20XX/10/21

2-2-3 データのグループ化

データの**グループ化**とは、ある属性が同じ値をもつタプルをまとめて取り扱うことであり、その際に使用する関数を**集合関数**（**集約関数**）という。

データのグループ化には、GROUP BY 節を使用する。

SELECT 抽出項目 **FROM** テーブル名 **GROUP BY** グループ化列名

- ・抽出項目には、グループ化列名、集合関数、定数を記述できる。
- ・代表的な集合関数は、次のとおりである。

集合関数	意味
SUM	グループの合計を求める。
AVG	グループの平均を求める。
MIN	グループの最小値を求める。
MAX	グループの最大値を求める。
COUNT	グループの件数（タプル数）をカウントする。 * ... すべてのタプルを数える。 DISTINCT 列名... 列の値が重複しないタプルを数える。

- ・ **AS** : 集合関数を使用して求めた列に、新たに列名を指定できる。
- ・ **HAVING** : 集合関数を使用した抽出条件を記述する。

ex14. “得点表” から、各科目の平均点を求めて抽出する。

```
SELECT 科目番号, AVG(得点) FROM 得点表
GROUP BY 科目番号
```

ex15. “得点表” から、各学生の受験した科目数を求めて抽出する。

```
SELECT 学生番号, COUNT(*) AS 科目数 FROM 得点表
GROUP BY 学生番号
```

ex16. “得点表” から、各学生の受験した日数を求めて抽出する。

```
SELECT 学生番号, COUNT(DISTINCT 受験日) AS 日数
FROM 得点表
GROUP BY 学生番号
```

< ex14 の実行結果 >

科目番号	AVG(得点)
K11	75
K12	90
K21	80

< ex15 の実行結果 >

学生番号	科目数
6724	2
6725	1
6817	3

< ex16 の実行結果 >

学生番号	日数
6724	2
6725	1
6817	2

ex17. “得点表” から、合計点（得点の合計）が 150 点以上の学生を抽出する。

```
SELECT 学生番号, SUM(得点) AS 合計点 FROM 得点表
GROUP BY 学生番号 HAVING SUM(得点) >= 150
```

< ex17 の実行結果 >

学生番号	合計点
6724	150
6817	270

※誤りの SQL 文（WHERE 句に抽出条件を記述）

```
SELECT 学生番号, SUM(得点) AS 合計点
FROM 得点表
WHERE SUM(得点) >= 150 ... 文法エラー
GROUP BY 学生番号
```

2-2-4 データの整列

データの**整列**とは、抽出結果を特定の属性の指定順に並べ替えることである（整列指定がない場合、元のテーブルに記録してある順に抽出される）。

データの整列には、ORDER BY 節を使用する。

```
SELECT 列名 1, 列名 2, ... FROM テーブル名
ORDER BY 整列指定する列名 整列順序
```

- ・ 整列順序として、次の 2 種類を指定できる。

整列順序	意味
ASC	昇順に整列する（整列順序を省略した場合のデフォルト値）。
DESC	降順に整列する。

- ・ 列の値が同じレコードは、元の記録順序が保たれるのが一般的である。
- ・ 整列指定する列名に、左端から数えて何番目の列かを指定することもできる。

ex18. “得点表” を、得点の降順に整列して抽出する。

```
SELECT * FROM 得点表 ORDER BY 得点 DESC
```

ex19. “得点表” から、各学生の最高得点を求めて昇順に抽出する。

```
SELECT 学生番号, MAX(得点) FROM 得点表
GROUP BY 学生番号 ORDER BY 2
```

< ex18 の実行結果 >

学生番号	科目番号	得点	受験日
6817	K21	95	20XX/10/21
6817	K12	90	20XX/10/20
6724	K21	85	20XX/10/21
6817	K11	85	20XX/10/20
6724	K11	65	20XX/10/20
6725	K21	60	20XX/10/21

< ex19 の実行結果 >

学生番号	MAX(得点)
6725	60
6724	85
6817	95

2-2-5 テーブルの結合

テーブルの**結合**とは、二つ以上のテーブルを一つのテーブルにまとめることである。結合にはいくつかの方法があるが、最も簡単な方法を次の二つのテーブルで説明する。

< α >

X	Y
X1	Y2
X2	Y1

< β >

Y	Z
Y1	Z1
Y2	Z2

二つ以上のテーブルを使用する場合、使用するテーブルをすべて FROM 句に列記する。これにより、指定されたテーブルの全タプルを組み合わせた結果（**直積**）が作成される。二つ以上のテーブルの直積をとる結合を、交差結合（**クロス結合**）という。

【交差結合】 **SELECT * FROM** α , β

二つ以上のテーブルを結合する場合、共通の意味をもつ列の値が同じタプル同士を組み合わせる場合が多い。そのため、WHERE 句の条件により、交差結合の結果から列の値が等しいタプルを抽出する（対応する列の値が等しいタプル同士の結合を**等結合**という）。

【等結合】 **SELECT * FROM** α , β **WHERE** $\alpha.Y = \beta.Y$

< 交差結合の実行結果 >

$\alpha.X$	$\alpha.Y$	$\beta.Y$	$\beta.Z$
X1	Y2	Y1	Z1
X1	Y2	Y2	Z2
X2	Y1	Y1	Z1
X2	Y1	Y2	Z2

< 等結合の実行結果 >

$\alpha.X$	$\alpha.Y$	$\beta.Y$	$\beta.Z$
X1	Y2	Y2	Z2
X2	Y1	Y1	Z1

ex20. “得点表”と“科目表”から、学生番号と受験した科目名、得点を抽出する。

SELECT 学生番号, 科目名, 得点 **FROM** 得点表, 科目表
WHERE 得点表. 科目番号 = 科目表. 科目番号

< ex20 の実行結果 >

学生番号	科目名	得点
6724	英語 I	65
6724	数学	85
6725	数学	60
6817	英語 I	85
6817	英語 II	90
6817	数学	95

[注]

二つのテーブルに共通する列を抽出する場合は、どちらのテーブルの列かを明確にしなければいけない。

例. **SELECT** 得点表. 科目番号, ~

テーブルを FROM 句で指定するときに**相関名**を付けると、ex20 の SQL 文は次のように簡略化して書くことができる。

```
SELECT 学生番号, 科目名, 得点 FROM 得点表 X, 科目表 Y
WHERE X.科目番号 = Y.科目番号
```

ex21. “学生表”，“得点表” および “科目表” から，英語Ⅱを受験した学生の学生番号と氏名を抽出する。

```
SELECT X.学生番号, 氏名 FROM 学生表 X, 得点表 Y, 科目表 Z
WHERE X.学生番号 = Y.学生番号 AND Y.科目番号 = Z.科目番号
AND 科目名 = '英語Ⅱ'
```

< ex21 の実行結果 >

学生番号	氏名
6817	山本 千代

テーブルを結合する方法として，FROM 句の中で JOIN 指定をする方法もある。

・ **CROSS JOIN** (交差結合)

複数のテーブルの直積をとる指定である (テーブルを列記した場合と同じ)。

・ **INNER JOIN** (内部結合)

結合条件により，等結合する指定である。

```
SELECT 学生番号, 科目名, 得点
FROM 得点表 X INNER JOIN 科目表 Y
ON X.科目番号 = Y.科目番号
```

なお，結合条件で指定する項目名が両方のテーブルで同じ場合は，次のように USING 句で結合列名を指定する方法もある。

```
SELECT 学生番号, 科目名, 得点
FROM 得点表 X INNER JOIN 科目表 Y USING (科目番号)
```

・ **OUTER JOIN** (外部結合)

結合条件が成立しなくても優先テーブルの全タプルを抽出する指定である。

・ 左外部結合 (左のテーブルが優先テーブル) : **LEFT OUTER JOIN**

```
SELECT 学生番号, 科目名, 得点
FROM 得点表 X LEFT OUTER JOIN 科目表 Y
ON X.科目番号 = Y.科目番号
```

・ 右外部結合 (右のテーブルが優先テーブル) : **RIGHT OUTER JOIN**

```
SELECT 学生番号, 科目名, 得点
FROM 得点表 X RIGHT OUTER JOIN 科目表 Y
ON X.科目番号 = Y.科目番号
```

<会員表>

会員番号	会員名
20012	池田
30143	梅田

<退会表>

会員番号	申請日	フラグ
10866	10/04	1
20012	12/10	0

【内部結合】

```
SELECT * FROM 会員表 X INNER JOIN 退会表 Y
ON X.会員番号 = Y.会員番号
```

X.会員番号	会員名	Y.会員番号	申請日	フラグ
20012	池田	20012	12/10	0

【左外部結合】

```
SELECT * FROM 会員表 X LEFT OUTER JOIN 退会表 Y
ON X.会員番号 = Y.会員番号
```

X.会員番号	会員名	Y.会員番号	申請日	フラグ
20012	池田	20012	12/10	0
30143	梅田	NULL	NULL	NULL

【右外部結合】

```
SELECT * FROM 会員表 X RIGHT OUTER JOIN 退会表 Y
ON X.会員番号 = Y.会員番号
```

X.会員番号	会員名	Y.会員番号	申請日	フラグ
NULL	NULL	10866	10/04	1
20012	池田	20012	12/10	0

図 4-12 内部結合と左外部結合，右外部結合の実行イメージ

〔表の結合方法〕

・ソートマージ結合法

結合する列の値で並べ替えたそれぞれの表の行を，先頭から順にアクセスして値が等しい行を結合する。表の大部分を結合する場合に有効である。

・ハッシュ結合法

一方の表の結合する列の値でハッシュ表を作成し，もう一方の表の結合する列と結合する。

・ネストループ法

一方の表の結合する列の値を順に読み出し，もう一方の表の結合する列と結合する。インデックスを利用して結合するネストループ法もある。

2-2-6 副照会（副問合せ）

副照会（副問合せ）とは、あるテーブルの検索結果をもとに、別のテーブルを検索することである。条件を求める SELECT 文を副照会文、副照会文の結果を利用してテーブルを検索する SELECT 文を主照会文と呼ぶ。副照会は、大きく二つに分類される。

(1) 単独で実行できる副照会文

主照会文の条件を得るために、単独で実行される副照会文である。この形式では、最初に () 内の副照会文が実行されて、その結果を使って主照会文が実行される。

SELECT 列名, ... **FROM** テーブル名 **WHERE** 列名 比較語 (副照会文)

・比較語には、IN 述語、比較演算子と組み合わせた ANY 述語、ALL 述語がある。

述語	意味
IN	副照会文の結果のいずれかと一致する。(= ANY と同じ)
ANY	副照会文の結果のいずれかと比較演算子の関係が成立する。
ALL	副照会文の結果のすべてと比較演算子の関係が成立する。

※副照会文の結果が必ず一つであれば、単体の比較演算子も使用できる。

ex22. “学生表”と“得点表”から、受験日が '20XX/10/20' の氏名を抽出する。

```
SELECT 氏名 FROM 学生表
WHERE 学生番号 IN ( SELECT 学生番号 FROM 得点表
                     WHERE 受験日 = '20XX/10/20' )
```

<ex22 の実行結果>

氏名
山本 一希
山本 千代

※副照会文の結果により、次の SQL 文と同じ意味になる。

```
SELECT 氏名 FROM 学生表
WHERE 学生番号 IN ('6724', '6817', '6817')
```

また、単独で実行できる副照会文の場合は、FROM 句に記述することで副照会文の実行結果をテーブルとして使用することもできる。

ex23. “得点表”から、受験日が '20XX/10/20' の科目番号と得点を抽出して WTBL 表とし、WTBL 表から得点が 80 点以上の科目番号を抽出する。

```
SELECT 科目番号
FROM ( SELECT 科目番号, 得点 FROM 得点表
        WHERE 受験日 = '20XX/10/20' ) WTBL
WHERE 得点 >= 80
```

(2) 相関副照会（単独では実行できない副照会文）

相関副照会とは、主照会文にあるテーブルの項目を条件に使用しているため、単独では実行できない副照会文である。この場合、主照会文にあるテーブルからタプルを一つずつ取り出して副照会文を実行し、副照会文の結果が1行以上存在するかどうかで該当タプルを抽出するかどうか判定する。この判定には EXISTS 述語が使用される。

ex24. “学生表”と“得点表”から、受験日が‘20XX/10/20’の氏名を抽出する。

```
SELECT 氏名 FROM 学生表 X
WHERE EXISTS ( SELECT * FROM 得点表 Y
               WHERE X.学生番号 = Y.学生番号 AND 受験日 = '20XX/10/20' )
```

ex24 の SQL 文では、()内の副照会文を単独で実行できない（相関名 X を解釈できない）。したがって、学生表のタプルを一つずつ取り出して副照会文を実行する。

- ・学生番号 ‘6724’ のタプルで副照会文を実行する。

```
SELECT * FROM 得点表 Y
WHERE '6724' = Y.学生番号 AND 受験日 = '20XX/10/20'
```

<実行結果>

学生番号	科目番号	得点	受験日
6724	K11	65	20XX/10/20

⇒ 結果が1行以上存在するので、学生番号 ‘6724’ のタプルを抽出する。

- ・学生番号 ‘6725’ のタプルで副照会文を実行する。

```
SELECT * FROM 得点表 Y
WHERE '6725' = Y.学生番号 AND 受験日 = '20XX/10/20'
```

<実行結果>

学生番号	科目番号	得点	受験日

⇒ 結果が1行も存在しないので、学生番号 ‘6725’ のタプルを抽出しない。

このようにして、すべてのタプルの学生番号を使って副照会文を実行することで、抽出するタプルを決定する。

なお、この例を、副照会を使用しない SQL 文で表記すると、次のようになる。注意するのは DISTINCT 指定を省略すると、氏名 ‘山本 千代’ が2回抽出される点である。

```
SELECT DISTINCT 氏名 FROM 学生表 X, 得点表 Y
WHERE X.学生番号 = Y.学生番号 AND 受験日 = '20XX/10/20'
```

また、IN 述語と EXISTS 述語は、論理演算子 NOT と組み合わせて使用することもできる。

- ・ NOT IN : 副照会の結果のいずれとも一致しなければ真
- ・ NOT EXISTS : 副照会の結果、タプルが存在しなければ真

2-2-7 その他の SQL の使用方法

(1) その他のデータ操作言語

- ・ INSERT 文：テーブルにタプルを挿入（追加）する場合に使用する。

```
INSERT INTO テーブル名 VALUES ( データ 1, データ 2, ... )
```

ex25. “学生表” に、学生番号が'6750'、氏名が'林 五十六'、性別が'男'のタプルを追加する。

```
INSERT INTO 学生表 VALUES ('6750', '林 五十六', '男')
```

< ex25 の実行結果 >

学生番号	氏名	性別
6724	山本 一希	男
6725	本山 十吾	男
6750	林 五十六	男
6816	山田 百音	女
6817	山本 千代	女

… 追加されたデータ

- ・ UPDATE 文：テーブルのデータを更新（変更）する場合に使用する。

```
UPDATE テーブル名 SET 列名 = 更新値 WHERE 更新指定条件
```

ex26. “学生表” の学生番号が'6750'の性別を'女'に変更する。

```
UPDATE 学生表 SET 性別 = '女' WHERE 学生番号 = '6750'
```

< ex26 の実行結果 >

学生番号	氏名	性別
6724	山本 一希	男
6725	本山 十吾	男
6750	林 五十六	女
6816	山田 百音	女
6817	山本 千代	女

… 更新されたデータ

- ・ DELETE 文：テーブルのタプルを削除する場合に使用する。

DELETE FROM テーブル名 **WHERE** 削除指定条件

ex27. “学生表” から学生番号が '6750' のタプルを削除する。

DELETE FROM 学生表 **WHERE** 学生番号 = '6750'

< ex27 の実行結果 >

学生番号	氏名	性別
6724	山本 一希	男
6725	本山 十吾	男
6816	山田 百音	女
6817	山本 千代	女

※ {'6750', '林 五十六', '女'} の
タプルが削除される。

(2) カーソル

カーソルは、親言語方式（埋込型 SQL）でデータベースのデータを利用するのに用いられる。通常、親言語（プログラム言語）はデータをレコード単位で扱うのに対し、SQL は集合単位で扱うので、カーソルで 1 行ずつ取り出して渡すことで橋渡しをする。

- ・ DECLARE CURSOR 文：処理対象となる導出表を定義し、カーソルを割り当てる。

DECLARE カーソル名 **CURSOR FOR** **SELECT** ~

※下線部の SELECT 文を、PREPARE 文で実行時に作成する**動的 SQL**もある。

- ・ OPEN 文：カーソルの処理を開始する。

OPEN カーソル名

- ・ FETCH 文：カーソルの位置からデータを読み取る。

FETCH カーソル名 **INTO** 変数名

- ・ CLOSE 文：カーソルの処理を終了する。

CLOSE カーソル名

〔構造化データ〕

RDB の行と列のように、データ構造に一定の規則性（構造定義）があるデータを**構造化データ**という。これに対して、規則性がないデータを**非構造化データ**と呼ぶ。非構造化データは、データベースなどで定型的に扱うのが難しい。そこで、非構造化データに規則性に関する区切りなどを付け、NoSQL などで扱えるようにしたデータが**半構造化データ**である。

3 いろいろなデータベース

現代社会では、データベース技術はさまざまな部分で応用されている。企業を例にとると、企業会計システム、在庫管理システム、文書管理システム、営業支援システムなど、多くのシステムでデータベースが活用されている。この章では、これらのシステムで活用されているデータベース技術について学習する。

3-1 分散データベース

分散データベースは、複数箇所に分散配置したデータベースを、あたかも一つのデータベースのように扱う技術である。本来、データベースとはデータを集中して一括管理する目的で用いるものである。しかし、ネットワークを経由した集中データベースには、データベースの障害がシステム全体の障害に波及する、処理が集中するためネットワークが混雑する、データベースの維持／管理が大変である、という問題点もある。そこで、データベースを分散して構築する分散データベースという考え方が生まれた。

分散データベースでは、**RDA 方式**（RDA：Remote Database Access；遠隔データベースアクセス）を採用し、どのデータベースをアクセスしているのかを利用者が意識することなく利用することができる（これを分散データベースの**透過性**という）。ただし、通常の分散処理システムと同様に、セキュリティ管理など運用面に手間がかかるという問題点もある。また、データの重複が発生する可能性もあるため、データの整合性がとれなくなる危険性もある。分散データベースでは、これらの問題を次の技術で解決している。

・ 2相コミットメント（2相コミット）

分散データベースにおける更新処理を、二つのフェーズ（相）に分けて実行するコミットメント制御方式である。更新の即時性が求められる場合に適している。なお、更新処理の可否を確認せずに、コミット（COMMIT）又はロールバック（ROLLBACK）を指示する制御方式を**1相コミットメント**という。

1) 第1フェーズ

全データベースに対して更新処理の可否を確認する。

2) 第2フェーズ

全データベースが更新可（肯定応答）なら確定（コミット）し、一つでも更新不可（否定応答）なら取り消す（ロールバック）。

・ レプリケーション

オリジナルのデータベースの複製（レプリカ）を分散データベースとして配置し、オリジナルに対する更新を一定間隔又は指定時間で複製データベースに反映させる方式である。更新の即時性が要求されない場合に適している。

3-2 データウェアハウス

データウェアハウスとは、データベースの機能を拡張させ、企業の経営戦略のために必要な意思決定情報を引き出す全社的な統合情報システム（多次元データベース）のことである。**OLTP**（On-Line Transaction Processing；オンライントランザクション処理）などにより蓄積された基幹系システムのデータを抽出し、情報分析用の巨大なデータベースを構築して、データの多元的な分析や予測を行う **OLAP**（On-Line Analytical Processing；オンライン分析処理）に活用する。

- ・ **ETL**（Extract／Transform／Load；抽出／加工／書出し）

基幹系システムなどに蓄積された生データを抽出し、名寄せや標準化などの作業によって利用しやすい形に加工し、データウェアハウスに書き出すこと、及びこれらの一連の処理を支援するソフトウェアである。

- ・ **データクレンジング**（**データクリーニング**）

データベース中の重複や、表記の不統一などを洗い出して整理することである。データベースの最適化や、ETL の加工処理などに利用される。

OLAP などで、経営にとって必要なデータや法則性を数学的・統計的手法により分析する方法を、**データマイニング**という。マイニングは“採掘”という意味であり、蓄積された膨大なデータを分析し、顧客の潜在ニーズなどを見つけてマーケティング戦略などに役立てる。例えば、膨大な販売データの中から、“食パンを購入する人は同時に牛乳も購入する”というような法則性を抽出する。

なお、データウェアハウスの構築からデータマイニングに至るまでの一連のデータ管理／分析作業を、**データアドミニストレーション**と呼ぶこともある。また、ある特定の部門（業務）に必要なデータだけを取り出して作成された小規模なデータウェアハウス（**データマート**）もある。データマートは、エンドユーザのニーズとレベルに合わせてデータを絞り込んだものなので使いやすいが、予想外の法則性を見落とす危険性もある。

3-3 ビッグデータ

インターネットや **IoT**（Internet of Things；世の中に存在するさまざまなものに通信機能をもたせ、インターネットなどを介して相互通信を行う仕組み）の普及により、現在は多くのデータ（例えば、インターネット上のソーシャルメディアの大量の書込みや、IoT 機器などから収集したセンサデータなど）の収集が可能になった（このように大量のデータソースから継続的に集められるデータを、**ストリーミングデータ**という）。

これらの多種多様な巨大なデータである **ビッグデータ**は、その名前が示すようにデータ量がペタバイト（PB； 10^{15} バイト）、エクサバイト（EB； 10^{18} バイト）となるものもある。

ビッグデータは、Volume（量）、Variety（多様性）、Velocity（速度）という三つのVの特徴をもっている。さらに、この三つのVに、Value（価値）、Variability（変動性）、Veracity（正確さ）などの特徴を加えることもある。

このようなビッグデータから、データマイニングによって新たな知識（傾向やパターンなど）を見つけ出すことが求められている。しかし、蓄積されたデータが膨大になると、一般的に使用されているデータベースソフト（RDBMS）では扱うことが困難である。そこで、この問題を解決する方法として、膨大なデータを扱うのに適している **NoSQL** や、大規模データの分散処理を実現する **Apache Hadoop** などが利用されている。

また、ビッグデータの分析においては、**AI**（Artificial Intelligence；**人工知能**）が活用されている。特に、人間の脳神経回路網を数学モデルで表した**多層ニューラルネットワーク**を用いて、コンピュータが人間と同じように認識・判断できるようにする**ディープラーニング**（深層学習）の大きな発展が期待されている。ディープラーニングは、蓄積されたデータなどから特定の規則や意味をもつパターンを識別する**パターン認識機能**、および認識されたパターンなどから、コンピュータが人間と同じように反復学習を繰り返すことで法則性やルールを見つけ出す**機械学習機能**を強化したものである。

ビッグデータの活用、AI、IoTなどの技術がもたらした第4次産業革命（**デジタルトランスフォーメーション**）によって、私たちの住んでいる社会はSociety 4.0（情報社会）からSociety 5.0（超スマート社会）へと変革を遂げている。Society 5.0では、IoT機器などによって収集されたビッグデータがAIで解析され、その結果が社会へと還元されていく自律的なフィードバックのループが期待されている。

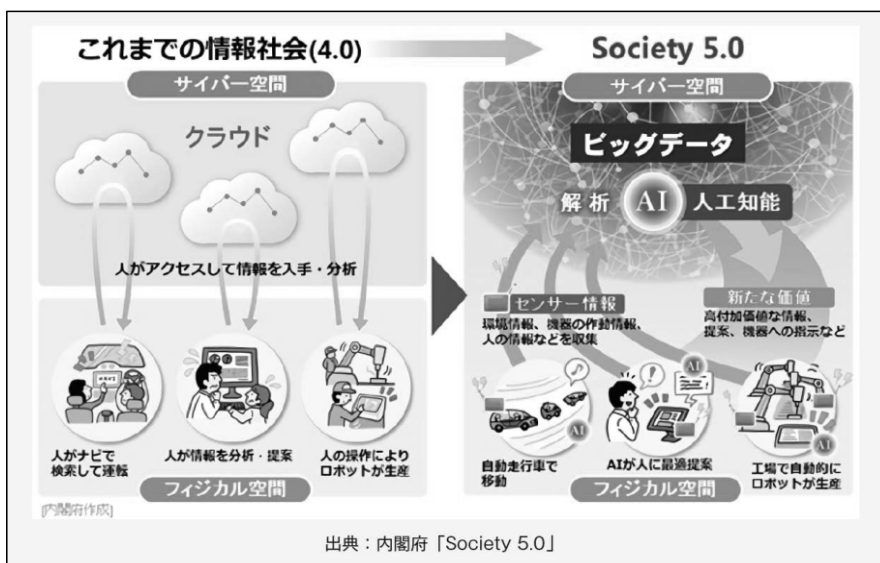


図 4-13 「これまでの情報社会(Society 4.0)」から「Society 5.0」へ

3 - 4 その他の関連技術

(1) IRDS (Information Resource Dictionary System)

IRDS (情報資源辞書システム) は、データの属性、意味内容、格納場所などの**メタデータ**を **DD/D** (**データディクショナリ**) に登録・管理するシステムである。ソフトウェア開発・保守などで、ソースコードなどを一元管理する**リポジトリ**としても用いられる。

(2) 特別なデータベース

・商用データベース

営利を目的に、独自に蓄積した情報を第三者に有償で提供するデータベースである。また、いくつかの商用データベースサービスでは、同義語や類義語などを体系的に分類／整理した辞書 (シソーラス) を用いて、さまざまな表記に対して漏れの少ない検索ができる**シソーラス検索**が可能なものもある。

・OODB (Object Oriented DataBase ; **オブジェクト指向データベース**)

データと処理 (手続) を一体化 (カプセル化) したオブジェクトで管理するデータベースである。利用者が特定の指示 (メッセージ) を与えるだけで、データを処理することができる。

・マルチメディアデータベース

文字や数値以外の情報 (画像、音声など) を扱えるデータベースである。利用者が画像や音声などの多様な情報を意識しながら使用するのは大変なので、OODB (オブジェクト指向データベース) の考え方を採用しているものが多い。

・ハイパテキストデータベース

インターネットなどで利用されるハイパテキスト (キーワードなどにより関連情報を自由に検索／表示できる文書) を管理する網型データベースである。現在では、画像や音声も扱えるハイパメディアデータベースなどがある。

・XML データベース

XML の文書構造 (タグなど) を、そのまま管理できるデータベースである。XML の特徴を活かした、柔軟性と拡張性に優れているデータベースである。

(3) データベース連携技術

・データマッピング

異なるアプリケーション間のデータを関連付ける表 (データマップ) を作成することである。異なる DBMS のデータベースの関連付けなどに用いられる。

・JDBC (Java DataBase Connectivity)

Java からデータベースにアクセスするために用いられる API である。DBMS に依存しない、汎用性の高いプログラムの開発を可能にする。

第4部 演習問題

問1

データベースシステムを導入することによって期待できる効果はどれか。

- ア コード設計作業の軽減
- イ 重複データの削減
- ウ データ転送の高速化
- エ 動的アクセスの実現

問2

関係データベースの説明として、適切なものはどれか。

- ア データは、利用者から見ると2次元の表として取り扱われる。レコード間の関係は相互のレコード中の項目の値を用いて関連付けられる。
- イ レコード間の関係を、1対多の親子関係を用いて表現する。
- ウ レコード間の関係を、網構造によって表現する。
- エ レコードの論理的な配置を、利用者が意識して表現する。

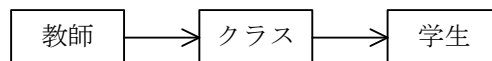
問3

関係演算の射影の説明として、適切なものはどれか。

- ア 表の中から、与えられた条件を満足するタプルを選んで、新しい表を作る。
- イ 表の中から、指定された属性だけを抜き出して、新しい表を作る。
- ウ 二つの表に共通に存在するタプルを抜き出して、新しい表を作る。
- エ 二つの表のタプルから条件に合致したタプル同士を結合して、新しい表を作る。

問4

次のE-Rモデルの解釈として、適切なものはどれか。



- ア 1人の学生は、1人の教師に担当されている。
- イ 1人の学生は、複数のクラスに所属している。
- ウ 1人の教師は、一つのクラスだけを担当している。
- エ 一つのクラスは、担任と副担任の2人の教師が担当している。

問5

“技能記録”表を第二正規形に正規化したものはどれか。ここで、下線部は主キーを表し、{ } は繰返し項目を表す。

技能記録（従業員番号、従業員氏名、{技能コード、技能名、技能経験年数}）

- ア

<u>従業員番号</u>	従業員氏名	<u>技能コード</u>	技能名	技能経験年数
- イ

<u>従業員番号</u>	従業員氏名	<u>技能コード</u>	技能経験年数
<u>技能コード</u>	技能名		
- ウ

<u>従業員番号</u>	<u>技能コード</u>	技能経験年数
<u>従業員番号</u>	従業員氏名	
<u>技能コード</u>	技能名	
- エ

<u>従業員番号</u>	<u>技能コード</u>	
<u>従業員番号</u>	従業員氏名	技能経験年数
<u>技能コード</u>	技能名	

問6

DBMS の排他制御機能に関する記述のうち、適切なものはどれか。

- ア オンラインでの更新時に行うべきものであり、バッチ処理による更新時には行う必要はない。
- イ 関係（表）単位にしか行うことができない。
- ウ 二重更新などにより、データの完全性が損なわれることを防止する。
- エ 排他制御の一つであるロック方式の目的は、デッドロックの防止である。

問7

データベース操作において障害が発生した際に、データベースの状態をトランザクション開始前の状態に戻す障害回復処理はどれか。

- ア 再編成
ウ ロールバック
イ チェックポイント
エ ロールフォワード

問8

“出庫記録”表に対するSQL文のうち、最も大きな値が得られるものはどれか。

出庫記録

商品番号	日付	数量
NP200	20XX/10/10	3
FP233	20XX/10/10	2
NP200	20XX/10/11	1
FP233	20XX/10/11	2

- ア SELECT AVG(数量) FROM 出庫記録 WHERE 商品番号 = 'NP200'
 イ SELECT COUNT(*) FROM 出庫記録
 ウ SELECT MAX(数量) FROM 出庫記録
 エ SELECT SUM(数量) FROM 出庫記録 WHERE 日付 = '20XX/10/11'

問9

“学生”表と“学部”表に対して次のSQL文を実行した結果として、適切なものはどれか。

```
SELECT 氏名 FROM 学生, 学部
      WHERE 所属 = 学部名 AND 学部.住所 = '新宿'
```

学生

氏名	所属	住所
合田知子	理	新宿
青木俊介	工	渋谷
川内聡	人文	渋谷
坂口祐子	経済	新宿

学部

学部名	住所
理	新宿
工	新宿
人文	渋谷
経済	渋谷

ア

氏名
合田知子

イ

氏名
合田知子
青木俊介

ウ

氏名
合田知子
坂口祐子

エ

氏名
合田知子
青木俊介
坂口祐子

問10

“商品”表と“在庫”表から、在庫数が100個未満の商品の商品情報（商品番号、商品名）を抽出するSQLとして、適切なものはどれか。

商品

商品番号	商品名	単価

在庫

商品番号	在庫数

- ア SELECT X.商品番号, 商品名 FROM 商品 X, 在庫 Y
WHERE X.商品番号 = Y.商品番号 OR 在庫数 < 100
- イ SELECT X.商品番号, 商品名 FROM 商品 X, 在庫 Y
WHERE 在庫数 < 100
- ウ SELECT 商品番号, 商品名 FROM 商品
WHERE EXISTS (SELECT * FROM 在庫 WHERE 在庫数 < 100)
- エ SELECT 商品番号, 商品名 FROM 商品
WHERE 商品番号 IN
(SELECT 商品番号 FROM 在庫 WHERE 在庫数 < 100)

問11

分散データベースシステムにおいて、一連のトランザクション処理を行う複数サイトに更新可能かどうかを問い合わせ、すべてのサイトが更新可能であることを確認した後に、データベースの更新処理を行う方式はどれか。

- ア 2相コミットメント イ データクレンジング
- ウ データマイニング エ レプリケーション

問12

ビッグデータを企業で活用している事例はどれか。

- ア カスタマセンタへの問合せに対し、登録済みの顧客情報から連絡先を抽出する。
- イ 最重要な取引先が公表している財務諸表から、売上利益率を計算する。
- ウ 社内研修の対象者リスト作成で、人事情報から入社10年目の社員を抽出する。
- エ 多種多様なソーシャルメディアの大量の書き込みを分析し、商品の改善を行う。

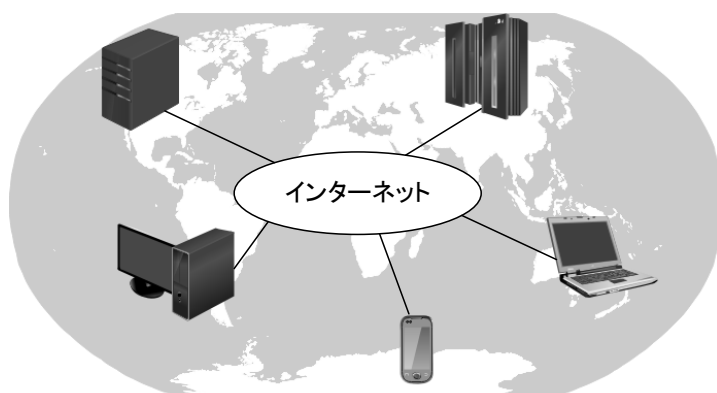
第5部

ネットワーク

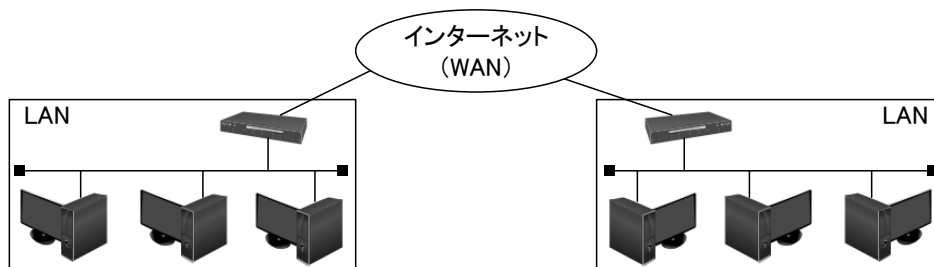
1 インターネット

情報社会において、情報のやり取り（データ通信）を支えているのは、ネットワークである。ネットワークを利用することで、コンピュータ同士で情報のやり取りを行うことができる。

現在、もっとも代表的なネットワークといえばインターネットであり、私たちにさまざまなサービスを提供している。インターネットは、広大なネットワーク（通信網）であり、世界中のコンピュータやモバイル端末、サーバなどがインターネットによってつながっている。実際にサービスを提供しているのは、インターネットを経由して接続されたこれらのサーバである。



インターネットなどのように比較的広い地域に敷設されたネットワークのことを WAN (Wide Area Network) と呼ぶ。これに対して、企業内や構内などのように比較的狭い範囲（敷地）内でコンピュータ、周辺機器などを私設回線で接続して構築されるネットワークのことを LAN (Local Area Network) と呼ぶ（LAN については、第 3 章で詳しく説明する）。現在は、LAN をインターネットに接続してサービスを利用したり、別の LAN とつなげたりすることが一般的である。そのため、インターネットは、“ネットワークのネットワーク” と呼ばれることもある。



この章では、インターネットの仕組みやインターネットが提供するサービスなどについて説明する。

1-1 インターネットの接続方法

インターネットを利用するには一般的に、**インターネットサービスプロバイダ**（ISP：Internet Service Provider）と契約する必要がある。インターネットサービスプロバイダ（単に、**プロバイダ**と呼ばれることもある）は、インターネット接続サービスを提供する**電気通信事業者**であり、回線提供事業者などが兼務していることも多い。

自宅などでインターネットを利用するためには、まず契約しているプロバイダにPCを接続する必要がある。このために使用される公衆回線サービスとして、**FTTH**（Fiber To The Home）がある。FTTHは、高速に通信できる光回線（**光ファイバケーブル**）をHome（一般家庭）まで施設して利用できるようにするサービスであった。しかし、現在では光回線を使用する通信は光通信サービス、光回線サービスと総称されることが多い。

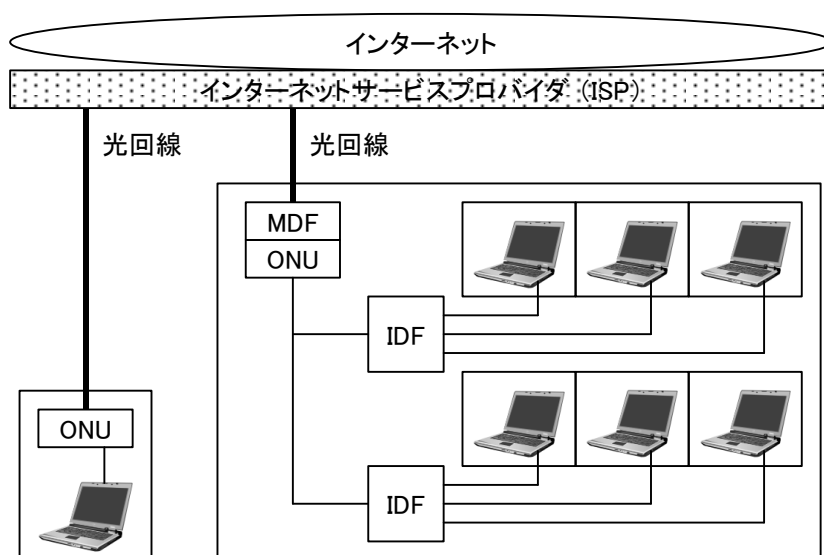


図 5-1 光回線を使用したインターネット接続

- **ONU**（Optical Network Unit）は、DTE（Data Terminal Equipment；データ端末装置）と光回線を接続するDCE（Data Circuit terminating Equipment；データ回線終端装置）で、PC内のデジタル信号を光回線用の光信号に変換／復元する。集合住宅などでは、壁にあるモジュラージャック（差込口）にケーブルをつなぐだけでONUがないように見える場合もあるが、外部事業者と接続する電話回線や通信回線をまとめて管理する**MDF**（Main Distribution Frame；主配線盤）のある部屋に、ONUも設置されているケースが多い。
- **IDF**（Intermediate Distribution Frame；中間配線盤）は、フロアごとに設置され、各家庭の回線をまとめる。

また、**ルータ**を使用することで、複数台の PC などをつなぐこともできる。ルータは、LAN と LAN や、LAN と WAN などを接続するとき使用される LAN 間接続装置である。ルータを使用することで複数台の PC など構築された LAN を、インターネット (WAN) につなぐことが可能となる (現在は、ONU とルータの二つの機能をもつ装置が多くなっている)。また、無線接続用の**アクセスポイント**の機能をもつ**無線 LAN ルータ** (Wi-Fi ルータ) もある (Wi-Fi は無線 LAN の一般的な名称)。Wi-Fi ルータには、プロバイダ側との接続にも無線を使う**モバイル Wi-Fi ルータ**や、衛星 (中継器) を使用した Wi-Fi ネットワーク (メッシュ Wi-Fi) を構築できる**メッシュ Wi-Fi ルータ**もある。

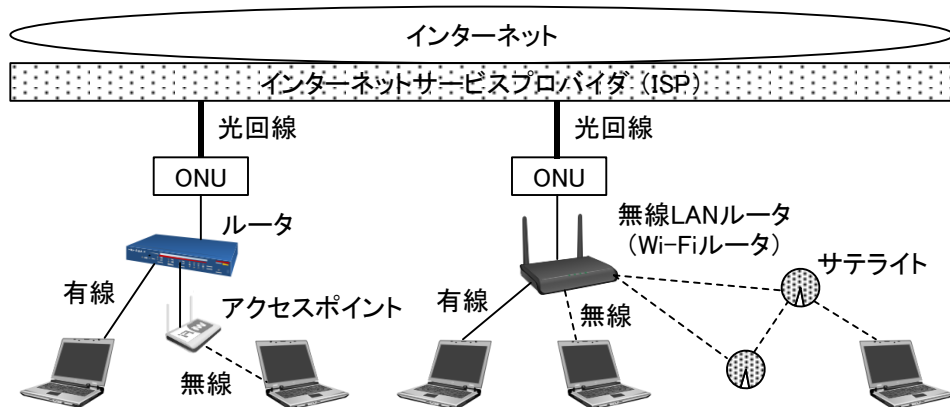


図 5-2 ルータを利用したインターネット接続

さらに、**テザリング**などを使用することで、携帯電話やスマートフォンなどの**モバイル通信** (移動体通信サービス) によって、ノート PC などをインターネットに接続することができます。テザリングでは、複数のアンテナを使用する MIMO などにより第 3 世代携帯電話の規格 (3G) を拡張した **LTE** (Long Term Evolution ; 移動体通信規格) や、**5G** (第 5 世代移動通信システム) による高速なデータ通信サービスを利用できる (現在の携帯電話では、**VoLTE** (Voice over LTE) による高速音声通信サービスも実現されている)。そのほかにも、複数の搬送波による通信を一体的に運用して高速化する**キャリアアグリゲーション**、自動車などに搭載した通信システムを利用してサービスを提供する**テレマティクス**、通信する基地局を移動しながら切り替える**ハンドオーバー**なども導入されている。

なお、光回線は、デジタル信号 (0 又は 1) を送信する**デジタル回線**である。これに対して、音声などのアナログ信号 (波形信号) を送信する**アナログ回線** (既存の電話回線など) もあり、上りと下りの伝送速度が異なるデータ伝送を提供する **ADSL** (Asymmetric Digital Subscriber Line) サービスなどに使用されている。ADSL サービスでは、ONU の代わりに、アナログ信号とデジタル信号を変調 / 復調する**モデム** (MODEM : Modulator / DEModulator) という DCE (データ回線終端装置) を利用する (ADSL 回線では **ADSL モデム** を使用し、**スプリッタ** (分波器) で音声信号とデータ信号を分離・合成する)。ただし、最近では ADSL サービスの利用は減ってきている。

私たちは、通信サービスを提供するプロバイダおよび回線事業者の利用料金や品質によって、契約先を選定する。

通信サービスの利用料金には、次のいずれかの課金方式が用いられることが多い。

- ・ **従量制**：通信サービスの利用時間や送受信データ量に課金する方式
- ・ **定額制**：月額固定料金など、一定の利用料金を最初に設定する方式
 ※一定利用までは基本料金とし、それ以降の利用は従量制で課金する **定額従量制**（半従量制）という方式もある。

一方、通信サービスの品質評価には、次のような指標が用いられる。

- ・ **回線速度（転送速度）**
 回線の伝送能力のことである。一般的には、1秒間に伝送できるデータ量を単位 **bps** (bit per second ; **ビット／秒**) で表す。回線速度と転送データ量、転送時間の関係は、

$$\text{転送時間 (秒)} = \text{転送データ量 (ビット)} \div \text{回線速度 (ビット／秒)}$$
 である。ただし、現実には回線の伝送能力を100%使用することはないので、**回線利用率** (%) を用いて、実際に利用できる実効回線速度を使用する。

$$\text{実効回線速度} = \text{回線速度} \times \text{回線利用率}$$
- ・ **回線容量**
 回線（ネットワーク）全体の伝送能力のことである。例えば、あるネットワークで500Mbpsの回線容量を得るには、実効回線速度が100Mbpsの回線が5本必要になる。このように、必要な回線数などを検討することを **キャパシティプランニング** という。
- ・ **ビット誤り率**（伝送誤り率）
 送信ビット数に対して、伝送誤りが何ビット発生するかを表す確率である。
 （ビット誤り率が 10^{-4} の場合、10,000 ビットごとに1ビットの誤りが発生）
- ・ **呼量／呼損率**
 呼量とは、呼（ネットワーク資源の要求）の延べ利用時間を単位時間で除算した値である（単位は **アーラン**）。呼量が大きくなると、発生した呼がネット

また、通信サービスの品質保証技術（又は品質保証を表す尺度）として、**QoS** (Quality of Service ; **サービス品質**) がある。QoSでは、あらかじめ専用帯域（通信路）を確保しておく帯域制御や、送信データに優先度を付ける優先制御などによって、最低限の回線速度（通信速度）を確保する。これを **ギャランティ型**（品質保証型）という。これに対して、回線速度の上限値を提供するために最大限努力することを保証する通信サービスを、**ベストエフォート型**（最善努力型）という。

1-2 インターネットの基本構成

企業などの社内 LAN から、インターネットを利用する場合の基本構成は、次のとおりである（この基本構成は一例であり、絶対的なものではない）。

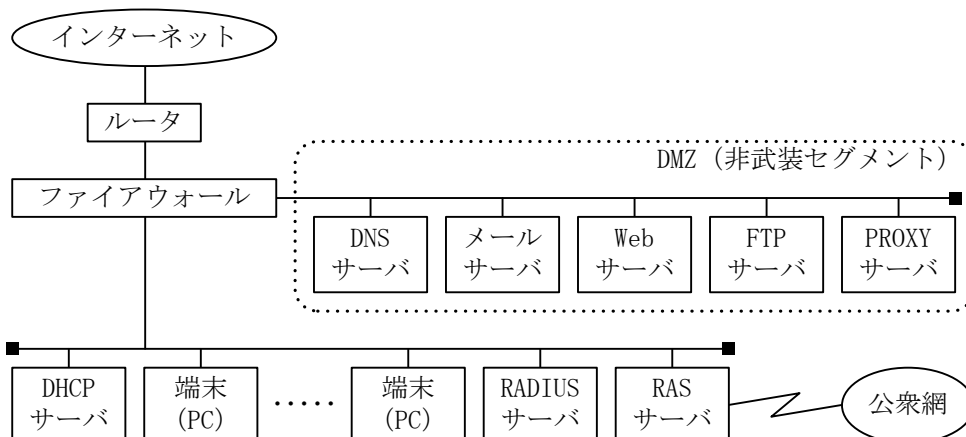


図 5-3 インターネットを利用する場合の基本構成（例）

図 5-3 中の構成要素（サーバなど）の役割は、次のようになる。なお、この後で詳細を説明するものについては、役割を簡単にまとめている。

・ファイアウォール

外部ネットワークからの不正侵入を防ぐための機器である。内部ネットワークと隔離した **DMZ**（DeMilitarized Zone；**非武装セグメント**）を構築し、外部ネットワークと内部ネットワークを直接、接続しないように分離する。

・DNS サーバ（DNS：Domain Name System）

名前解決サービスを提供するサーバである。

・PROXY サーバ（プロキシサーバ又は代理サーバ）

内部ネットワークのクライアント端末に代わって、インターネットに代理接続するサービスを提供するサーバである。

・DHCP サーバ（DHCP：Dynamic Host Configuration Protocol）

IP アドレス（機器を識別するための論理アドレス）の動的割当てサービスを提供するサーバである。

・RADIUS サーバ（RADIUS：Remote Authentication Dial-In User Service）

システム全体の認証を一元化し、ネットワーク管理を簡単にするサービスを提供するサーバである。

・RAS サーバ（RAS：Remote Access Service）

電話回線などの公衆網を経由して、ネットワークにリモートアクセスできるサービスを提供するサーバである。

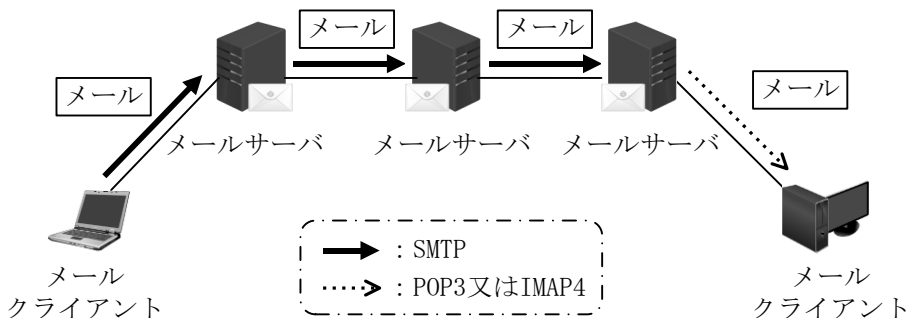
1-3 インターネットサービス

(1) 電子メール (e-mail)

電子メール（以下、メールと略記）は、指定した宛先に送信する電子的な手紙である。テキスト（文字）データのほかに、**MIME**（Multipurpose Internet Mail Extensions）というプロトコルを利用することで、画像データや音声データも送信できる（プロトコルは、通信などに関する取決めのこと）。MIME では、データを 6 ビットずつに区切り、64 種類の英数字に変換する **BASE64** という符号化方式を用いる。

メールは、まず最初に送信側の**メールクライアント**が接続している**メールサーバ**宛てに送られる。メールサーバは、メールの宛先（**メールアドレス**）から送信先メールサーバを決定して、最適な経路を選択する（相手のメールサーバに届く前に、別のメールサーバを経由することもある）。相手のメールサーバにメールが届くと、受信側のメールクライアントが好きなときにメールサーバからメールを取得できる（メールサーバ間を転送されることで伝送遅延が生じたり、送信順序と受信順序が異なったりする場合もある）。

メールをメールサーバに転送する、あるいはメールサーバ間でメールを転送するには、メール転送用プロトコル **SMTP**（Simple Mail Transfer Protocol）が使用される。一方、メールサーバからメールを取得するには、メールサーバ内の全情報を一括して取得する **POP3**（Post Office Protocol version 3）や、ヘッダ情報だけ先に取得してメールを選択できる **IMAP4**（Internet Message Access Protocol version 4）などのメール受信用プロトコルが使用される。IMAP4 は、モバイル環境でのメール利用に効果的である。



メールのヘッダフィールド（ヘッダ情報）には、差出人（From）や宛先（To）のほかに作成日時（Date）、メールソフト（X-Mailer）、メール転送経路（Received）などが記録されている。さらに、同一内容のメールを宛先（To）以外に送る Cc や Bcc という指定が可能である。Cc に指定した場合は、本人以外にもメールを送ったことが全員にわかるが、Bcc に指定した宛先は転送過程で削除されるので本人以外にはわからない。また、宛先メールサーバが見つからなければ“Host unknown”，宛先ユーザが見つからなければ“User unknown”というメッセージ（メール）が差出人に戻される。インターネットを利用しないメールについては、ITU-T X.400（**MHS**：Message Handling System）で勧告されている。

(2) Web／WWW (World Wide Web)

Web／WWW とは、個人や企業が **Web サーバ** に登録した情報 (Web ページ, ホームページ) を、簡単な操作で手に入れることができる仕組み (サービス) のことである。

Web サーバで公開される Web ページは、HTML や XML などのマークアップ言語で作成されていることが多い。Web ページは、関連する Web ページにアクセスできる参照情報 (**ハイパーリンク**) が埋め込まれた、ハイパertext またはハイパメディアである。そのため、**Web クライアント** は Web サーバから Web ページを取得するために、ハイパertext 転送用プロトコル **HTTP** (HyperText Transfer Protocol) を利用する。取得した Web ページは、Web クライアントにある **ブラウザ** と呼ばれるソフトウェアで閲覧することができる。

一般的に、Web ページを参照するには **URL** (Uniform Resource Locator) を使用する。URL は、“プロトコル名：//ホスト名＋ドメイン名/ファイル名”で構成されている (例えば “http://www.its.co.jp/map.html”)。URL は、インターネット上の Web サーバが蓄積しているファイルの所在地を示していて、IP アドレスに変換して利用される。

〔Web 関連技術〕

- **CGI** (Common Gateway Interface)

Web サーバがブラウザからの処理要求に応じて、アプリケーションプログラム (CGI プログラム) を起動する仕組みである。

- **SSI** (Server Side Include)

Web サーバで Web ページ (HTML) に埋め込まれた処理を実行して、その結果を Web クライアントに送信する仕組みである。

- **cookie** (クッキー)

Web サーバに対するアクセスがどのユーザからのものかを識別するために、ユーザ情報 (IP アドレスやユーザ名)、セッション ID、最終セッション確立日/時間などを、ブラウザに書き込む仕組みである。この仕組みによってセッションを解除した後でもユーザを識別して、サービスを提供できる。

- **Web ビーコン**

ユーザのアクセス動向などの情報を収集するために用いられる、Web ページに埋め込まれた小さな画像のことである。cookie と併用される。

- **RSS** (RDF Site Summary)

Web サーバからの情報収集/発信を効率良く行うために、ページの見出しや要約、更新時間などのメタデータを構造化して記述する文書形式である。更新情報 (フィード) を、**RSS リーダ** で簡単に収集することができる。

- **Ajax** (Asynchronous JavaScript + XML)

XML 文書を非同期に送受信して、動的に画面を再描画する仕組みである。

- **CDN** (Contents Delivery Network)

大容量コンテンツを配信するために、最適化されたネットワークである。

(3) 検索エンジン（サーチエンジン）

検索エンジン（サーチエンジン）は、情報を検索するための Web ページの集まりである検索サイトで検索処理を行うプログラムである。情報検索サイトによる情報検索サービスは、Web サービスの一つに含まれる。

検索エンジンによる情報検索の仕組みは、大きく三つに分類される。

- ・ **全文検索型** : 検索文字列を Web ページの全文書中から検索する。
- ・ **ディレクトリ型** : 人手により作成した階層型索引を利用して検索する。
- ・ **ロボット型** : インターネット上の情報を自動で収集してデータベース化するソフトウェア（**クローラ**）を利用して、蓄積・解析された情報の中から検索する。利用者のあいまいな検索要求に対し、その意図や目的を適切に判断・理解する高度な検索技術である**セマンティック検索**が活用されることもある。

(4) ファイル転送サービス

ファイル転送サービスとは、**FTP サーバ**からクライアントにファイルを**ダウンロード**したり、クライアントから FTP サーバにファイルを**アップロード**したりするサービスのことである。ファイル転送サービスでは、ファイル転送用プロトコル **FTP** (File Transfer Protocol) を使用するので、URL のプロトコル名の部分は“ftp”となる。

(5) その他のサービス・技術

インターネットの技術などを応用した、次のようなサービス（活用方法）がある。

- ・ **イントラネット**
インターネットの技術（サービス）を利用した、企業内ネットワークのことである。導入が比較的容易で、初期コストも抑えられる。
- ・ **エクストラネット**
複数の企業間でイントラネットを相互接続したネットワークのことである。
EC（電子商取引）や**EDI**（電子データ交換）などに活用されている。
 - ・ **EDI** (Electronic Data Interchange)
企業間の取引データを電子化し、ネットワークを通じて交換するサービスである。インターネット上で実現している場合を Web-EDI という。
- ・ **オーバーレイネットワーク**
インターネットを基盤として構築される論理的なネットワークのことである。インターネットよりも **QoS**（サービス品質）を向上させたネットワークなどを、論理的に構築して提供することができる。

また、インターネットに接続する**通信サービス**（電気通信事業者が提供しているサービス）や関連するサービス・技術には、そのほかにも次のようなものがある。

- **VPN** (Virtual Private Network)

特定の相手とだけ通信できる**専用線サービス**を、仮想的に実現して提供する仮想私設通信網である。インターネット上で実現される VPN は **インターネット VPN** と呼ばれる。これに対して、電気通信事業者が保有する IP ネットワークを利用して構築した VPN は、**IP-VPN** と呼ばれる。

- **CATV** (CAble TeleVision ; ケーブルテレビ)

本来は、同軸ケーブルを利用した共同アンテナからの配信サービスであったが、現在はインターネット接続やビデオ・オンデマンドにも活用されている。

- **広域 Ethernet** (広域イーサネット)

遠隔地の LAN 同士を接続するサービスである。社内 LAN を直接、接続でき、高速／高品質な通信回線を利用できる。

- **衛星通信サービス**

通信衛星を利用した公衆回線サービスで、ネットワークが構築しにくい山間部などでも利用できる。また、海外との**国際通信サービス**にも活用される。

- **VAN** (Value Added Network ; 付加価値通信網)

非回線通信事業者が、回線通信事業者から回線を借りて構築したネットワークに、何らかの付加価値を付けて提供する通信サービスである。移動体通信事業者から移動体回線網を借りて再販する**仮想移動体通信事業者 (MVNO)**もある。

- **IP 電話**

音声情報（アナログ信号）をデジタル信号に変換し、IP パケットの形式で相手に届ける **VoIP** (Voice over IP) 技術を利用したサービスである。既存の公衆電話網を使用せず、インターネットなどの IP ネットワークを共通の通信基盤（インフラ）として使用できるため、通信コストを削減できる。

- **PLC** (Power Line Communications ; **電力線通信**)

電力線を通信回線として利用する技術である。PLC アダプタをコンセントに接続するだけで、電力線を通信回線として利用することができる。

- **SDN** (Software-Defined Networking)

仮想ネットワークをソフトウェアで制御するネットワーク技術である。オーバレイ技術を利用する方式や、データ転送と経路制御の機能を論理的に分離する **OpenFlow** 技術を用いて、データ転送に特化したネットワーク機器とソフトウェアによる経路制御の組合せで実現する方式がある。

- **ユビキタスネットワーク**

ブロードバンドネットワークと呼ばれる高速ネットワークサービスと情報家電や携帯電話などの複数の情報機器を組み合わせた情報サービスで、日常生活のあらゆる所で利用されている（“ユビキタス”＝“至るところにある”）。

1-4 インターネットの標準プロトコル

TCP/IP はインターネットの標準プロトコルとして、次の4層構成で複数のプロトコルを体系化している（TCP/IPの詳細は第2章で説明する）。

各層の名称	代表的なプロトコル
アプリケーション層	SMTP, POP3, IMAP4, HTTP, FTP
トランスポート層	TCP, UDP
インターネット層	IP, ARP, ICMP
データリンク層	PPP, (Ethernet)

TCP/IP では、送信する**パケット**に各層の管理情報を**ヘッダ**として付加していくことにより、データの送受信を実現している。TCP を用いたコネクション型通信（論理的な通信路を確立する通信）における各層の役割と各ヘッダに含まれる代表的な管理情報は、次のとおりである。

① アプリケーション層

提供サービスに対応するプロトコルが送受信するデータを管理する。

送受信データ

② トランスポート層（プロトコル：TCP）－ 単位名：セグメント

TCP ヘッダにより、サービス（プロトコル）に関する情報を管理する。

TCP ヘッダ			送受信 データ
送信元 ポート番号	宛先 ポート番号	シーケンス 番号	

- ・ **ポート番号**：サービス／プロトコルを表す番号
- ・ **シーケンス番号**：パケットの順序を表す番号

③ インターネット層（プロトコル：IP）－ 単位名：パケット

IP ヘッダにより、IP アドレスに関する情報を管理する。

IP ヘッダ			TCP ヘッダ	送受信 データ
プロトコル 番号	送信元 IP アドレス	宛先 IP アドレス		

- ・ **プロトコル番号**：IP ヘッダの次のヘッダのプロトコルを表す番号
- ・ **IP アドレス**：ネットワークや機器を識別する論理アドレス

④ データリンク層 － 単位名：フレーム

イーサネットヘッダにより、MAC アドレスに関する情報を管理する。

イーサネットヘッダ		IP ヘッダ	TCP ヘッダ	送受信 データ
宛先 MAC アドレス	送信元 MAC アドレス			

- ・ **MAC アドレス**：機器（ハードウェア）固有の識別子（物理アドレス）

1-4-1 トランスポート層の役割

トランスポート層の主たる役割は、送信元のアプリケーションが利用しているサービス（プロトコル）から送信されたデータを、宛先のアプリケーションが利用しているサービス（プロトコル）に届けることである。また、シーケンス番号を利用した順序制御や再送制御なども行われる。

トランスポート層では、送信元及び宛先のサービス（プロトコル）を識別するために、**ポート番号**を利用する。ポート番号には、トランスポート層のプロトコル（TCP、UDP）ごとに決められている、**ウェルノウンポート番号**（well-known）を利用するのが一般的である。**TCP**（Transmission Control Protocol）で使用する代表的なウェルノウンポート番号は、次のとおりである。

ポート番号	サービス	プロトコル
20, 21	ファイル転送サービス（20：転送用，21：制御用）	FTP
25	電子メール転送サービス	SMTP
80	HTML ファイル転送サービス	HTTP
110	電子メール受信サービス	POP3

1-4-2 インターネット層の役割

インターネット層の主たる役割は、送信元の機器（コンピュータなど）から送信されたデータを、ネットワークを経由して宛先の機器に届けることである。

インターネット層のプロトコルである **IP**（Internet Protocol）では、送信元・宛先の機器を識別するために **IP アドレス**を利用する。IP アドレスは、それぞれの機器に割り当てられるユニークなアドレスであり、**NIC**（Network Information Center；ネットワークインフォメーションセンター）、および **JPNIC**（日本ネットワークインフォメーションセンター）により、世界中で重複が起きないように管理されている。

IP アドレスには、**IPv4**（IP version 4）で規定されたものと、**IPv6**（IP version 6）で規定されたものがある。IPv4 で規定された IP アドレスは 32 ビットで構成され、通常は 8 ビットずつ四つの部分に区切って 10 進数で表現する。

IP アドレス	10110011	00110110	01101101	00010010
	↓	↓	↓	↓
	(179	. 54	. 109	. 18)

一方、IPv6 で規定された IP アドレスは 128 ビットで構成され、通常は 16 ビットずつ八つの部分に区切って 16 進数で表し、コロン（:）で区切って表現する。IPv6 は、IPv4 による IP アドレス不足を解消すると同時に、IPsec というセキュアプロトコルを標準仕様とし、パケットの暗号化や認証によってセキュリティ機能を強化している。

IPv4 で規定された IP アドレス（32 ビット）を用いて、IP アドレスの仕組みについて説明する（以降の IP アドレスは IPv4 で規定された IP アドレスのことである）。

(1) IP アドレスの分類

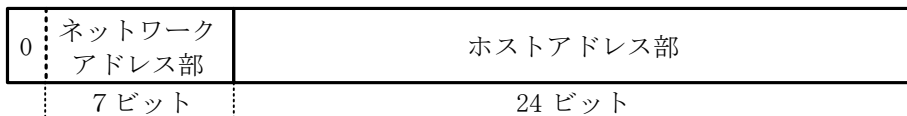
IP アドレスは、ネットワークアドレス部とホストアドレス部で構成されている。ネットワークアドレス部には機器が接続されたネットワークを表す識別子（**ネットワークアドレス**）を、ホストアドレス部には機器を表す識別子（**ホストアドレス**）を記録する。

IP アドレスは、各部で使用するビット数によって、クラス A～D に分類される。このうち、クラス D は **マルチキャスト**（特定のグループに所属する全部の機器にデータを送信する）通信でのみ使用される。また、クラス A～C の IP アドレスで、ホストアドレスをすべて 0 にしたものはネットワークアドレス、すべて 1 にしたものはブロードキャストアドレス（ネットワークに接続する全部の機器にデータを送信する **ブロードキャスト** 通信で利用するアドレス）という特殊な意味があるため、一般的には使用しない。なお、単一の機器を表すホストアドレスを指定した 1 対 1 の通信は、**ユニキャスト** 通信という。

クラス A～D の IP アドレスの構成は、次のとおりである。

・クラス A

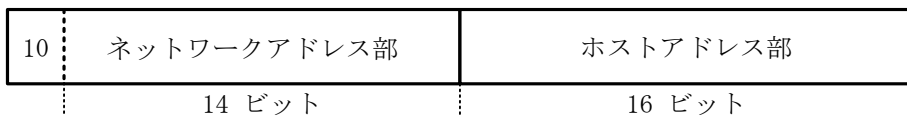
大規模ネットワークに割り当てられるクラスである。



※127. 0. 0. 1～127. 255. 255. 254 は、自分自身を表す **ループバックアドレス**（一般的には、127. 0. 0. 1 をホスト名 localhost として使用する）

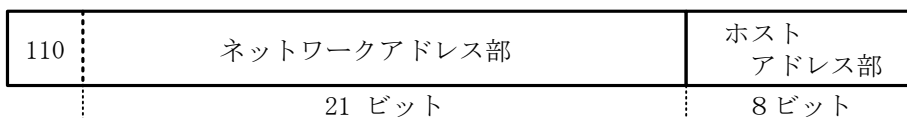
・クラス B

中・大規模ネットワークに割り当てられるクラスである。



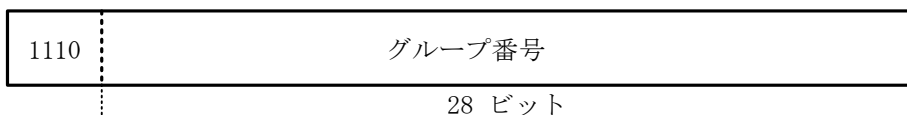
・クラス C

小規模ネットワークに割り当てられるクラスである。



・クラス D

マルチキャスト通信で使用されるクラスである。



(2) IP アドレスの活用

IP アドレスを有効に利用する方法として、次の二つがある。

・サブネット

ホストアドレス部の一部をサブネットアドレス部として利用することで、一つのネットワークアドレスを複数の**サブネットアドレス**とする考え方である。このとき、サブネットアドレスを論理積演算で取り出すために使用されるビット列を**サブネットマスク**という。サブネットの考え方は、大きなネットワークを小さなネットワークに細分化して管理する場合にも使用される。

	ネットワークアドレス部		ホストアドレス部		
IP アドレス (179. 33. 108. 178)	10110011	00100001	011	01100	10110010
サブネットマスク (255. 255. 224. 0)	11111111	11111111	111	00000	00000000
サブネットアドレス (179. 33. 96. 0)	10110011	00100001	011	00000	00000000

この部分を取り出され、ネットワークを識別する。

・CIDR (Classless Inter-Domain Routing)

クラス概念を排除して IP アドレスを利用する考え方である。CIDR では、ネットワークアドレス部のビット数を固定的に定めずに、IP アドレス割当て時に決定する。例えば、先頭から 19 ビットをネットワークアドレスとして使用する場合、“179. 33. 96. 0/19” のように、ネットワークアドレスのビット数 (**プリフィックス値**) を “/” の後に併記する。さらに、可変長サブネットマスク **VLSM** (Variable Length Subnet Mask) を利用することにより、グループごとに異なる長さのネットワークアドレスを利用することも可能にしている。

(3) グローバル IP アドレスとプライベート IP アドレス

グローバル IP アドレスは、機器が識別できるユニークな IP アドレスである。すべての機器にグローバル IP アドレスを割り当てることができれば問題はないが、割り当てられる IP アドレスの数には限りがある。そこで、限定範囲内だけで使用できる**プライベート IP アドレス**という考え方が生まれた。プライベート IP アドレスは、定められた範囲内で識別できればよいので利用者が任意に設定できる。ただし、設定可能なプライベート IP アドレスの範囲は、クラスごとに次のように定められている。

クラス A	10. 0. 0. 0/8 ～ 10. 255. 255. 255/8
クラス B	172. 16. 0. 0/12 ～ 172. 31. 255. 255/12
クラス C	192. 168. 0. 0/16 ～ 192. 168. 255. 255/16

利用範囲が異なれば、プライベート IP アドレスが重複しても問題ない。ただし、外部ネットワークでは一意性を保てないため、プライベート IP アドレスで外部ネットワークに接続することはできない。そのため、プライベート IP アドレスを設定した端末で外部ネットワークに接続するためには、ルータなどでプライベート IP アドレスをグローバル IP アドレスに変換する必要がある。この変換方法には、次の2種類がある。

- **NAT** (Network Address Translation)
プライベート IP アドレスとグローバル IP アドレスを1対1で変換する。
- **NAPT** (Network Address Port Translation)
ポート番号も含めて、複数のプライベート IP アドレスを一つのグローバル IP アドレスに変換する。**IP マスカレード**とも呼ばれる。

なお、IPv6 には、グローバル IP アドレスに該当するグローバルユニキャストアドレスと、プライベート IP アドレスに該当するリンクローカルユニキャストアドレスがある。

(4) DNS (Domain Name System)

DNS は、**FQDN** (Fully Qualified Domain Name ; 完全修飾ドメイン名=ホスト名+ドメイン名) を IP アドレスに変換 (名前解決) するプロトコルである。

IP アドレスは、数字の羅列であるため、人間には覚えにくい。そこで、電子メールの**メールアドレス**やホームページを閲覧する**URL** (Uniform Resource Locator) には、人間に理解しやすい名前 (FQDN) が用いられている (例えば、URL “www1.its.co.jp” とは、ホスト名 (サーバ名) が “www1”、ドメイン名が “its.co.jp” の FQDN である)。

FQDN は、ネットワークの中では IP アドレスに変換して使用される。そのため、FQDN と IP アドレスの対応を管理している **DNS サーバ** により、IP アドレスへの変換が行われる。ただし、1 台の DNS サーバに全世界の FQDN と IP アドレスの対応を記録することは不可能である。そこで、DNS サーバ自体を階層構造で管理している。これにより、上位 DNS サーバから下位 DNS サーバへと順番にたどっていくことで、目的とする IP アドレスの情報が記録された DNS サーバに到達できる仕組みになっている。

※www1.its.co.jp の
IP アドレスを探索

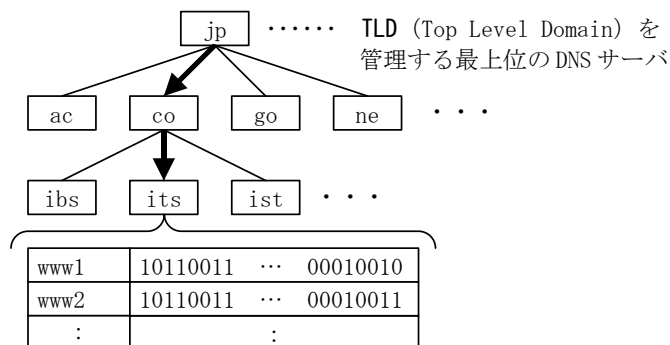


図 5-4 階層構造で管理された DNS サーバ

(5) DHCP (Dynamic Host Configuration Protocol)

DHCP は、ネットワークに接続した機器（コンピュータ）に、IP アドレスを動的に割り当てるためのプロトコルである。DHCP を利用することにより、一つの IP アドレスを複数の機器で共用でき、利用者が IP アドレス関連の設定をする手間も省けるようになる。ただし、同じ IP アドレスを同時に利用することはできないので、**DHCP サーバ**と **DHCP クライアント**では、次の手順によって IP アドレスが未使用であることを確認する。

【DHCP サーバによる確認】

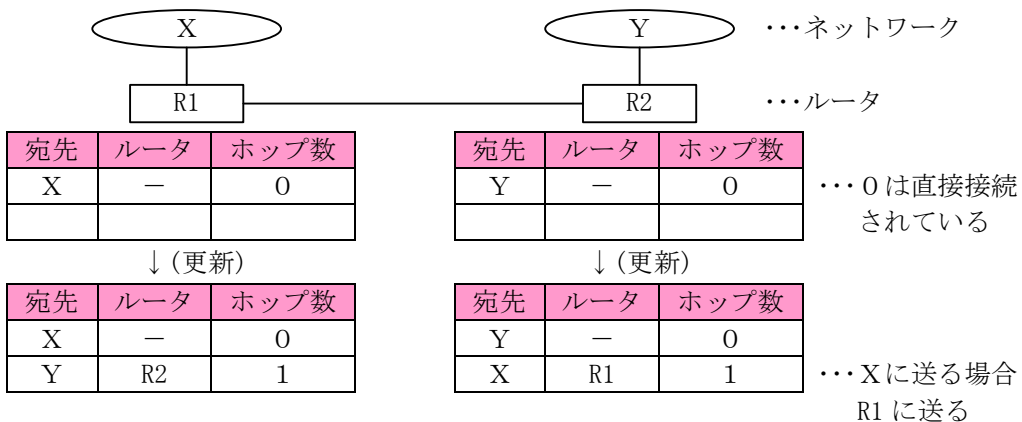
ICMP のエコー要求パケットに割当て予定の IP アドレスを設定して送出し、エコー応答パケットが返ってこないことを確認する（ICMP は、通信エラーやネットワークの状態を通知するプロトコルであり、エコー機能は要求に対する応答があることでパケット到達を確認するものである）。

【DHCP クライアントによる確認】

ARP の要求パケットに割り当てられた IP アドレスを設定して送出し、応答パケットが返ってこないことを確認する（ARP は、IP アドレスに対応する MAC アドレスを取得するプロトコルである）。

(6) RIP (Routing Information Protocol)

RIP は、IP アドレスを利用して目的のネットワークまでの経路選択（**ルーティング**）をするためのプロトコルである。RIP では、経由するホップ数（ルータ数）を基準として、ホップ数が最小となる経路を選択する距離ベクトルアルゴリズムを利用している。各ルータはルーティングテーブルを保持し、ルーティングテーブルの内容を一定時間ごとに送受信することで、ルーティングテーブルの内容を更新していく。ただし、“並列経路が作成できない”，“最小ホップ数=最短経路とはならない”などの欠点がある。

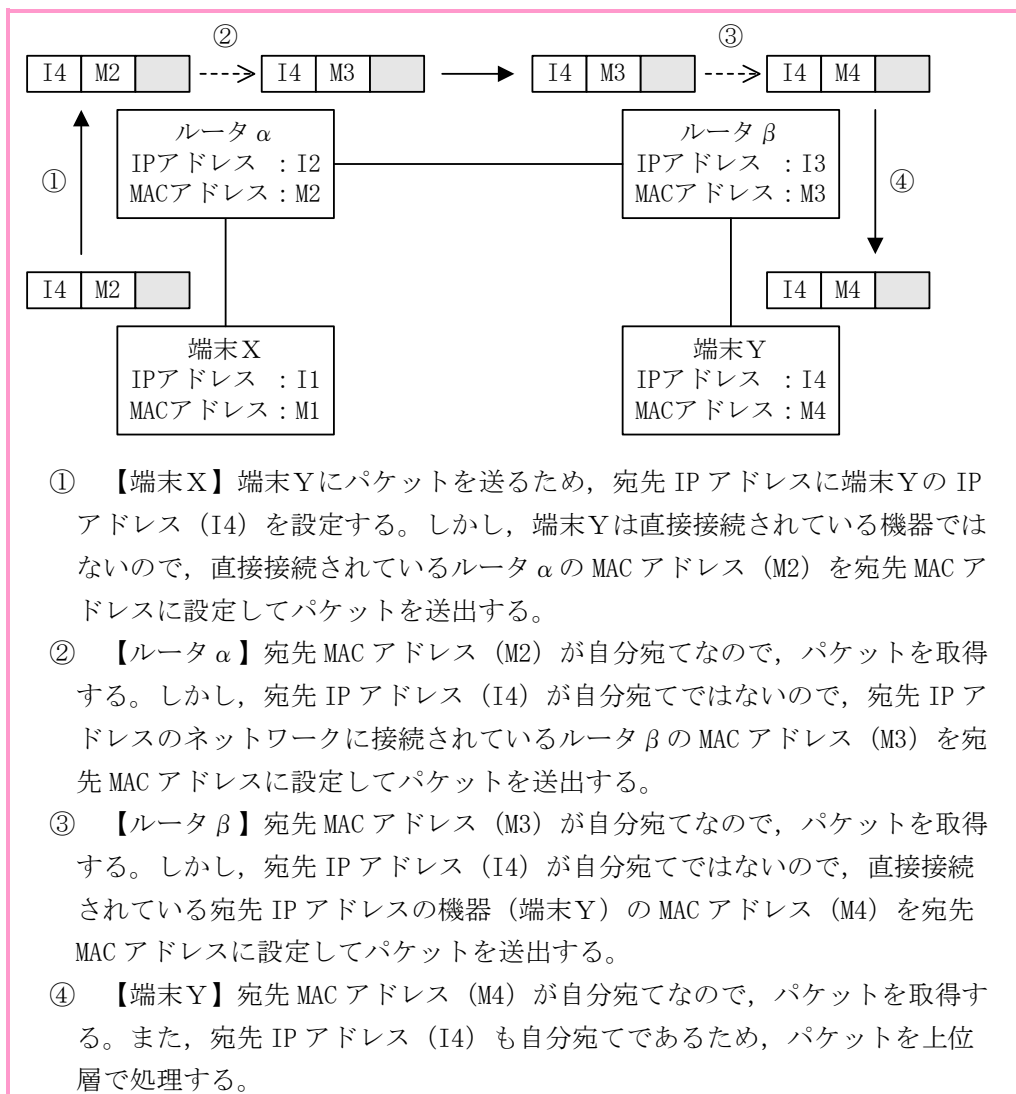


1-4-3 データリンク層の役割

データリンク層の主たる役割は、通信媒体（ケーブルなど）で直接接続された機器（コンピュータなど）に、データを届けることである。

データリンク層では、それぞれの機器を識別するために **MAC アドレス** を利用する。MAC アドレスは、機器（ハードウェア）固有の識別子である（ただし、IP アドレスのように一元管理されていないので、絶対にユニークであるという保証はない）。ネットワークに接続された機器は、通信路上を流れるパケットの宛先 MAC アドレスが自分の MAC アドレスと一致したとき、そのパケットを取得する。

IP アドレスと MAC アドレスを利用した通信の流れは、次のとおりである。なお、宛先 IP アドレスに対応した MAC アドレスは、**ARP** (Address Resolution Protocol) によって取得する。



2 ネットワークアーキテクチャ

JIS では、ネットワークアーキテクチャを“ネットワークの論理構造及び動作原則”と定義している。具体的にいえば、ネットワークを構築するために必要な各種プロトコルやインタフェースを階層的に整理し、体系化したものである。

2-1 ネットワークアーキテクチャとは

ネットワークを通してデータを正しく伝送するためには、通信相手との間でさまざまな取決めをしておく必要がある。伝送信号の電圧レベルや送受信の方法、送信するデータの形式など、決めなければならないことは非常に多い。これらの取決め（約束事）を、**プロトコル**（通信規約）という。

ネットワークが拡大して多くの相手とデータ通信を行うようになると、相手ごとにプロトコルを設定するのは大変になってきた。そこで、プロトコルを体系化して、ネットワークの構築／設計を簡単にするために**ネットワークアーキテクチャ**が考えられた。

2-2 OSI（開放型システム間相互接続）

OSI（Open Systems Interconnection；開放型システム間相互接続）は、国際標準化機構（ISO）や国際電気通信連合（ITU-T）などが中心となって 1983 年に制定した、国際標準のネットワークアーキテクチャである。ネットワークアーキテクチャが考えられた当初は、各メーカーが固有のネットワークアーキテクチャを発表した。しかし、異なるメーカーの製品でネットワークを構築するには、メーカー固有のネットワークアーキテクチャでは役に立たない。そこで、標準的なネットワークアーキテクチャが必要とされたのである。

OSI では、通信に必要な機能が 7 層（レイヤ）に階層化されている。これを、**OSI 基本参照モデル**という。

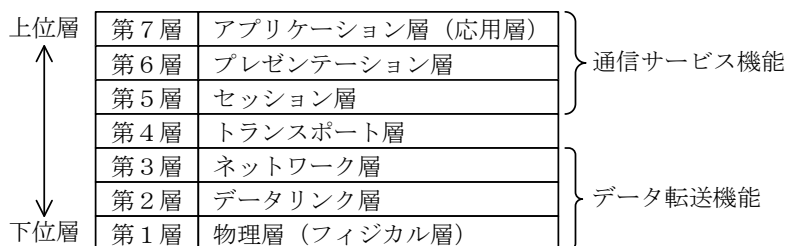


図 5-5 OSI 基本参照モデル

OSI の各層には、機能モジュール（エンティティ）が複数存在する（第N層にあるエンティティを(N)エンティティという）。(N)エンティティは、一つ下の層に属する(N-1)エンティティからサービスを提供され、一つ上の層に属する(N+1)エンティティにサービスを提供する（第N層のエンティティが提供するサービスを(N)サービスという）。また、送信側と受信側の同一レベルに属する(N)エンティティ間のプロトコルを、(N)プロトコルという。(N)エンティティ間では、論理的な通信路である(N)コネクションを使用して、データのやり取りを行う。

OSI の各層が提供するサービスと、具体的な機能は次のとおりである。

① **アプリケーション層**（応用層）

OSI 基本参照モデルの最上位に位置し、アプリケーションプログラムに通信機能（サービス）を提供する。具体的には、ファイル転送アクセス管理（FTAM）や、遠隔データベースアクセス（RDA）などがある。

② **プレゼンテーション層**

アプリケーション層が交換するデータの表現形式（抽象構文）を、汎用的な構文（転送構文）に変換する。具体的には、異なるコード体系のデータを、同じコード体系でデータ通信を行えるように変換する。

③ **セッション層**

各種アプリケーションで共通に利用できるデータ転送機能を提供する。具体的には、同期点（データの区切り）の設定や送信権（トークン）の制御などの会話制御機能によって、データ交換を管理している。

④ **トランスポート層**

送信側と受信側（エンド・ツー・エンド間）に、透過的（トランスペアレント）で高品質なデータ転送を提供する。具体的には、下位層で不足しているサービスを補完して、誤り制御やフロー制御などを行う。

⑤ **ネットワーク層**

エンド・ツー・エンド間での通信路を提供する。具体的には、データ伝送を行うための経路選択（ルーティング）機能や、データの転送／中継機能などがある。

⑥ **データリンク層**

隣接するノード間での、信頼性・透過性の高いデータ伝送を提供する。具体的には、データリンクコネクションの確立や、HDLC 手順などに代表される伝送制御などを行う。

⑦ **物理層**（フィジカル層）

通信回線を用いた、ビット単位での伝送機能を提供する。具体的には、コネクタの形状などの機械的な規格、電圧レベルなどの電気的な規格、信号線の制御方式などの論理的な規格などを定めている。

2 - 3 TCP/IP

TCP/IP は、インターネットなどで使用されるプロトコルである。アメリカの ARPANET（現在のインターネットの原型）で使用されていたプロトコルであったが、利用者の増加に伴い、標準プロトコルとなった **デファクトスタンダード**（業界標準規格）である。

TCP/IP は複数のプロトコルの集合体であり、OSI と同様に階層的に体系化されている。TCP/IP の各層と OSI 基本参照モデルの対応は、次のとおりである。

OSI 基本参照モデル	TCP/IP
アプリケーション層	アプリケーション層（AP 層）
プレゼンテーション層	
セッション層	
トランスポート層	トランスポート層／TCP 層
ネットワーク層	インターネット層／IP 層
データリンク層	データリンク層（リンク層） ／ネットワークインタフェース層（NI 層）
物理層	

※TCP/IP の層名における（ ）内は略称、／以下は別称を表す。

(1) アプリケーション層（AP 層）

アプリケーション層は、ユーザに対して、各種サービスを提供する。インターネットで提供されるサービスは、この層のプロトコルにより実現されている。

プロトコル名	基本的な役割
DNS	FQDN（完全修飾ドメイン名）を IP アドレスに変換する。
DHCP	IP アドレスを動的に割り当てる。
SMTP	メールサーバ宛て又はメールサーバ間で電子メールを転送する。
POP	メールサーバから電子メールを取り出す。
IMAP	メールサーバから電子メールを取り出す。
MIME	電子メールで音声・映像のデータを扱えるようにする。
HTTP	ハイパテキスト（HTML 文書など）を転送する。
FTP	ファイルを転送する。
SNMP	ネットワークを簡易的に管理する。
TELNET	遠隔地の端末からリモートログインする。
NTP	複数のノードにおける時刻の同期を図る。
NNTP	ネットニュースの記事を配信する。
RTP	映像と音声データをリアルタイムに適した形で転送する。
BOOTP	OS の起動（ブート）時にネットワークの設定を取得する。
SOAP	XML をベースにしたデータやサービスを呼び出す。

(2) トランスポート層／TCP 層

トランスポート層／TCP 層は、エンド・ツー・エンド間の透過的・高品質・高速なデータ伝送を提供する（OSI 基本参照モデルのトランスポート層）。

プロトコル名	基本的な役割
TCP	論理的な通信路を確立するコネクション型通信において、高信頼性を保証する。
UDP	論理的な通信路を確立しないコネクションレス型通信において、信頼性を保証しない代わりに高速性を提供する。

(3) インターネット層／IP 層

インターネット層／IP 層は、データ伝送を行うための経路選択（ルーティング）機能や、データの転送／中継機能を提供する（OSI 基本参照モデルのネットワーク層）。

プロトコル名	基本的な役割
IP	IP アドレスを利用してパケットを転送する。
RIP	通信経路を選択するために使用される。
ARP	IP アドレスから MAC アドレスを得る。
RARP	MAC アドレスから IP アドレスを得る。
ICMP	通信エラーやネットワークの状態を通知する。

(4) データリンク層／ネットワークインタフェース層（NI 層）

データリンク層／ネットワークインタフェース層は、トランスペアレントで誤りのないデータ伝送を提供する。OSI 基本参照モデルにおける、物理層とデータリンク層に対応する。なお、データリンク層は、LLC（論理データリンク）副層と MAC（メディアアドレス制御）副層に分割して取り扱うのが一般的である。

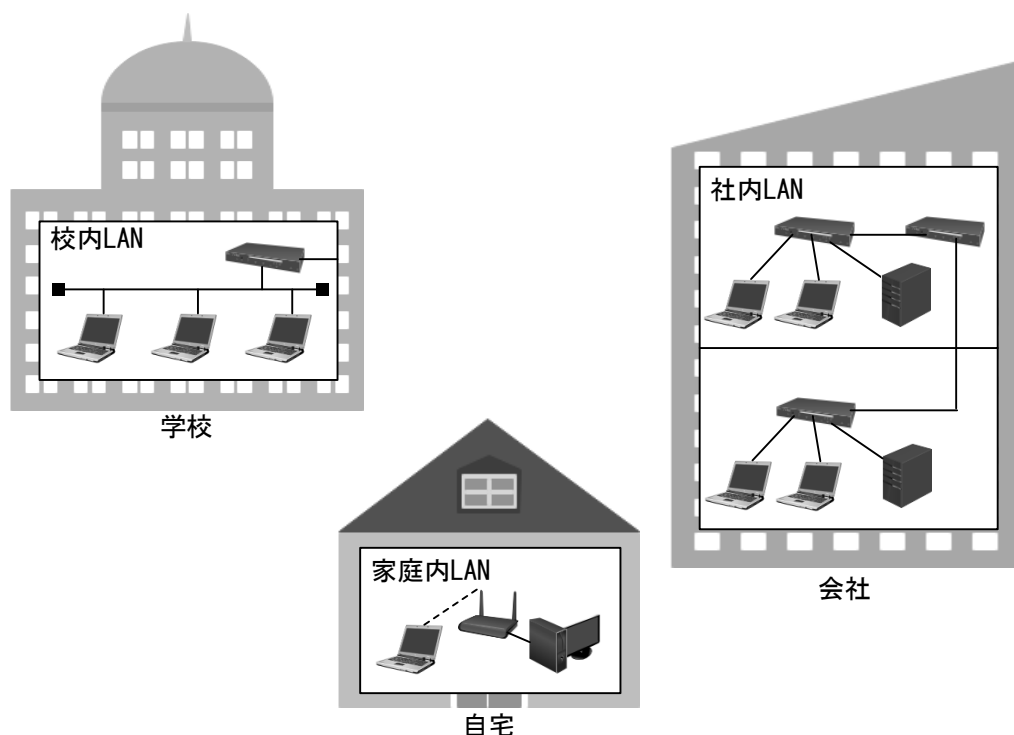
プロトコル名	基本的な役割
PPP	電話回線を使用してネットワークに接続する。
PPPoE	ダイヤルアップ接続をイーサネット上で行う。
IPoE	IP パケットをイーサネットで伝送する（IPv6 のみ対応）。

〔CORBA (COmmon Request Broker Architecture)〕

CORBA は、分散ネットワーク環境の下で、異なったプログラム言語で作成したオブジェクト同士のメッセージ交換を可能にする標準仕様（規格）である。広い意味で、ネットワークアーキテクチャの一つといえる。

3 LAN

LAN (Local Area Network) は、狭い範囲（敷地）内に敷設されたネットワークであり、現在では、インターネットと双璧をなすネットワークといっても過言ではない。学校や会社、自宅などには数多くの LAN が構築され、私たちはそれらの LAN に接続された機器を使ってネットワークを活用している。



LAN を構築することによって、次のようなメリットが得られる。

- ・私設回線なので、回線設置費用はかかるが、回線使用料は発生しない。
- ・限定地域内のネットワークなので、データ伝送が高品質・高速である。
- ・グループウェアなどを利用することにより、グループ活動を効果的に実施できる。
また、オフィスなどのペーパーレス化も推進される。
- ・プリンタなどのハードウェア資源から、ファイルやデータベースなどに至るまで、さまざまな資源を共有することができる。

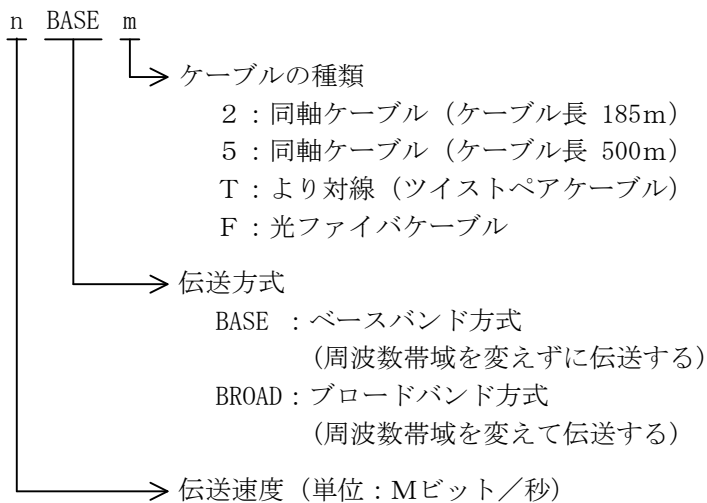
このように数多くのメリットがある LAN は、今後も、ネットワークの中心に位置付けられ、広く活用されていくと考えられる。この章では、このような LAN を構築する機器や技術などについて学習する。

3-1 LAN の基礎技術

(1) 有線 LAN

LAN には、企業内や構内のコンピュータ、周辺機器などが私設回線で接続される。**有線 LAN** は、通信回線（ケーブル）を利用して接続する LAN で、IEEE 802.3 などの規格がある。IEEE 802.3 規格の LAN は**イーサネット**（Ethernet）とも呼ばれ、次の標準 LAN 記号によって表される。例えば、1000BASE-T は伝送速度が 1,000Mビット/秒で、より対線（ツイストペアケーブル）を使用したベースバンド方式の LAN である。

【標準 LAN 記号】



有線 LAN では、機器を接続する媒体（ケーブル）として次の 3 種類を利用する。また、PC を LAN（ケーブル）に接続するために、**LAN モジュール**（標準搭載又は後付けの **LAN ボード**や **LAN カード**／**NIC**（Network Interface Card）など）を使用する。

・より対線（ツイストペアケーブル）

電話回線として使用されているケーブルである。低価格であるが、雑音に弱く、伝送距離も数百 m 程度である。小規模 LAN で一般的に使用される。

・同軸ケーブル

中心導体を外部導体や絶縁体でくるんだケーブルである。価格はやや高いが、雑音に強く、伝送距離も数十 km まで利用可能である。

・光ファイバケーブル

光を伝送する光ファイバを束ねたケーブルで、非常に高速／高品質なデータ伝送が可能である。雑音にも強く、光の減衰が少ないため 100km 程度の長距離伝送が可能である。

(2) 無線 LAN

無線 LAN は、電波や赤外線などを利用して接続する LAN で、**IEEE 802.11** という規格がある（現在は、IEEE 802.11ac が主流である）。なお、IEEE 802.11 の規格を使用した相互通信（無線 LAN）については、**Wi-Fi** の名称が一般的に普及している。

規格名	周波数帯域	最大伝送速度
IEEE 802.11a	5GHz 帯	54Mbps
IEEE 802.11b	2.4GHz 帯	11Mbps
IEEE 802.11g	2.4GHz 帯	54Mbps
IEEE 802.11n	2.4GHz／5GHz 帯	600Mbps
IEEE 802.11ac	5GHz 帯	6.9Gbps
IEEE 802.11ad	60GHz 帯	6.8Gbps

無線 LAN は、ケーブルを使用しないため、レイアウト変更などに柔軟に対応でき、障害物がなければ 50～100m 程度離れていても通信可能である。無線 LAN を利用する場合は、端末を**無線 LAN アクセスポイント**（単に**アクセスポイント**と呼ばれることが多い）に接続する。このとき、接続するアクセスポイントを識別するために、アクセスポイント識別子 **SSID** (Service Set Identifier) を設定する。SSID を設定することにより、同じ SSID が設定された機器同士を接続することができる。しかし、SSID はアクセスポイントに固有の値であるため、同じネットワークに接続する複数のアクセスポイントを渡り歩くことはできない。そこで、SSID の拡張版であるネットワーク識別子 **ESSID** (Extended SSID) を設定する。ESSID はネットワークに固有の値であるため、同じネットワークに接続されているアクセスポイントであれば、電波状態の良いほうを選んで渡り歩くことができる（これを**ローミング**という）。なお、ESSID を利用する場合、アクセスポイントを識別するために機器固有の識別子 **BSSID** (Basic SSID) を用いる。この BSSID には、機器固有の MAC アドレスをそのまま用いるのが一般的である。SSID 又は ESSID を設定することにより、同じ SSID 又は ESSID をもつ装置だけを正当な利用者（装置）と認識して、不正な利用（アクセス）を防止することができる。ただし、この方法だけで安全性を確保するのは難しいため、セキュリティ対策を十分に検討する必要がある。

また、無線 LAN は無線 LAN アクセスポイントを経由して、有線 LAN やインターネットと接続させることも可能である。

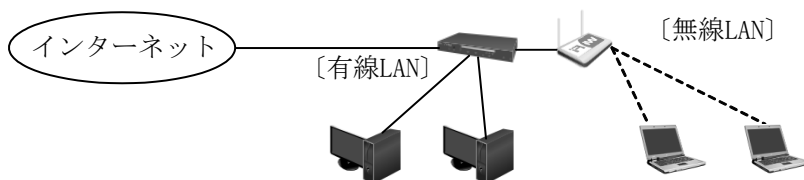


図 5-6 無線 LAN と有線 LAN／インターネットの接続

3-1-1 トポロジ（接続形態）

LAN の**トポロジ**（接続形態）には、次のような種類がある。なお、LAN のトポロジには物理的トポロジと論理的トポロジがあり、両者が必ず一致するとは限らない。

- ・ **スター型**

中央に設置してあるノード（集線装置など）に、すべてのノード（機器）を接続する形態である。一つのノードの障害が全体に波及することはないが、集線装置で障害が発生した場合は、その影響が全体に及ぶ。

- ・ **リング型**

ノードを環状に接続する形態である。通信路が短く、制御も簡単で効率的であるが、一つのノードの故障が全体に及ぶ可能性がある。

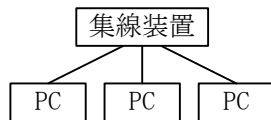
- ・ **バス型**

1本の基幹ケーブル（共用回線）に、すべてのノードが接続されている形態である。ノードの追加や削除は、全体に影響を与えることなく容易に行うことができる。しかし、ノードが多くなって、トラフィック量が増大すると、基幹ケーブル上で衝突が多発し、急激に伝送効率が悪くなる。

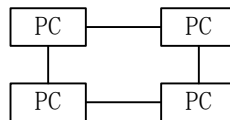
- ・ **ツリー型**

集線装置を**カスケード接続**（多段階接続）することで、根から枝葉が伸びていくようにノードを接続する形態である。

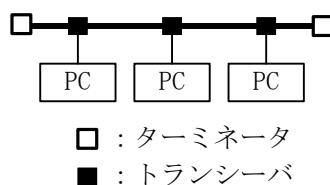
スター型



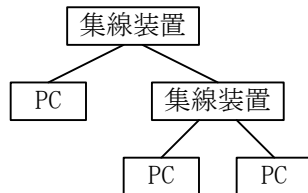
リング型



バス型



ツリー型



なお、無線 LAN などでは、端末同士を直接つなぐ**ポイントツーポイント方式**がある。無線 LAN の接続形態には、子機であるクライアント端末（PC など）同士をポイントツーポイントで直接、接続する**アドホックモード**と、親機に当たるアクセスポイントを経由して接続する**インフラストラクチャモード**の二つがある。

3-1-2 MAC (Media Access Control)

MAC (メディアアクセス制御) は、データの送受信方法や誤り検出方法などを規定するものである。LAN では、主にデータを送信する際のアクセス権の取得方法を制御する。LAN の MAC に関しては、OSI 基本参照モデルのデータリンク層の MAC 副層で規定されている。そのため、データ (フレーム) を送受信する際に、送信元と宛先をデータリンク層の識別子として **LAN モジュール** などに記録されている **MAC アドレス** で特定する。

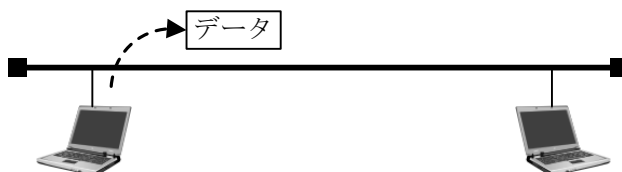
MAC アドレス	OUI (ベンダ ID) 24 ビット	固有製造番号 24 ビット

(1) CSMA/CD

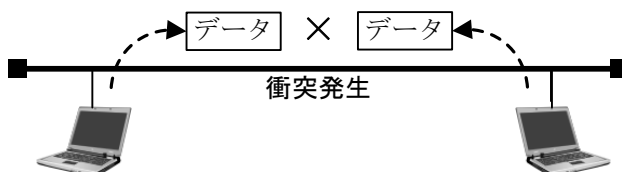
CSMA/CD (Carrier Sense Multiple Access with Collision Detection) は、主にバス型 LAN で使用される (バス型 LAN の規格である IEEE 802.3 で規定されている)。

データを送信する場合、最初に、基幹ケーブル上にデータが流れているかどうかを確認する。流れていなければ、アクセス権を得たものとして送信を開始する。このとき、ほぼ同時にデータを送信する別のノードがあると、基幹ケーブル上でデータの衝突 (コリジョン) が発生する (衝突が起きる範囲のことを **コリジョンドメイン** という)。この衝突を検知した場合、データの送信を一旦停止し、各ノードに設定された一定時間 (ノードごとに異なる時間) の経過後に再送信を行う。

- 1) 基幹ケーブルにデータがないので、送信を開始する。



- 2) 基幹ケーブル上でデータの衝突が発生した場合、送信を停止する。
→ 一定時間経過後、再度送信を開始する。



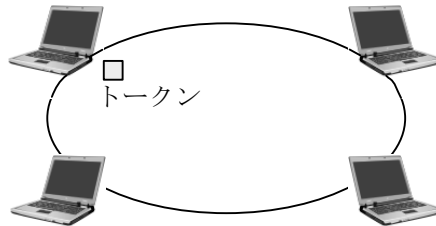
なお、CSMA/CD は半二重通信 (データを交互に送信する双方向通信) での利用を前提としているため、全二重通信 (データを同時に送れる双方向通信) が一般的である現在ではあまり使用されていない。

また、無線 LAN では衝突を検出できないため、受信側からの ACK 信号によりデータ到着を確認する **CSMA/CA** (CSMA with Collision Avoidance) が使用される。

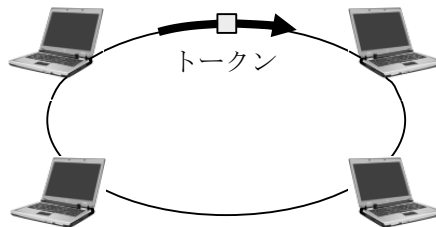
(2) トークンパッシング

トークンパッシングは、トークン（送信権）と呼ばれる特殊なデータを取得した端末がデータ送信を行う方式である。リング型 LAN の場合を**トークンリング**（IEEE 802.5）、バス型 LAN の場合を**トークンバス**（IEEE 802.4）という。

- 1) トークンを取得した端末がデータを送信する。



- 2) データ送信が終了した端末は、トークンを巡回させる。
→ 次の端末がトークンを取得して、データ送信を開始できる。



(3) TDMA (Time Division Multiplex Access ; 時分割多元接続)

TDMA（時分割多元接続）は、タイムスロットが割り当てられたノードだけがデータ送信を行える方式である。一定時間ごとにデータ送信が行えるので、トラフィック量が多い高負荷時でも大きな伝送遅延が発生しにくくなる。

3-1-3 接続機器の関係

LAN の接続機器の関係は、クライアントサーバ型とピアツーピア型に分類される。

クライアントサーバ型は、接続される機器に明確な上下関係がある。具体的には、処理を依頼する（サービスを要求する）クライアントと、処理を行う（サービスを提供する）サーバにより構成される。

一方、ピアツーピア型（**P2P**）は、接続される機器がすべて対等の立場にある関係である。ある一時点において、処理を要求したクライアントであった機器が、別の時点では処理を要求されるサーバの役割もこなすことになる。

3-1-4 LAN 間接続装置

LAN 間接続装置は、複数の LAN を接続する装置である。LAN は、本来、限定地域内でのネットワークであった。しかし、処理内容の多様化により、他の LAN と接続して広域的なネットワークにすることが求められ、LAN 間接続装置が必要となった。

- ・ **リピータ**

LAN を物理層レベルで接続する装置である。ネットワークを物理的に接続し、信号の増幅や整形だけを行う。したがって、接続された LAN は伝送路長を延長するだけで、物理的には一つの LAN として取り扱われる。同等の機器として、**ハブ（リピータハブ）**がある。ハブは、**カスケード接続（多段階接続）**をすることができるが、接続段数に制限があるので注意が必要である。

- ・ **ブリッジ**

LAN をデータリンク層レベルで接続する装置である。MAC アドレスの学習機能及びフィルタリング機能があるため、同一 LAN 内への送信データは別の LAN に流れない（コリジョンドメインを分割できる）。このため、LAN 内のデータ量（トラフィック量）を抑えることができる。ただし、ARP 要求などのようなブロードキャスト通信はブリッジを越えて送られる（ブロードキャスト通信が届く範囲を**ブロードキャストドメイン**という）。また、LAN の回線が循環してデータがいつまでも流れ続けることを防ぐために、仮想的なツリー構造を構成する**スパニングツリー**機能もある。ブリッジと同等の機能をもつ機器として、**スイッチングハブ（レイヤ2スイッチ**又は**L2 スイッチ）**がある。

- ・ **ルータ**

LAN をネットワーク層レベルで接続する装置である。ネットワークアドレス（IP アドレス）によるルーティング機能をもつため、目的の LAN にだけ最適経路でデータを送信することが可能となる。ルーティング機能の中には、目的の LAN に到達するまでに経由するルータまで厳密に定める**ストリクトルーティング**という機能もある。また、レイヤ2スイッチに簡易ルータ機能（ルーティング機能）を付加した**レイヤ3スイッチ（L3 スイッチ）**もある。

そのほかに、ルータには、WAN との接続に使用される回線接続装置としての役割もあり、公衆回線に接続するための**ダイヤルアップルータ**などがある。

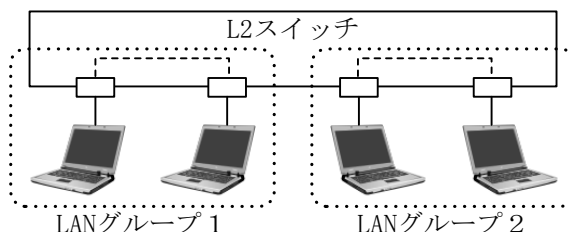
- ・ **ゲートウェイ**

LAN をトランスポート層以上の上位層レベルで接続する装置である。プロトコルが異なる LAN を接続する以外に、WAN などとの接続にも使用される。このとき、LAN 内の PC に代わってインターネットに接続する**PROXY サーバ（プロキシサーバ**又は**代理サーバ）**の役割をもつゲートウェイもある。なお、ゲートウェイには LAN の出入口という意味もあるため、**デフォルトゲートウェイ**には LAN の標準的な出入口となるルータを指定することが多い。

3-2 その他の LAN 技術

(1) VLAN (Virtual LAN)

VLAN は、物理的な接続とは別に形成された仮想的な LAN グループであり、L2 スイッチなどを利用して構築される。通常、1 台の L2 スイッチに接続された機器は、すべて同じ LAN グループ（ブロードキャストドメイン）として扱われるが、VLAN 機能を利用すると複数の LAN グループに分割することができる。



- ・ **ポート VLAN** : L2 スイッチのポートごとに LAN グループを設定する方法
- ・ **タグ VLAN** : 送受信フレームに VLAN タグを付加し、LAN グループを設定する方法

(2) FDDI (Fiber-Distributed Data Interface)

FDDI は、リング型 LAN を発展させた LAN で、基幹ケーブルとして光ファイバケーブルを使用し、100Mビット/秒程度の伝送能力をもつネットワークである。

パケット交換中心の FDDI-I と、音声や画像も伝送できる FDDI-II がある。

(3) 高速イーサネット

高速イーサネット は、高速なデータ伝送を実現したバス型 LAN (Ethernet) である。転送速度が 100Mビット/秒以上の **ファストイーサネット** (100BASE-T, 100BASE-FX) と、1000Mビット/秒以上の **ギガビットイーサネット** (1000BASE-T, 1000BASE-FX) がある。高速イーサネットでは、カテゴリ 5 以上のより対線や光ファイバケーブルを使用する。より対線のカテゴリとは、使用する周波数や信号の衰退などによる分類である。シールド付きのより対線を STP、シールドなしを UTP と表すこともある。また、1000BASE-T は、1 クロックで 2 ビットの情報を伝送する 4 対の信号線で 1 Gビット/秒を実現する。

(4) ATM-LAN

ATM-LAN は、ATM (Asynchronous Transfer Mode ; 非同期転送モード) 技術を使用した高速な LAN である。動画などのマルチメディアデータを、高速に伝送できる。

4 ネットワークの仕組み

現在は、インターネットや LAN が主流であるが、それ以前は、データセンタなどの汎用機（メインフレーム、ホストコンピュータ）に遠隔地の端末を接続する形態が主流であった。この章では、現在までに使用されてきたネットワークの仕組みについて説明する。

4-1 ネットワークの構成要素

光回線を使用する場合の ONU やルータ、アクセスポイント、無線 LAN ルータなどのほかにも、ネットワークを構成するための機器や装置には、次のようなものがある。

- CCU (Communication Control Unit ; 通信制御装置)

送受信時にデータの直並列変換（文字の組立て／分解）や、データに対する誤り制御、複数回線の制御などを行う装置である。単体の機器として利用されることは少なくなったが、機能はサーバや FEP が実現している。

- PBX (Private Branch eXchange ; 構内交換機)

端末装置や電話機を、電話回線などの公衆回線と接続する装置である。企業内での内線電話同士の接続にも使用される。電話やファクシミリをコンピュータシステムに統合して自動応答したり、最適な着信者に振り分けたりする CTI (Computer Telephony Integration) にも利用される。

- DSU (Digital Service Unit ; 宅内回線終端装置)

デジタル回線でデータ伝送を行う場合に使用される DCE である。コンピュータ内のデジタル信号を、伝送に適した形式に変換する。

- NCU (Network Control Unit ; 網制御装置)

公衆電話網などでデータ伝送を行う場合に使用される DCE である。相手と回線を接続するためのダイヤル機能などをもつ。モデム内蔵型が多い。

【端末インタフェース】

端末インタフェースとは、端末間でデータ伝送を行うための、コネクタ（接続部）の規格や信号に関する規定などのことである。

ITU (International Telecommunication Union ; 国際電気通信連合) の下部組織である **ITU-T** (ITU-Telecommunication Standardization Sector ; 国際電気通信連合-電気通信標準化部門) では、端末インタフェースとして次のようなシリーズを ITU-T 勧告としている。

- V シリーズ : アナログ回線用の DTE-DCE 間インタフェース (V. 24 など)
- X シリーズ : デジタル回線用の DTE-DCE 間インタフェース (X. 25 など)

4-2 ネットワークの基礎技術

4-2-1 変調方式

変調とは、デジタル信号をアナログ信号に、又はアナログ信号をデジタル信号に、変換することである（変換した信号を元に戻すことを、**復調**という）。

アナログ回線でデータ伝送を行うために、モデムなどでデジタル信号をアナログ信号に変調する方式としては、次のような種類がある。

- ・ **AM** (Amplitude Modulation ; **振幅変調**)
デジタル信号の0と1を、アナログ信号のON/OFF（波のある／なし）に対応させる。この方式は実現が簡単であるが、雑音に弱いという欠点がある。
- ・ **FM** (Frequency Modulation ; **周波数変調**)
デジタル信号の0と1を、アナログ信号の低周波と高周波に対応させる。この方式は振幅変調方式の次に実現が簡単であり、雑音にも強い。
- ・ **PM** (Phase Modulation ; **位相変調**)
デジタル信号を波形のずれで表現する。デジタル信号の0を通常波形、1を180°ずらして表現する2相位相変調方式や、'00'を通常波形、'01'を90°、'10'を180°、'11'を270°ずらして表現する4相位相変調方式などがある。この方式は、波形のずらし方を細かくすることにより、一つの波形で複数のビットを送信することができるという利点がある。そのため、1秒間に変調される回数を表す**変調速度**（単位：ボ－）と、1秒間に伝送できるビット数を表す**信号速度**（又は回線速度）が同じになるとは限らない。また、1を送信するときに波形を反転する2相差分位相変調方式などもある。

一方、音声信号（アナログ信号）を符号（デジタル信号）化する変調方式としては、**PCM** (Pulse Code Modulation ; **パルス符号変調**)がある。PCMは、標本化／量子化／符号化の手順によって、アナログ信号をデジタル信号に変調する。

【PCMによる符号化手順（P.145より抜粋）】

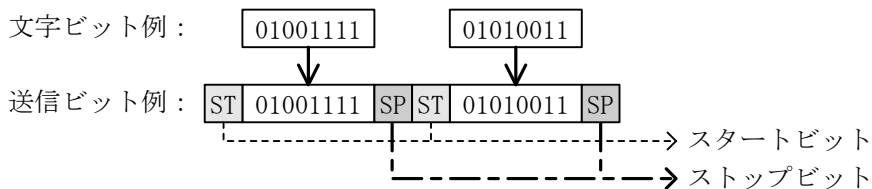
- 1) **標本化**
符号化するアナログ信号を、一定間隔ごとにサンプリングする。この標本間隔（サンプリング間隔）は、**標本化周波数**で示される。
- 2) **量子化**
サンプリングした標本値を、整数値にまとめる。
- 3) **符号化**
量子化した整数値を、2進数で表現することで符号化する（符号として使用するビット数を**量子化ビット数**という）。

4-2-2 同期方式

同期とは、送信側と受信側でタイミングをとることである。代表的な同期方式は、次の3種類である。

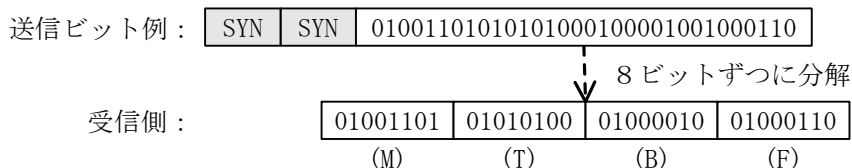
・調歩同期方式（ビット同期方式）

文字データの前後に、スタートビット（1ビット）とストップビット（1～2ビット）を付加して文字の区切りを認識する。送受信のタイミングをとらない（文字単位で判断する）ため、非同期式（非同期方式）とも呼ばれる。



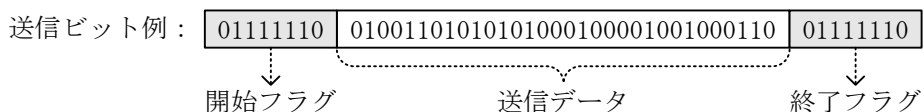
・SYN 同期方式（キャラクタ同期方式）

ブロック（文字データの集まり）の前に同期文字（SYN 符号：00010110）をいくつか（通常は二つ）付加することで、ブロックの始まりを認識する。受信側では、受信ブロックを8ビットごとに区切って文字データとして利用する。そのため、ブロックとして送信するビット数は、文字データの単位で区切れる8ビットの整数倍でなければならないという制約がある。



・フラグ同期方式（フレーム同期方式）

フレーム（ビットの集まり）の前後に、フラグ（01111110）を付加することで、フレームの始まりと終わりのタイミングを知らせる。前後のフラグにより区切りが明確になるので、送信データは文字データ単位（8の倍数）である必要はない。また、フラグ以外の部分で1のビットが5個連続すると、0が強制的に挿入されるため、送信データに制約がない同期方式といえる。



4-2-3 誤り制御方式

誤り制御とは、データ伝送を行う際に通信路上で発生するビット誤りを制御することである。シャノンは、情報伝送における信頼性、効率性、安全性を向上させるために、次の二つの符号化を**符号理論**として提唱している。

(1) 情報源符号化

情報源符号化とは、情報源を符号化して、送信データ量（ビット数）を少なく（**データ圧縮**）することである。通信路などのビット誤り率が固定値であれば、送信ビット数が少ないほうが誤りは発生しにくいという考え方である。

情報源符号化の代表的な方法として、次の2種類がある。

・ **ハフマン符号**

情報源中にある出現頻度の高い情報に短い符号、低い情報に長い符号を割り当てる符号化である。

情報源	A	A	A	A	B	C	C	C	C	C	C
↓											
符号化	10	10	10	10	11	0	0	0	0	0	0

・ **ランレングス符号**

情報源中に同じ情報が繰り返されるラン（連なり）がある場合、そのランを繰り返される情報と長さ（レングス）の組に置き換える符号化である。

情報源	A	A	A	A	B	C	C	C	C	C	C
↓											
符号化	A	*	4	B	C	*	7				

(2) 通信路符号化

通信路符号化とは、情報源とは別の冗長ビットを付加して符号化することである。付加した冗長ビットにより、通信路で発生した誤りを検出・訂正できるという考え方であり、一般的には誤り検出・訂正方式と呼ばれる。

代表的な誤り検出・訂正方式として、次のような種類がある。

	名称	基本用途
誤り検出方式	パリティチェック	奇数個のビット誤りを検出する。
	チェックサム	ブロック単位での誤りを検出する。
	CRC	バースト誤りを検出する。
誤り訂正方式	ハミング符号	1ビットの誤りを訂正する。

① パリティチェック

送信するビット列に検査用冗長ビット（パリティビット）を付加することで、誤りを検出する方式である。冗長ビットの規則性と冗長ビットを付加する対象により、次のように細分化することができる。

【規則性による分類】

- ・ **奇数パリティ**：1の数が奇数になるように、0又は1を付加する。
- ・ **偶数パリティ**：1の数が偶数になるように、0又は1を付加する。

【付加対象による分類】

- ・ **垂直パリティ**：ビット列に対し、垂直（文字）方向に付加する。
- ・ **水平パリティ**：ビット列に対し、水平（ブロック）方向に付加する。
- ・ **水平垂直パリティ**：水平方向と垂直方向の、両方向に付加する。

	A	S	C	I	I	
b ₁	1	1	1	1	1	1
b ₂	0	1	1	0	0	0
b ₃	0	0	0	0	0	0
b ₄	0	0	0	1	1	0
b ₅	0	1	0	0	0	1
b ₆	0	0	0	0	0	0
b ₇	1	1	1	1	1	1
b ₈	0	0	1	1	1	1

水平パリティ

垂直パリティ

図 5-7 水平垂直パリティ（偶数パリティ）の例

パリティチェックでは、奇数個のビット誤りは検出できるが、偶数個のビット誤りは検出できない。また、水平垂直パリティチェック方式を使用した場合、1ビットの誤りと、その位置を特定できる（誤りが起きている列と行の交点）。しかし、複数のビットに誤りが発生している可能性もあるため、パリティチェックには誤り訂正機能をもたせないのが一般的である。

② チェックサム

ブロック単位に簡単な計算により求められる冗長ビット（合計値）を付加することで、誤りを検出する方式である。

【チェックサムの手順】

- 1) 送信側で、送信するデータのブロックごとの合計値を計算する。
- 2) データと合計値（冗長ビット）を送信する。
- 3) 受信側で、受信したデータのブロックごとの合計値を計算する。
- 4) 求めた合計値と受信した合計値を比較し、誤りがないか確認する。

③ CRC (Cyclic Redundancy Check ; 巡回冗長検査)

伝送データを多項式で表現し、あらかじめ決められた生成多項式で伝送データの多項式を除算し、そのときの余りをビット列に変換した検査用符号 (CRC 符号) を付加する方式である。受信側は、多項式で表現した受信データを生成多項式で除算し、割り切れるか否かで誤りの有無を検出する。連続したビット誤り (バースト誤り) を検出することができる、精度の高い誤り検出方式である。

④ ハミング符号

ビットの誤り検出・訂正ができる方式である。ハミング符号のように誤りの訂正を目的とした符号を、ECC (Error Correcting Code ; 誤り訂正符号) と呼ぶ。ECC は、メモリのエラー訂正などにも利用される。

例 1. 伝送データ $(b_4, b_3, b_2, b_1) = (0110)$ のハミング符号を求める。

- 1) 伝送データのビットをグループ化し、各グループで排他的論理和演算などを行って、チェックビット $c_1 \sim c_3$ を求める。

$$c_1 = b_4 \text{ XOR } b_3 \text{ XOR } b_2 = 0 \text{ XOR } 1 \text{ XOR } 1 = 0$$

$$c_2 = b_4 \text{ XOR } b_3 \text{ XOR } b_1 = 0 \text{ XOR } 1 \text{ XOR } 0 = 1$$

$$c_3 = b_4 \text{ XOR } b_2 \text{ XOR } b_1 = 0 \text{ XOR } 1 \text{ XOR } 0 = 1$$

- 2) 伝送データにチェックビットを付加して、ハミング符号を求める。

$$\begin{aligned} \text{ハミング符号} &= (b_4, b_3, b_2, c_1, b_1, c_2, c_3) \\ &= (0110011) \end{aligned}$$

例 2. 受信したハミング符号 (0100011) の誤りビットを訂正する。

- 1) 受信したハミング符号を、ビット単位に分解する。

$$\begin{aligned} \text{ハミング符号} &= (d_7, d_6, d_5, d_4, d_3, d_2, d_1) \\ &= (b_4, b_3, b_2, c_1, b_1, c_2, c_3) \\ &= (0, 1, 0, 0, 0, 1, 1) \end{aligned}$$

- 2) ビットをグループ化し、排他的論理和演算などで $h_1 \sim h_3$ を求める。

$$h_1 = b_4 \text{ XOR } b_3 \text{ XOR } b_2 \text{ XOR } c_1 = 0 \text{ XOR } 1 \text{ XOR } 0 \text{ XOR } 0 = 1$$

$$h_2 = b_4 \text{ XOR } b_3 \text{ XOR } b_1 \text{ XOR } c_2 = 0 \text{ XOR } 1 \text{ XOR } 0 \text{ XOR } 1 = 0$$

$$h_3 = b_4 \text{ XOR } b_2 \text{ XOR } b_1 \text{ XOR } c_3 = 0 \text{ XOR } 0 \text{ XOR } 0 \text{ XOR } 1 = 1$$

- 3) 誤りビット d_n を特定し、ビット誤りを訂正する。

$$n = h_1 \times 4 + h_2 \times 2 + h_3 = 1 \times 4 + 0 \times 2 + 1 = 5$$

$$\text{※誤りビット } d_5 \text{ を訂正する} \rightarrow (010011)$$

ハミング符号で検出・訂正できるビット数は、ハミング距離 (同一ビット長の情報ビットで異なっているビット数のこと) で決まる。例 1 のハミング符号はハミング距離が 3 以上になるため、1 ビットの誤り訂正と 2 ビットの誤り検出が可能である。

4-2-4 交換方式

交換方式とは、**公衆回線**などのように通信相手が特定されていない回線で、どのように相手先にデータを伝送するかという方式である。そのため、通信相手と固定的に直結している**専用線**（専用回線）を提供する**専用線サービス**では、交換方式を考える必要がない。

(1) 回線交換方式

回線交換方式は、公衆電話網と同じような構成で、データ伝送要求が発生するたびに物理的な通信路を回線交換機で設定して、データを伝送する方式である。回線交換方式のネットワークを**回線交換網**といい、このネットワーク内で利用者同士を回線交換機で接続して通信環境を提供するサービスを**回線交換サービス**という。

【回線交換方式の特徴】

- ・高密度／大量データの一括伝送に適している。
- ・送信側、受信側の通信機器は、同一の通信速度でなければならない。
- ・ネットワーク内での伝送遅延は発生しない。

(2) 蓄積交換方式

蓄積交換方式は、送信先に対して物理的な通信路を設定せず、データ単位に付加されているアドレスをもとに、蓄積交換機を経由してデータを伝送する方式である。蓄積交換方式では、通信回線の混み具合などに応じて最適な中継回線を決定する。これは、通信回線上を流れるトラフィック（データ）が多くなりすぎると伝送効率が悪くなり、遅延が発生しやすくなるからである。

【蓄積交換方式の特徴】

- ・低密度／少量データの伝送に適している。
- ・送信側、受信側の通信機器は、同一の通信速度でなくてもよい。
- ・伝送データは蓄積交換機にいったん蓄積されるため、ネットワーク内での伝送遅延が発生する。また、送信順序と受信順序が異なる場合がある。

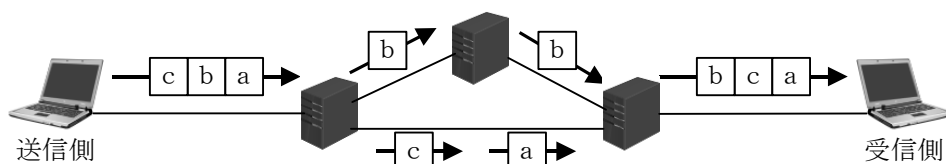


図 5-8 蓄積交換方式のイメージ

蓄積交換方式の代表的な伝送技術／サービスには、次の3種類がある。

① パケット交換（パケット交換サービス）

伝送データを**パケット**という単位に分割して送信する方式である。パケットには、相手先のアドレスなどが記録されたヘッダが付加される。通常は、パケット単位での組立て／分解を行う PAD 機能が内蔵された PT（パケット形態端末）を用いて、パケット交換網に接続する。しかし、NPT（非パケット形態端末）でも、交換機に PAD 機能をもたせれば実現可能である。

【パケット交換の特徴】

- ・パケット単位での誤り制御や送信確認によって、高品質のデータ伝送を行うことができる。ただし、パケット単位の制御であるため、通信速度は低めになる。
- ・通信回線を有効活用できる（中継回線の共有、ルーティング機能）。
- ・論理的な通信回線を複数設定することで、同報通信が可能となる。
- ・相手先が決まっている固定接続（PVC：Permanent Virtual Circuit）と送信相手を選択できる回線接続（VC：Virtual Circuit）がある。

② フレームリレー（フレームリレーサービス）

伝送データを可変長の**フレーム**単位に分割して送信する方式で、パケット交換方式を高速にしたものと考えればよい。フレームには、相手先を表すデータリンクコネクション識別子（DLCI）などを含んだアドレス部が付加される。フレームの組立て／分解を行うフレーム組立分解装置（FRAD）により、フレームリレー網に接続される。

【フレームリレーの特徴】

- ・フレーム単位での誤り制御や送信確認の省略によって、高速なデータ伝送を実現することができる。
- ・CIR（認定速度情報）が設定されるのが一般的である。CIR には、通常状態での最低通信速度を保証する場合と、通信回線が非常に混み合っている状態（輻輳状態）での最低通信速度を保証する場合がある。

③ セルリレー／ATM 交換（ATM サービス）

伝送データを固定長の**セル**単位に分割して送信する方式である。セルリレーでは、**ATM**（Asynchronous Transfer Mode；**非同期転送モード**）技術を用いたハードウェア的なルーティングによって、高速なデータ伝送が実現されている。

【セルリレー／ATM 交換の特徴】

- ・セルはデータ部 48 バイト、ヘッダ部 5 バイトの固定長で構成される。
- ・通信回線が混み合っているときに、優先度の高いセルを選択する機能や、迂回路を設定する機能などによって高速性を維持する。

4-2-5 その他の通信技術

(1) 伝送方式

- ・ **直列伝送（シリアル伝送）**：データを1ビットずつ送信する方式
- ・ **並列伝送（パラレル伝送）**：複数のビットを同時に送信する方式

(2) 通信方式

- ・ **単方向通信**：データを単一方向にしか送信できない通信方式
- ・ **半二重通信**：データを双方向に送信できるが、同時に送受信できない通信方式
- ・ **全二重通信**：データを同時に、双方向に送信できる通信方式

(3) 接続方式

- ・ **ポイントツーポイント方式**（2地点間接続）
コンピュータとコンピュータを、1対1で接続する方式
- ・ **マルチポイント方式**
通信回線に分岐装置を設け、複数のコンピュータを同一回線に接続する方式
- ・ **集線方式**
複数の回線を集線装置により1本に束ねて、遠隔地と接続する方式

(4) 多重化方式

- ・ **FDM** (Frequency Division Multiplex ; 周波数分割多重化方式)
アナログ回線で使用する多重化方式。使用する周波数帯域を変えることにより、1回線で複数のデータを同時に伝送する。
- ・ **TDM** (Time Division Multiplex ; 時分割多重化方式)
デジタル回線で使用する多重化方式。一定時間（タイムスロット）ごとに接続先を変えることにより、細切れのデータを同時に伝送する。
- ・ **WDM** (Wavelength Division Multiplex ; 波長分割多重化方式)
光ファイバケーブルで使用する多重化方式。データ伝送に使用する光の波長を変えることにより、1回線で複数のデータを同時に伝送する。
- ・ **CDM** (Code Division Multiplex ; 符号分割多重化方式)
移動体通信（携帯電話など）で使用する多重化方式。利用者ごとにコードを割り当てることで、同一周波数帯域中を流れる情報を区別して伝送する。

4-3 伝送制御手順

伝送制御とは、データ伝送を効率良く、正確に行うための制御のことである。伝送制御では、次のような制御が行われている。

- ・ **回線制御**
通信回線の物理的な接続／切断を行う。
- ・ **同期制御**
送受信間で同期をとったり、転送速度の制御（**フロー制御**）を行う。
- ・ **誤り制御**
誤りデータの検出（再送指示）や訂正を行う。
- ・ **データリンク制御**
データリンク（論理的な通信路）の確立／終結を行う。
- ・ **ルーティング制御**
データ伝送を行うための最適な通信経路の選択を行う。

データ伝送を行う場合、伝送制御は次の各段階（フェーズ）を経て実行される。

- 1) フェーズ1 <通信回線の接続>
通信回線を物理的に接続する。（専用線では不要）
- 2) フェーズ2 <データリンクの確立>
送信相手と呼び出し、論理的な通信路を接続する。
- 3) フェーズ3 <データ伝送>
同期をとり、データを伝送する。同時に伝送データの誤り制御も行う。
- 4) フェーズ4 <データリンクの終結>
送信相手とデータ伝送終了を確認し、論理的な通信路を閉じる。
- 5) フェーズ5 <通信回線の切断>
物理的に接続した通信回線を切断する。（専用線では不要）

伝送制御手順とは、これらの制御を効率良く行うための手順である。ただし、フェーズ1とフェーズ5は、専用線では不要なフェーズであるため、伝送制御手順ではフェーズ2～4を規定している。

なお、フェーズ2～4のように、データリンクを確立してから通信を行う方式を**コネクション方式**という。これに対して、データリンクを確立しないで通信を行う方式を**コネクションレス方式**という。

また、一つのデータリンクを利用して通信を行う方式を**シングルリンク手順**というのに対して、並行する複数のデータリンクを、あたかも一つのデータリンクであるかのようにまとめて利用して、通信を行う方式を**マルチリンク手順**という。

4-3-1 無手順 (TTY 手順)

無手順 (TTY 手順) は、伝送制御手順を考慮しないで、人間がデータ伝送に関する制御を指定する方式である。最低限の伝送制御しか行えないため、低速回線で使用される。

同期制御	調歩同期方式 (ビット同期方式)
誤り制御	パリティチェック (垂直パリティ)

4-3-2 ベーシック手順 (基本型データ転送手順)

ベーシック手順 (基本型データ転送手順) は、データをブロック単位で伝送し、伝送制御キャラクタを利用して制御を行う方式である。ベーシック手順は、相手の返答を待つて次の処理を行う会話型の伝送制御手順である。

同期制御	SYN 同期方式 (キャラクタ同期方式)
誤り制御	パリティチェック (水平垂直パリティ)

(1) 伝送制御キャラクタ

ベーシック手順では、次の 10 個の伝送制御キャラクタを利用する。伝送制御キャラクタは、文字コードの 0 列と 1 列で定義されている特殊符号である。

コード (列/行)	符号	意味
(01)	SOH	ヘディング開始
(02)	STX	テキスト開始
(03)	ETX	テキスト終了
(04)	EOT	伝送終了
(05)	ENQ	問合せ
(06)	ACK	肯定応答
(10)	DLE	伝送制御拡張
(15)	NAK	否定応答
(16)	SYN	同期信号
(17)	ETB	伝送ブロック終了

(2) 伝送メッセージ

ベーシック手順では、情報をメッセージという単位で取り扱う。このメッセージを分割したものがブロックであり、このブロック単位に情報を伝送する。

SOH	ヘディング	STX	テキスト	ETX

図 5-9 ブロックのイメージ

(3) データリンクの確立

ベーシック手順でデータリンクを確立する方法には、次の二つの方式がある。

- ・ **コンテンション方式**
ポイントツーポイント方式で接続されている場合の、データリンク確立方式である。送信側は、相手先に対して、送信要求（ENQ）を出して、受信可能かどうかの問合せを行う。受信側から肯定応答（ACK）が送られてくれば、データリンクが確立されてデータを送信する。否定応答（NAK）が送られてきた場合は、データリンクが確立するまで問合せ（送信要求）を繰り返す。
- ・ **ポーリング／セレクトイング方式**
マルチポイント方式で複数の端末が同一回線に接続されている場合の、データリンク確立方式である。通信を管理するコンピュータ（制御局）が、各端末（従属局）に対して「送信要求はないか」を問い合わせ（ポーリング）、要求がある場合には送信を許可する。また、端末に「受信可能か」を問い合わせ（セレクトイング）、受信可能という応答があれば、その端末に送信する。

4-3-3 HDLC 手順（ハイレベルデータリンク制御手順）

HDLC 手順（HDLC：High-level Data Link Control；ハイレベルデータリンク制御）は、データをフレーム単位で伝送する、高速伝送を行う場合の伝送制御手順である。

同期制御	フラグ同期方式（フレーム同期方式）
誤り制御	CRC（巡回冗長検査）

(1) フレームの形式

HDLC 手順では、次の形式のフレームが伝送単位となる。

F	A	C	I	FCS	F

- F：フラグシーケンス（8ビット）
フレームの先頭と末尾に付ける開始／終了フラグ（01111110）である。
- A：アドレスフィールド（8ビット）
フレームの送信局又は受信局のアドレスが記録される。
- C：制御フィールド（8ビット）
フレームの種類や送信順序が記録される。
- I：情報フィールド（任意のnビット）
伝送データが記録される。
- FCS：フレームチェックシーケンス（16ビット）
A，C，Iのフィールドを対象とした誤り制御用のCRC符号が記録される。

(2) フレームの種類

HDLC 手順で利用されるフレームは、その目的により次の三つに分類できる。

フレームの種類	役割
I フレーム（情報伝送形式）	情報を伝送するのに使用する。
S フレーム（監視形式）	データリンクの監視や、受信確認／再送要求などの制御に使用する。情報フィールドがない。
U フレーム（非番号性形式）	モード設定や異常状態の報告などに使用する。情報フィールドがある場合と、ない場合がある。

(3) データリンクの確立

HDLC 手順では、接続するコンピュータ（端末）を局と呼ぶ。局の種類によって、データリンクの確立方式が異なる。

・不平衡型データリンク

データリンクの確立などを制御する一次局と、管理される二次局の間で行われるデータリンクの確立方式である。一次局がすべての制御を行うため、ポーリング／セレクトイング方式とほぼ同様の手順となる。

・正規応答モード（NRM : Normal Response Mode）

一次局から二次局に送信許可（コマンド）フレームが届いたときに限り、二次局は一次局に対して応答（レスポンス）フレームを送れる。

・非同期応答モード（ARM : Asynchronous Response Mode）

一次局から二次局に送信許可（コマンド）フレームが届かなくても、二次局が自由に応答（レスポンス）フレームを送れる。

送信フレームの種類	アドレスフィールドに設定される値
コマンド	受信局（二次局）のアドレス
レスポンス	送信局（二次局）のアドレス

・平衡型データリンク

一次局と二次局の機能を兼ね備えた複合局における、データリンクの確立方式である。どちらからもコマンド／レスポンスを送れるため、コンテンション方式とほぼ同様の手順となる。

・非同期平衡モード（ABM : Asynchronous Balanced Mode）

送信相手からの送信許可がなくても、コマンドフレーム／レスポンスフレームのどちらでも送信できる。

HDLC 手順では、データリンクを確立した後、制御フィールドの情報（送信順序番号／受信順序番号）を使用することにより、情報（フレーム）の連続転送が可能となる。

4-4 IoT 関連技術

IoT (Internet of Things ;モノのインターネット) は、世の中に存在するさまざまなものに通信機能をもたせ、インターネットなどを介して相互通信する仕組みである。IoT 機器などから収集した膨大なデータは、ビッグデータなどとしても活用される。

IoT では、次のようなネットワーク関連技術が利用されている。

- ・ **センサネットワーク**

センサを取り付けた複数の端末（機器）を接続し、さまざまなデータを採取することを目的としたネットワークである。センサネットワークは、誰でも、いつでも、どこでも通信できる **ユビキタスコンピューティング** を実現する基盤技術の一つでもある。

- ・ **LPWA** (Low Power Wide Area)

通信速度は低速（数 k ビット／秒～数百 k ビット／秒程度）であるが、一般的に用いられる電池でも数年～数十年は使用できる省電力性や、数 km～数十 km の遠距離通信を可能とする広域性などの特徴をもつ無線通信方式である。

- ・ **BLE** (Bluetooth Low Energy)

省電力での通信を可能にした Bluetooth の拡張仕様である。Bluetooth4.0 の規格の一部であり、低消費電力モードとも呼ばれる。通信速度は低速（最大 1 Mビット／秒）であるが、小さなボタン電池でも数年間は稼働できるため、IoT 機器やスマートデバイスなどに活用されている（代表的な活用例としては、発信機の位置情報を受信機側で特定するための **BLE ビーコン** などがある）。

- ・ **MQTT** (Message Queuing Telemetry Transport)

IoT システムなどで利用される、メッセージ交換用プロトコルである。送信するヘッダ情報などのデータ量が、一般的なプロトコル（HTTP など）に比べて非常に小さいことから、軽量プロトコルとも呼ばれる。

- ・ **エッジコンピューティング**

IoT 機器や端末などのすぐ近くに、サーバなどの制御用機器を配置して処理する方式である。そのため、高速でリアルタイムな制御を実現できる。元々は、インターネット経由でサービスの提供を受けるクラウドコンピューティングの対義語として使用されていたが、エッジの捉え方によってクラウドコンピューティングとの境界があいまいになっている。

- ・ **IoT エリアネットワーク**

IoT デバイス（IoT 機器など）と IoT ゲートウェイ（IoT デバイスをインターネットなどのネットワークに接続するための装置）間のネットワークのことである。IoT デバイスの通信距離が比較的短い場合、IoT ルータを利用して中継することなどもある。

5 ネットワーク管理

ネットワーク管理とは、ネットワークの運用を管理することである。ネットワーク管理の目的は、ネットワークの信頼性・安全性・効率性を高めることである。

5-1 ネットワーク運用管理

ネットワーク運用管理では、次の五つの管理を重点的に実施する。

① 構成管理

ネットワークを構成する資源（機器など）の情報を、収集／管理する。ネットワークの構成変更及びバージョン管理なども、この管理に含まれる。

② 障害管理

ネットワーク上で発生した障害に関する情報を収集し、障害対策や原因究明を行い、障害の再発防止／予防などに役立てる。

【一般的な障害対応手順】

1) 障害の検出

ネットワークの情報を収集して、障害を発見する。

2) 障害の分析

発見した障害の内容を分析して切分け（分類）を行い、障害の原因を特定する。

3) 障害の対応

障害原因を取り除くための復旧措置を実施する。また、障害に関する情報（障害内容、障害原因、対応策など）を記録し、障害の再発防止や予防に活用する。

③ セキュリティ管理

ネットワークの利用状況（アクセス状況）を監視して、不正アクセスによる情報漏えいや不正使用を防止する。

④ 性能管理

ネットワーク上のトラフィック量を監視したり、データの転送時間や処理のレスポンスタイム（応答時間）を計測したりすることで、ネットワークの性能情報を収集／管理する。このとき、ネットワークの性能が基準値より低下していれば、ネットワークの性能を維持するための対策（回線の増設や機器の交換など）を実施する。

⑤ 課金管理

ネットワークの利用状況（アクセス状況）を収集して、利用者に対する課金情報を管理する。

5-2 ネットワーク管理手法

(1) ネットワーク管理ツール

ネットワーク管理ツールは、実際にネットワークの情報を収集したり、収集した情報を解析したりする。代表的なネットワーク管理ツールには、次のようなものがある。

- ・ **LAN アナライザ**

LAN 上のパケットやトラフィックを監視したり、故障を診断したりする機器やソフトウェアである。監視や故障診断などのために、パケットを解析／表示できる機能を備えたものもあり、悪用されないように注意する必要がある。

- ・ **ネットワーク管理用コマンドプログラム**

コマンドを入力してプログラムを実行することで、ネットワークの状況などを確認できるツールである。

- ・ **ping**

機器の接続状況の確認（疎通確認）やレスポンスを測定／診断する。接続状況を確認したいネットワーク機器の IP アドレスを指定してメッセージを送信し、相手から返信をもらうことで確認する。

- ・ **ifconfig／ipconfig**

IP アドレスなどのネットワーク設定状況を確認する。

- ・ **arp**

ARP テーブル（IP アドレスと MAC アドレスの対応表）を確認する。

- ・ **netstat**

ネットワークに関する各種統計情報を確認する。

(2) SNMP (Simple Network Management Protocol)

SNMP は、TCP/IP の簡易ネットワーク管理プロトコルである。**SNMP 管理ステーション**で動作する**マネージャ**と管理対象で動作する**エージェント**間で、情報の収集／設定、管理対象からの割込みなどのやり取りを行うことでネットワークを管理する。管理対象機器は、自らの情報を**MIB** (Management Information Base ; **管理情報ベース**) として提供する。

(3) ネットワーク OS

ネットワーク OS は、ネットワーク管理に必要となるファイル、プリンタの共有機能やセキュリティ機能を提供する。従来の OS には、このような機能がなかったために、ノベル社が開発した **NetWare** などのネットワーク OS が利用されていた。現在では、これらの機能をもつ OS が一般的であり、ネットワーク OS の概念はなくなっている。

第5部 演習問題

問1

携帯電話端末の機能の一つであるテザリングの説明として、適切なものはどれか。

- ア 携帯電話端末に、異なる通信事業者のSIMカードを挿して使用すること
- イ 携帯電話端末をモデム又はアクセスポイントのように用いて、PC、ゲーム機などから、インターネットなどを利用したデータ通信をすること
- ウ 契約している通信事業者のサービスエリア外でも、他の事業者のサービスによって携帯電話端末を使用すること
- エ 通信事業者に申し込むことによって、青少年に有害なサイトなどを携帯電話端末に表示しないようにすること

問2

64kビット／秒の回線を用いて 10^6 バイトのファイルを送信するとき、データ転送におよそ何秒かかるか。ここで、回線利用率は80%とする。

- ア 19.6 イ 100 ウ 125 エ 157

問3

RADIUS サーバの提供するサービスとして、適切なものはどれか。

- ア FQDN を IP アドレスに変換する。
- イ 外部ネットワークに代理接続する。
- ウ 公衆網を経由してネットワークにリモートアクセスする。
- エ システム全体の認証を一元化する。

問4

SMTP の説明として、適切なものはどれか。

- ア Web サーバに格納されている情報を検索するプログラムである。
- イ Web サーバに格納されている情報を転送するためのプロトコルである。
- ウ ファイルサーバからファイルをダウンロードするためのプロトコルである。
- エ メールサーバに電子メールを転送するためのプロトコルである。

問5

Web サーバに対するアクセスがどの PC からのものであるかを識別するために、Web サーバの指示によってブラウザにユーザ情報などを保存する仕組みはどれか。

- ア CGI イ cookie ウ SSL エ URL

問6

音声情報を符号化し、インターネットで利用されるパケット形式での送受信を実現する技術はどれか。

- ア EDI イ PIAFS ウ VAN エ VoIP

問7

OpenFlow を使った SDN (Software-Defined Networking) の説明として、適切なものはどれか。

- ア RFID を用いる IoT (Internet of Things) 技術の一つであり、物流ネットワークを最適化するためのソフトウェアアーキテクチャ
イ 音楽や動画、オンラインゲームなどの様々なソフトウェアコンテンツをインターネット経由で効率的に配信するために開発された、ネットワーク上のサーバの最適配置手法
ウ データ転送と経路制御の機能を論理的に分離し、データ転送に特化したネットワーク機器とソフトウェアによる経路制御の組合せで実現するネットワーク技術
エ データフロー図やアクティビティ図などを活用して、業務プロセスの問題点を発見し改善を行うための、業務分析と可視化ソフトウェアの技術

問8

TCP/IP で利用されるパケットのイーサネットヘッダに含まれる情報はどれか。

- ア IP アドレス イ MAC アドレス
ウ シーケンス番号 エ ポート番号

問9

クラスCのIPアドレスとして、コンピュータに付与できるものはどれか。

ア 192.0.0.255

イ 192.0.256.16

ウ 192.128.0.0

エ 192.128.0.128

問10

IPv4 において、インターネット接続用ルータの NAT 機能の説明として、適切なものはどれか。

ア インターネットへのアクセスをキャッシュしておくことによって、その後に同じ IP アドレスの Web サイトへアクセスする場合、表示を高速化できる機能である。

イ 通信中の IP パケットを検査して、インターネットからの攻撃や侵入を検知する機能である。

ウ 特定の端末宛ての IP パケットだけを通過させる機能である。

エ プライベート IP アドレスとグローバル IP アドレスを相互に変換する機能である。

問11

ネットワークに接続したコンピュータに、IP アドレスを動的に割り当てるプロトコルはどれか。

ア DHCP

イ DNS

ウ NAPT

エ RIP

問12

OSI 基本参照モデルにおけるネットワーク層の説明として、適切なものはどれか。

ア エンド・ツー・エンド間の透過的なデータ伝送を実現するために、ルーティングやデータの転送／中継機能などを提供する。

イ 物理的な通信媒体の特性の差を吸収し、機械的・電子的な規格などを定めることによって上位層に透過的な伝送路を提供する。

ウ 最も利用者（アプリケーションプログラム）に近い部分で、ファイル転送や電子メールなどの機能を提供する。

エ 隣接するノード間での透過的なデータ伝送を実現するために、伝送制御手順（誤り検出、再送制御など）を提供する。

問13

インターネットで使われるプロトコルである TCP 及び IP と、OSI 基本参照モデルの 7 階層との関係を適切に表しているものはどれか。

	ア	イ	ウ	エ
トランスポート層	IP		TCP	
ネットワーク層	TCP	IP	IP	TCP
データリンク層		TCP		IP

問14

LAN におけるメディアアクセス制御 (MAC) のうち、伝送媒体上でのデータフレーム衝突を検出する機能をもつ方式はどれか。

ア CSMA/CA	イ CSMA/CD
ウ トークンバス	エ トークンリング

問15

LAN 間接続装置に関する記述のうち、適切なものはどれか。

- ア ゲートウェイは、OSI 基本参照モデルにおける下位層（第 1 層～第 3 層）のプロトコル変換に使用される。
- イ ブリッジは、IP アドレスを基にしてフレームを中継する。
- ウ リピータは、信号を増幅することによって伝送距離を延長する。
- エ ルータは、MAC アドレスによるルーティング機能を提供する。

問16

調歩同期方式のデータ伝送において偶数パリティによる誤り検出方式を用いて、文字 “T” (JIS 7 単位符号：1010100) を送出した。この場合、受信側で正しく受信できたときのビット列はどれか。ここで、送出はスタートビット (0)、文字の低位ビットから高位ビット、パリティビット、ストップビット (1) の順とし、受信したビットを左から順に記してある。

ア 0001010101	イ 0001010111
ウ 1001010110	エ 1001010111

問17

送信側では、ビット列をある生成多項式で割った余りをそのビット列に付加して送信し、受信側では、受信したビット列が同じ生成多項式で割り切れるか否かで誤りの発生を判断する誤り検査方式はどれか。

ア CRC 方式	イ 垂直パリティチェック方式
ウ 水平パリティチェック方式	エ ハミング符号方式

問18

ATM 交換機に関する記述として、適切なものはどれか。

- ア 事業所などの限られた範囲の構内に設置された内線電話機相互間の接続や、加入者電話回線と内線電話機との接続に用いる構内交換機の総称である。
- イ 伝送データをセルと呼ばれる固定長のブロックに分割し、各セルに宛先情報を含むヘッダを付加することによって、種々のデータを統一的に扱う交換機である。
- ウ 伝送データをブロック化された単位に区切って転送する蓄積型の交換機であり、伝送速度は数十kビット／秒程度までである。
- エ フレームと呼ばれる単位に区切られたデータを交換する交換機であり、伝送誤りに対する再送を行わないので、ネットワーク内の処理を高速化することができる。

問19

HDLC 手順で伝送されるフレームにある FCS の役割として、適切なものはどれか。

- ア フレームの誤り制御用符号を記録する。
- イ フレームの開始／終了を識別する符号を記録する。
- ウ フレームの種類や送信順序を記録する。
- エ フレームの送信局／受信局を識別する情報を記録する。

問20

ネットワーク管理ツールの一つである ping の役割として、適切なものはどれか。

- ア IP アドレスと MAC アドレスの対応表を確認する。
- イ IP アドレスなどのネットワーク設定状況を確認する。
- ウ IP アドレスを指定したメッセージを送信して、機器の接続状況を確認する。
- エ LAN 上のパケットやトラフィックの状態を確認する。

第6部

セキュリティ

1 情報セキュリティの概要

情報システムやインターネットが社会基盤となっている現代社会では、IT への依存が高まるにつれて、情報セキュリティに対する重要性も増大している。情報セキュリティで実施される対策は、技術的対策と管理（マネジメント）的対策に大別されるので、この章では技術・管理の両面から情報セキュリティ全般について説明する。

1-1 情報セキュリティの概念

JIS Q 27000:2019（情報セキュリティマネジメントシステム—用語）では、情報セキュリティを「情報の**機密性**、**完全性**及び**可用性**を維持すること。さらに、**真正性**、**責任追跡性**、**否認防止**、**信頼性**などの特性を維持することを含めることもある。」と規定している（機密性、完全性、可用性の頭文字を並べて“情報セキュリティの CIA”とも呼ぶ）。

機密性 (Confidentiality)	認可されていない個人、エンティティ又はプロセスに対して、情報を使用させず、また、開示しない特性
完全性 (Integrity)	正確さ及び完全さの特性
可用性 (Availability)	認可されたエンティティが要求したときに、アクセス及び使用が可能である特性
真正性 (Authenticity)	エンティティは、それが主張するとおりのものであるという特性
責任追跡性 (Accountability)	利用者やアクセスなどが一意に追跡できる特性 注記：JIS Q 27000:2019 では、用語として定義されていない。
否認防止 (Non-Repudiation)	主張された事象又は処置の発生、及びそれらを引き起こしたエンティティを証明する能力
信頼性 (Reliability)	意図する行動と結果とが一貫しているという特性

情報セキュリティでは、情報の機密性、完全性及び可用性を維持するため、三つの管理対象と五つの管理機能に着目する。

【情報セキュリティの管理対象】

資産	組織にとって価値をもつもの
脅威	システム又は組織に損害を与える可能性がある、望ましくないインシデント（望まない又は予期しない事象）の潜在的な原因
脆弱性	一つ以上の脅威によって付け込まれる可能性のある、資産又は管理策の弱点

【情報セキュリティの管理機能】

予防機能	脅威の発生を未然に防ぐ機能
抑止機能	脅威を発生させる行為者をけん制する機能（予防機能の一種）
検知機能	発生した脅威を発見・検出する機能
最小化機能	発生した脅威の被害を最小限に抑え、拡大を防止する機能
復旧機能	脅威による被害を速やかに復旧する機能

情報セキュリティの概念を理解するために、情報セキュリティの管理対象について、順番に詳細を説明する。

【OECD セキュリティガイドライン】

OECD（Organization for Economic Cooperation and Development；経済協力開発機構）が、参加者（情報システムの所有者、提供者、利用者）に対し、九つの原則を勧告している **OECD セキュリティガイドライン**（情報システム及びネットワークのセキュリティのためのガイドライン）である。

- ① **認識**：参加者は、情報システム及びネットワークのセキュリティ（以下、単にセキュリティと略す）の必要性、並びにセキュリティを強化するために自己が貢献できることを認識すべきである。
- ② **責任**：参加者は、セキュリティに責任を負うべきである。
- ③ **対応**：参加者は、セキュリティの事件に対する予防、検出及び対応のため、適切な時期に協力的に行動すべきである。
- ④ **倫理**：参加者は、他者の合法的な利益を尊重すべきである。
- ⑤ **民主主義**：セキュリティは、民主主義の本質的価値に適合すべきである。
- ⑥ **リスクアセスメント**：参加者は、情報システム及びネットワークのリスクアセスメントを実践すべきである。
- ⑦ **セキュリティの設計及び実装**：参加者は、情報システム及びネットワークの本質的な要素として、セキュリティを組み込むべきである。
- ⑧ **セキュリティマネジメント**：参加者は、情報システム及びネットワークに対して、総合的セキュリティマネジメントアプローチを採用すべきである。
- ⑨ **再評価**：参加者は、セキュリティのレビュー及び再評価を行い、セキュリティの方針、実践、手段及び手続に適切な修正をすべきである。

なお、OECD では「**OECD プライバシーガイドライン**」として、個人情報保護に関する8原則（収集制限の原則、データ内容の原則、目的明確化の原則、利用制限の原則、安全保護の原則、公開の原則、個人参加の原則、責任の原則）も勧告している。

1-1-1 情報セキュリティの管理対象

(1) 資産 (asset)

資産とは、組織にとって守るべき価値のあるものである。このうち、重要な情報に関わる資産のことを、特に**情報資産**という。

【情報資産の種類】

・有形情報資産

ハードウェア（コンピュータや通信機器など）のような物理的資産や、紙・ドキュメント類などのソフトウェア資産、及び人的資産など

・無形情報資産

顧客情報、営業情報、知的関連財産情報、人事情報などの各種データ、OS・アプリケーション類、組織の評判・イメージ、サービスなど

(2) 脅威 (threat 又は peril)

脅威とは、情報資産に損失を与える可能性があるものである。脅威は、**物理的脅威**、**人的脅威**、**技術的脅威**の三つに分類される。

① 物理的脅威

機器自体や機器が設置されている建物に対する脅威である。物理的脅威の代表的な例は、次のとおりである。

・災害

地震、洪水などの自然災害や火災などの人的災害によって、機器や建物が壊れたり、機器自体がなくなってしまうりすることである。

・事故／故障

偶発的な事故や故障などによって、機器や建物が使えなくなることである。

・破壊

悪意のある第三者によって、機器や建物が壊されることである。

・盗難

情報が保存されている PC や USB メモリなどが盗まれることであり、情報が外部に漏れる**情報漏えい**が起きる危険性もある。

・不正侵入

機器が設置されている建物や室に、権限のない者が侵入することであり、破壊や盗難などが起きる場合もある。

② 人的脅威

人間の行為（悪意の有無を問わない）が原因の脅威である。人的脅威の代表的な例は、次のとおりである。これらのうち、不正であることを認識した上の行為のことを**不正行為**といい、特に組織内の人間（社員など）が行う不正は**内部不正**と呼ばれる。内部不正による**情報セキュリティインシデント**（情報セキュリティを阻害する事象）は、組織の信用失墜などの大きな影響を与えることも多い。

- ・ **情報漏えい**

情報を第三者に漏らすことである。情報提供料を目当てにした意図的な情報漏えい（故意）や、重要な情報を第三者に偶然、聞かれる過失、重要な情報でないと誤謬して話す意図しない情報漏えいなどがある。

- ・ **紛失／破損**

情報が保存されている PC や USB メモリなどを、どこかに置き忘れて失くしたりたり、使用中に壊したりすることである。

- ・ **誤操作**

操作を間違えて、データを消去したりすることである。電子メールの宛先を間違えて、重要な情報を漏えいさせることなども含まれる。

- ・ **不正利用**

権利をもっていない利用者が、不正に利用することである。他人のクレジットカードの不正利用などが該当する。

- ・ **ソーシャルエンジニアリング**

日常的、一般的な手段で情報を盗み取る行為である。

- ・ **トラッシング（スキャベンジング、ダンプスターダイビング）**

ごみ箱に捨てられたメモや書類などから、情報を探し出すこと

- ・ **なりすまし**

第三者があたかも本人であるかのように装い、電話などで暗証番号やパスワードを聞き出すこと

- ・ **盗み見**

他人のキーボード操作や画面に表示された情報などを、気づかれないように見て盗み出すこと（肩越しからこっそり見て盗み出す行為は、**ショルダハッキング**という）

- ・ **SNS の悪用**（SNS : Social Networking Service）

コミュニケーションツールである SNS を悪用して、個人情報や不正に取得したり、他人のアカウントを乗っ取ったりすることである。

なお、人的脅威に代表される不正行為は、機会・動機・正当化の3要素（**不正のトライアングル**）がそろくと発生しやすいため、不正の起きやすい状況を改善する**状況の犯罪予防**が重要となる。

③ 技術的脅威

悪意のある第三者が、コンピュータ技術を利用して攻撃する脅威、又はコンピュータの技術的な原因による脅威である。技術的脅威の代表的な例は、次のとおりである。

- ・ **不正アクセス**

アクセス権をもっていない利用者がシステムやデータを不正に使用する、又は不正に使用できる状態にすることである。

- ・ **盗聴**（悪意の**スニッフィング**）

通信データなどが傍受されることである（善意のスニッフィングは、パケット解析などの目的で行われる）。

- ・ **改ざん**

電子メールや Web ページが不正に書き換えられることである。

- ・ **なりすまし**

別の人物（正当な利用者など）の ID やパスワードなどを使い、別の人物のふりをしてシステムなどを不正に使用することである。

- ・ **窃取**

保存されているファイルやデータが盗まれることである。

- ・ **破壊**

保存されているファイルやデータが破壊されることである。

- ・ **クラッキング**

悪意をもって他人の PC やシステムなどに侵入（不正アクセス）して、データを盗み見たり、破壊したりする行為である。クラッキングを行う人を**クラッカ**という。なお、クラッカが侵入した後に利用するプログラムやツールなどをまとめたパッケージを **rootkit**（ルートキット）、次回以降の侵入に利用するために設置する経路を**バックドア**（裏口）という。また、悪意のある**ハッキング**（システムにアクセスして、構造解析や改変をする行為）を行うクラッカのことを**ブラックハット**と呼んで、善意のハッキングを行う**ホワイトハット**又は**ホワイトハッカー**とは区別することもある。

- ・ **サービス妨害**

他者が提供しているサービスを、提供できなくする妨害行為である。

- ・ **エラー**

システムやアプリケーションの誤りなどにより、ファイルやデータが消されたりすることである。

- ・ **危殆化**

コンピュータ技術が向上することで、安全性が低下（例えば、技術の向上によってパスワードの解読時間が短縮）することである。

また、クラッカのような攻撃者は動機などによって、次のように分類される。

【攻撃者の種類】

・愉快犯

世間を騒がせることを目的とする攻撃者である。インターネットなどに公開されているクラッキングツールを利用して興味本位でクラッキングを行うスクリプトキディなどは、愉快犯に分類される場合が多い。

・詐欺犯

相手をだまし、金銭を奪取することなどを目的とする攻撃者である。

・故意犯

自身の行為が正当でないことを理解していながら、不正行為を行う攻撃者である。不正行為であることを認識しながら内部不正を行う企業などの内部関係者は、故意犯とみなすことができる。

・確信犯（ハクティビストやサイバーテロリストなど）

自分の行動は正しいと信じて、**ハクティビズム**（社会／政治的な主義・主張のもとにハッキング活動を行うこと）や**サイバーテロリズム**（インターネットなどを利用した大規模な破壊活動）を行う攻撃者である。なお、違法取引や違法活動には**ダークウェブ**（アクセスに特定の認証などが必要なダークネットに存在する Web コンテンツ）が用いられることが多い。

(3) 脆弱性 (vulnerability 又は hazard)

脆弱性とは、脅威につけ込まれて、さらに脅威が拡大する要因となる、資産の弱点や欠陥である。設備面、技術面、管理面など、さまざまな脆弱性が問題となる。

・セキュリティホール

ソフトウェアの設計上の欠陥やバグなどによって生じる、ソフトウェア又はシステムの脆弱性のことである。セキュリティホールを作らないためには、情報セキュリティを企画・設計段階から確保するための方策である**セキュリティバイデザイン**が重要となってくる。

・人為的脆弱性

企業・組織・個人に対する行動規範の不徹底／未整備などによって生じる、人間の行為が原因となる脆弱性のことである。私物の携帯端末を会社の許可を得ずに業務に利用する**シャドー IT**なども人為的脆弱性に該当する。

脆弱性については、**脆弱性検査ツール**（VS : Vulnerability Scanner）などを用いて検査したり、**CVSS**（Common Vulnerability Scoring System ; 共通脆弱性評価システム）などを使って評価したりすることで、適切に管理する必要がある（**脆弱性管理**）。

1-1-2 マルウェア

マルウェアは、悪意をもって作成された不正プログラムの総称である。マルウェアも、技術的脅威に分類される。代表的なマルウェアは、次のとおりである。

・コンピュータウイルス

経済産業省「コンピュータウイルス対策基準」では，“第三者のプログラムやデータベースに対して意図的に何らかの被害を及ぼすように作られたプログラムであり，次の機能を一つ以上有するもの”と定義している。

自己伝染機能	自らをコピーして，他のシステムに伝染する。
潜伏機能	発病するまで症状を出さない。
発病機能	データの破壊など，設計者の意図しない動作をする。

しかし，現在は一般的に，特定のファイルに感染するファイル感染型ウイルスを，コンピュータウイルスと呼んでいる（狭義のウイルス）。

・プログラムファイル感染型ウイルス

アプリケーションなど，実行可能なプログラムファイルに感染する。

・ブートセクタ感染型ウイルス

OS の起動前に読み込まれるブートセクタ（ブートプログラムがあるシステム領域）に感染する。

・インタプリタ型ウイルス

データファイルなど，実行可能なプログラムファイル以外に感染する。アプリケーションソフトのマクロ機能を利用して感染する**マクロウイルス**や，JavaScript，VBScript などのスクリプト言語を利用して感染するタイプの**スクリプトウイルス**がある。

・偽セキュリティ対策ソフト型ウイルス

正規のセキュリティ対策ソフトに見せかけて，“ウイルスに感染している”，“ハードディスクに異常が検出された”などの警告メッセージを表示し，それらを解決するためとして有償版製品を購入させようとしたり，罠のある Web サイトに誘導してクレジットカード番号などを入力させることで不正取得を試みたりする。

・ワーム

感染対象のプログラムを必要とせず，自分自身の複製をネットワーク経由でほかのコンピュータにコピーして増殖する。電子メールの添付ファイルとして自動的に自分自身のコピーを拡散させるものや，ネットワークを利用して次々に感染していくものが多い。

・SQL Slammer

市販 DBMS の脆弱性（セキュリティホール）を利用することにより，宿主となるデータベースサーバを探して自己伝染を繰り返し，インターネットなどのトラフィックを急増させるワームの一種である。

- ・ **ボット** (bot)

感染したコンピュータを、ネットワーク（インターネット）を通じて外部から操る**遠隔操作型ウイルス**である。ボットを送り込まれたコンピュータ（ゾンビマシン）で構成されるボットネットを、**C&C サーバ**（C&C : Command and Control）などで統制する攻撃者を**ボットハーダ**と呼ぶ。

- ・ **スパイウェア**

利用者の個人情報やアクセス履歴などを詐取し、利用者以外に自動的に送信するプログラムである。

- ・ **アドウェア**

Web サイト閲覧時などに、強制的に広告を表示するソフトウェアである。利用者のアクセス履歴などを外部に送信する機能をもつものもある。

- ・ **トロイの木馬**

便利なソフトウェアに見せかけて利用者にインストールさせ、その後、被害を与える不正プログラムである。ファイルに感染せず、自己増殖もしないが、PC 内のファイルをインターネット上に送信したり、ファイルやディスク内容を破壊したりする。バックドアを仕掛ける**バックドア型**や、別のマルウェアをダウンロードする**ダウンロード型**などもある。

- ・ **ランサムウェア**

利用者のデータを暗号化などで人質（使えないよう）にし、身代金（ransom）を要求するプログラムのことである。攻撃者は身元を特定されないように仮想通貨の一種である **Bitcoin** や **Tor**（接続経路の匿名化技術）などを利用する。スマートフォンや IoT 機器などを狙うタイプ、取得した個人情報を公開すると脅すタイプ、機器を操作不能状態にするタイプなど多種多様になっている。

- ・ **コインマイナー**

資源を不正利用することで、仮想通貨（**Bitcoin** など）の採掘（マイニング）を代行させる不正プログラムである。

- ・ **キーロガー**

キーボード入力を記録するソフトウェアである。本来は、キーボード操作を記録し、誤操作を行っていないかなどを確認する目的で利用される。しかし、スパイウェア代わりに使われたり、不特定多数の人間が利用する PC などに仕込んでおき、他人が入力した情報（パスワードなど）を不正入手するために悪用されたりする。

- ・ **ブラウザハイジャック**

ブラウザを乗っ取り、設定変更、意図しない広告・ツールバーの表示、意図しない Web サイトへのリダイレクトなどを行う不正プログラムである。

- ・ **RAT** (Remote Administration Tool)

バックドアの役割をもつプログラムである。遠隔メンテナンスなどの目的で使用される場合もあるが、不正利用を目的として仕掛けられることもある。

1-1-3 攻撃手法

情報セキュリティの機密性、完全性及び可用性を脅かす攻撃については、攻撃自体を技術的脅威とする場合もある。また、さまざまな攻撃手法のうち、**サイバー空間**（多様なサービスのサプライチェーンやコミュニティなどが形成された、インターネットなどの社会領域）に対する攻撃は、**サイバー攻撃**と総称されることもある。

(1) パスワードクラック

パスワードクラックは、正当な利用者本人であることを認証するため、システムなどにあらかじめ登録された**パスワード**（文字列）を、不正に解読・入手する攻撃手法である。

- ・ **総当たり攻撃** (brute force attack ; **ブルートフォース攻撃**)
ユーザ ID を固定し、すべての文字の組合せを順番にパスワードとしながら、順次ログインを試みる方式である。
- ・ **逆総当たり攻撃** (**リバースブルートフォース攻撃**)
パスワードは使われやすい文字列に固定し、ユーザ ID を変えながら、順次ログインを試みる方式である。
- ・ **辞書攻撃**
パスワードに使われやすい文字列や英単語などが登録された辞書ファイルから抽出した候補をパスワードとしながら、順次ログインを試みる方式である。
- ・ **類推攻撃**
利用者に関する情報（生年月日や電話番号など）から推測した候補をパスワードとしながら、順次ログインを試みる方式である。
- ・ **事前計算攻撃** (オフライン攻撃)
パスワードファイルを盗み、辞書ファイルなどの文字列をハッシュ化して記録内容（ハッシュ化して記録されたパスワード）と比較する方式である。ハッシュ値を効率的に管理するレインボーテーブルを用いた事前計算攻撃は、特に**レインボー攻撃**とも呼ばれる。

次に示す攻撃も、他人のパスワードを利用して不正ログインを試みるため、パスワードクラックに該当する。

- ・ **パスワードリスト攻撃**
あらかじめ入手してリスト化したユーザ ID とパスワードを利用して、自動プログラムなどで、別システムへのログインを試みる方式である。
- ・ **リプレイ攻撃**
正当な利用者が送信するユーザ ID やパスワードの情報を盗聴して複写し、複写した情報をそのまま送ってログインを試みる攻撃である。

(2) Web サイトへの攻撃

Web サイトへの攻撃は、Web サイト（又はWeb サーバで動作する OS やアプリケーション プログラムなど）の脆弱性（セキュリティホール）を悪用して、Web サイトの利用者や Web サイト（Web サーバ）そのものに対して行われる攻撃手法である。

- ・ **クロスサイトスクリプティング**（XSS : Cross Site Scripting）
悪意をもったスクリプトを、脆弱性のある標的サイト経由で利用者に送り、その標的サイトにアクセスした利用者にスクリプトを実行させて、情報を盗み出す攻撃である。スクリプトを含んだリンクは短縮 URL など偽装する。
- ・ **クロスサイトリクエストフォージェリ**（CSRF : Cross-Site Request Forgeries）
ある Web サイトにログインした状態で、罠のある別の Web サイトにアクセスした結果、ログインしている Web サイトに悪意のある要求（リクエスト）が、利用者の要求と偽って（フォージェリ）送られ、実行される攻撃である。
- ・ **クリックジャッキング**
利用者がログインしている Web ページを透明化して罠サイトの前面に仕掛け、クリック操作を誘導して利用者の意図しない処理を実行させる攻撃である。
- ・ **ドライブバイダウンロード**
Web サイトの閲覧時に、利用者の許可なしに（又は、利用者に気付かれないように）、マルウェアなどをダウンロードさせる攻撃である。罠サイトには、検索サイトの検索結果の上位に罠サイトが並ぶように細工する **SEO ポイズニング**（又は**クローキング**）などで誘導する。
- ・ **ガンブラー**
有名企業や公共機関の Web サイトを改ざんし、Web サイトを閲覧した利用者を罠サイトなどに誘導してマルウェアなどをダウンロードさせる攻撃である。
- ・ **ディレクトリトラバーサル**
ファイル名の指定に“../”を含めてディレクトリをさかのぼったり、パス名を含めて指定したりすることで、本来は非公開のディレクトリ（ファイル）にアクセスする攻撃である。
- ・ **SQL インジェクション**
データベースと連動した Web サイトのプログラム（CGI プログラム）に対して、パラメタとして SQL 文の一部を与えることで、データベースを改ざんしたり、不正に情報を入手したりする攻撃である。
- ・ **OS コマンドインジェクション**
Web サイトの入力欄に OS の操作コマンドを埋め込んで Web サーバに送信し、Web サーバを不正に操作する攻撃である。
- ・ **メールヘッダインジェクション／HTTP ヘッダインジェクション**
ヘッダ情報を改ざんし、不正情報を埋め込む（不正操作する）攻撃である。

(3) サービス妨害

サービス妨害は、各種サーバ（Web サーバ、メールサーバなど）によるサービスの提供を妨害する攻撃手法である。

- **DoS 攻撃** (Denial of Service attack ; サービス拒否攻撃)

標的サーバに大量のデータを送信し続け、サーバの CPU やメモリなどに過剰な負荷をかけてサービスを妨害する攻撃である。近年は、ボットネットを構成する多数の PC などを利用した、**DDoS 攻撃** (Distributed DoS attack ; 分散型サービス拒否攻撃) が主流である。

- **メールボム** (メール爆弾)

特定の相手に大量の電子メールを送り、電子メールサービスを妨害する攻撃である。不特定多数に送信される広告メールやチェーンメール (**SPAM ; スパムメール**) は、メールボムと同じ結果が生じることもある。SPAM はマルウェアの感染拡大に利用される場合もあるので、十分な注意が必要である。

- **バッファオーバーフロー**

長い文字列などを送り続け、プログラムが確保したメモリ領域 (バッファ) をあふれさせることにより、プログラムのアクセス権を支配し、誤作動を起こさせる攻撃である (不正アクセスの手段として利用されることもある)。

(4) 通信関連の攻撃

通信関連の攻撃は、通信の仕組みを悪用する攻撃手法である。

- **中間者攻撃** (Man-in-the-middle)

通信している二者間に攻撃者が入り込み、送信者および受信者の両方になりすましながら、通信内容の盗聴や改ざんなどを行う攻撃である。

- **キャッシュポイズニング**

キャッシュ情報 (サーバなどに一時的に保存されている情報) を不正に書き換える攻撃である。DNS のキャッシュ情報を書き換える **DNS キャッシュポイズニング** は、利用者を罠サイトに誘導する手段などに利用されることが多い。

- **IP スプーフィング**

送信元 IP アドレスを偽装 (詐称) して、相手にパケットを送る攻撃である。

- **セッションハイジャック**

正規の利用者が Web サイトなどと通信をしている最中に、そのセッション (特定利用者間の一連の通信群) を乗っ取る攻撃である。

- **第三者中継 (踏み台攻撃)**

攻撃用メールの送受信や不正アクセスなどを行うときに、自由に使用できるサーバや PC など “踏み台 (中継地点)” として悪用する攻撃である。

(5) 標的型攻撃

標的型攻撃は、ある特定の組織や人間をターゲット（標的）として行われる攻撃手法である。特に、ターゲットに対して複数の攻撃手法を組み合わせ、継続的に、執拗に行われる高度標的型攻撃は **APT 攻撃**（APT：Advanced Persistent Threats）と呼ばれる。高度標的型攻撃の基本シナリオは、次のとおりである。なお、攻撃者の行動を“偵察→武器化→配送（デリバリ）→攻撃（エクスプロイト）→インストール→C&C（遠隔操作）→侵入拡大→目的実行”とする**サイバーキルチェーン**（Cyber Kill Chain）という考え方もある。

1) 計画立案段階

攻撃に先立って、標的に関する情報収集などを行う段階（攻撃対象の情報を事前に収集する**フットプリンティング**などが行われる）

2) 攻撃準備段階

攻撃に必要な環境（標的型攻撃メール、C&C サーバなど）を準備する段階

・標的型攻撃メール

特定の組織や人間を狙って送り付けられる攻撃メールである。マルウェアに感染させるファイルを添付しているものや、アクセスするとマルウェアが自動的にダウンロードされる罠サイトの URL を記しているものがある。

- ・やり取り型：無害な電子メールのやり取りで相手を安心させる手法
- ・ばらまき型：標的組織内の複数人に同じ内容の電子メールを送る手法

3) 初期潜入段階

標的内の PC などを、マルウェアに感染させる（攻撃の起点を作る）段階

・水飲み場型攻撃

特定の組織や人間が利用すると推測される Web サイトを改ざんし、そのサイトにアクセスした利用者（PC）をマルウェアに感染させる攻撃である（水飲み場で草食動物を待ち伏せする肉食動物の行動に由来している）。

4) 基盤構築段階

PC などに感染したマルウェアから C&C サーバに**コネクトバック通信**（リモートコントロール通信経路）を開通させ、各種ツールを使ってネットワークやサーバの位置情報や認証情報を収集する段階

5) 内部侵入・調査段階

マルウェアに感染した PC などを中心に、ネットワーク内のほかの PC やサーバに侵入範囲を拡大していく段階

6) 目的遂行段階

攻撃者のコントロール下に置かれたサーバなどから機密情報を窃取して外部に送信したり、システムやデータの破壊活動を行ったりする段階

7) 再侵入段階

標的に開通させたコネクトバック通信を使って継続的に再侵入し、システム内の探索活動を続けていく段階

(6) その他の攻撃

その他の攻撃としては、次のような手法がある。なお、この中にはコンピュータに関連する**コンピュータ犯罪**も含まれている。

- ・ **フィッシング**

実在する企業（金融機関など）からの電子メールなどを装って利用者を偽の Web サイトに誘導し、クレジットカード番号、銀行口座番号、暗証番号、個人情報などを詐取する攻撃である。なお、フィッシングという行為、又はフィッシングによって不正に取得した情報を利用して行われる詐欺行為のことを総称して**フィッシング詐欺**という。

- ・ **スミッシング詐欺**（SMS+phishing）

SMS（ショートメールサービス）や類似サービスを使って、悪意のあるサイトに誘導しようとするフィッシング詐欺のことである。

- ・ **サイドチャネル攻撃**

動作中の IC チップの消費電力、放射電磁波などの副次的情報（サイドチャネル情報）を測定・解析して、機密情報を入手する攻撃である。

- ・ **ゼロデイ攻撃**

ベンダ企業からソフトウェアの脆弱性の修正プログラムが提供される前に、その脆弱性を悪用して行われる攻撃である。

- ・ **アドレススキャン／ポートスキャン**

IP アドレス／ポート番号を変えながらアクセスを繰り返し、攻撃できそうな機器やサービスがないかを探す攻撃（行為）である。不正アクセスの段階を、事前調査、権限取得、不正実行、後処理としたとき、最初の事前調査段階で実施されることが多い。

- ・ **ファイル交換ソフト／ファイル共有ソフト**

不特定多数の利用者が、ファイルを交換・共有できるソフトウェアである。マルウェアの感染拡大に悪用されることもあるため、利用する際には注意が必要である。さらに、故意又は過失によって重要な情報が記録されたファイルを共有することで、情報漏えいなどが生じる可能性もある。

- ・ **サラミ法**

全体では無視できる程度に少しずつ、資産を詐取する窃盗行為である。例えば、全銀行口座から 1 円未満の端数を特定口座に集める手口などがある。

- ・ **ワンクリック詐欺**

出会い系サイトやアダルトサイトなどで、画像やリンクをクリックしただけで料金を不当に請求されるなどの詐欺行為である。

- ・ **ビジネスメール詐欺**（BEC：Business E-mail Compromise）

巧みな騙しの手口を駆使した偽の電子メールを組織・企業に送り付け、従業員をだまして送金取引に係る資金を詐取する詐欺行為である。

1-2 情報セキュリティ技術

情報セキュリティ技術とは、情報セキュリティ対策として実施される技術的対策の中で利用されるコンピュータ技術のことである。

1-2-1 暗号化技術

暗号化技術は、情報漏えい対策及び通信内容盗聴対策として実施される暗号化の技術である。暗号化技術に関する用語について、次のように定義する。

用語	意味
平文	暗号化されていない状態のデータのことである。
暗号文	暗号化された状態のデータのことである。
暗号化	平文を暗号文に変換することである。
復号	暗号文を平文に変換することである。
暗号化鍵（暗号鍵）	暗号化に用いる特殊なデータである。
復号鍵	復号に用いる特殊なデータである。
暗号化アルゴリズム	暗号化／復号を行うアルゴリズム（プログラム）である。 ※暗号化と復号のアルゴリズムをまとめた総称
鍵長	暗号化鍵及び復号鍵の長さ（通常、ビット単位）である。 鍵長が長いほど解読されにくくなる。
解読	不正な方法で暗号文から平文を得ることである。
暗号強度	暗号文が解読されにくい度合いである。

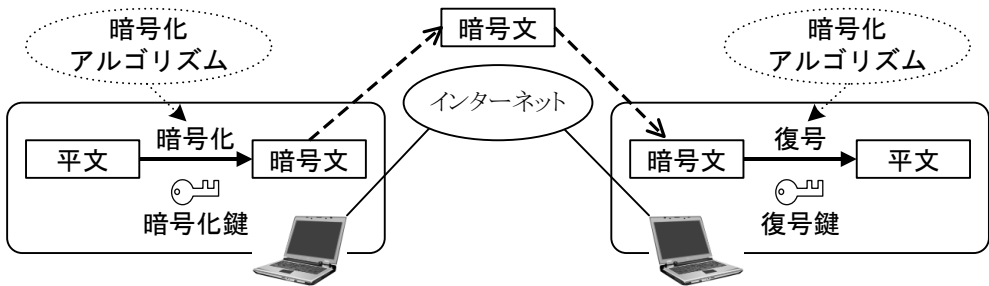


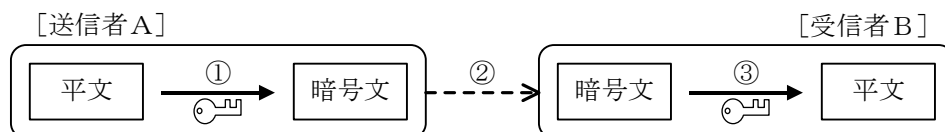
図 6-1 暗号化技術による暗号化通信のイメージ

暗号文は、原則として、復号鍵がなければ元の平文に復号できない。したがって、通信データを暗号化しておけば、通信が盗聴されても元のデータには戻せないで通信内容を秘匿できる（盗聴は防げない）。また、**ディスク暗号化**（記録媒体全体の暗号化）や**ファイル暗号化**（記録情報の暗号化）によって、情報漏えい防止効果が期待できる。

(1) 共通鍵暗号方式

共通鍵暗号方式は、データをやり取りする当事者間で、**共通鍵**を利用して暗号化／復号を行う方式である。この暗号方式は、同じ鍵で暗号化と復号を行うので、当事者は事前に共通鍵をもち、当事者以外には秘密にしておかなければならない（そのため、**秘密鍵暗号方式**と呼ばれることもある）。送信者と受信者で共通鍵を安全に共有するためには、離散対数問題を利用した **DH** (Diffie-Hellman key exchange) 法などが利用される。

【共通鍵暗号方式の手順】



- ① 送信者Aは、**共通鍵**で平文を暗号化する。
- ② 送信者Aは、暗号文を受信者Bに送信する。
- ③ 受信者Bは、暗号文を**共通鍵**で復号する。

代表的な共通鍵暗号方式には、次のようなものがある。

- **AES** (Advanced Encryption Standard)
ラインダール (Rijndael) 法を採用した NIST (アメリカ国立標準技術研究所) の標準暗号方式 (鍵長 (ビット数) に応じて、AES-128, AES-192, AES-256 がある)
- **IDEA** (International Data Encryption Algorithm)
排他的論理和演算・加算・乗算という組合せ (ラウンド) を複数回繰り返す暗号方式
- **KCipher-2**
九州大学と KDDI 研究所により共同開発されたストリーム暗号方式
- **Triple DES**
アメリカの旧国家暗号規格 DES を 3 回繰り返す暗号方式

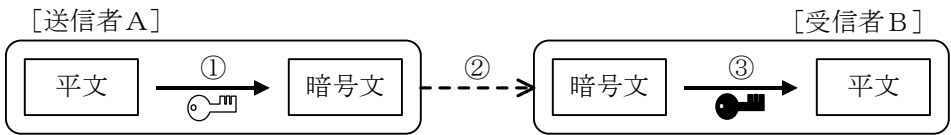
【鍵の解読手法】

- **総当たり攻撃** (brute force attack ; **ブルートフォース攻撃**)
一組の平文と暗号文を入手し、すべての鍵候補を一つずつ試す全数探索方式
- **ショートカット法**
平文と暗号文の数学的な関係などから、全数探索より効率良く鍵を見つける方式 (代数式を手掛かりとする**線形解読法**, 統計的相関を手掛かりとする**関連鍵攻撃**, 平文の変化量に対する暗号文の変化を手掛かりとする**差分解読法**)

(2) 公開鍵暗号方式

公開鍵暗号方式は、一組の**秘密鍵**と**公開鍵**を利用する方式である。秘密鍵は所有者本人が秘匿し、公開鍵は公開又は配布して誰でも使用できるようにする。この暗号方式では、一方の鍵で平文を暗号化し、暗号文はもう一方の対になる鍵でだけ復号できる。ただし、公開鍵＝暗号化用、秘密鍵＝復号用ではなく、目的に応じて使い分ける（詳細は後述）。情報漏えい対策及び通信内容盗聴対策の場合は、暗号文を受信者以外に復号されないようにするため、暗号化を誰でも使用できる“受信者の公開鍵”で、復号を受信者本人以外には知らない“受信者の秘密鍵”で行う。

【公開鍵暗号方式の手順】



- ① 送信者Aは、**受信者Bの公開鍵**で平文を暗号化する。
- ② 送信者Aは、暗号文を受信者Bに送信する。
- ③ 受信者Bは、暗号文を**受信者Bの秘密鍵**で復号する。

代表的な公開鍵暗号方式には、次のようなものがある。

- ・ **RSA**（略語ではなく、3人の開発者 Rivest, Shamir, Adleman の頭文字）
非常に大きな数の素因数分解を利用した暗号方式
- ・ **エルガマル暗号方式**
離散対数問題を応用した暗号方式
- ・ **楕円曲線暗号方式**
楕円曲線の数式を利用した暗号方式（RSA 暗号と比べて、短い鍵長で同レベルの安全性を実現できる）

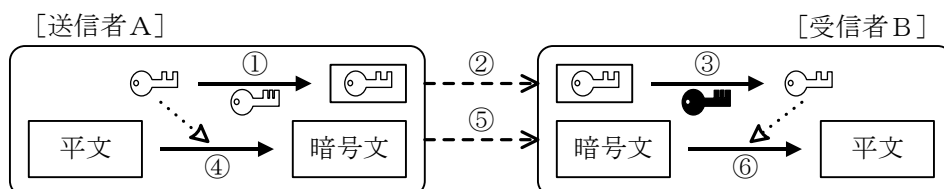
共通鍵暗号方式と公開鍵暗号方式の長所／短所をまとめると、次のようになる。

	共通鍵暗号方式	公開鍵暗号方式
長所	・ 同一鍵長ならば、公開鍵暗号方式よりも暗号強度が高い。 ・ 公開鍵暗号方式よりも暗号化／復号にかかる時間が短い。	・ 鍵の管理・受渡しが容易である。 （n人で暗号化通信をする場合、鍵の数はn組 [鍵の種類は2n種類]）
短所	・ 長時間の盗聴によって、共通鍵が漏えいする可能性がある。 ・ 鍵の管理・受渡しが煩雑である。 （n人で暗号化通信をする場合、鍵の種類は $n(n-1)/2$ 種類）	・ 同一鍵長ならば、共通鍵暗号方式よりも暗号強度が低い。 ・ 暗号化／復号アルゴリズムが複雑なので、処理に時間がかかる。 ・ なりすましが生じる可能性がある。

(3) セッション鍵暗号方式（ハイブリッド暗号）

セッション鍵暗号方式（ハイブリッド暗号）は、共通鍵暗号方式の短所である鍵の受渡しに公開鍵暗号方式を利用し、暗号化通信を共通鍵暗号方式で行う方式である。

【セッション鍵暗号方式の手順】



- ① 送信者Aは、生成した**共通鍵**を**受信者Bの公開鍵**で暗号化する。
- ② 送信者Aは、暗号文（暗号化した**共通鍵**）を受信者Bに送信する。
- ③ 受信者Bは、暗号文を**受信者Bの秘密鍵**で復号し、**共通鍵**を取得する。
- ④ 送信者Aは、**共通鍵**で平文を暗号化する。
- ⑤ 送信者Aは、暗号文（暗号化した平文）を受信者Bに送信する。
- ⑥ 受信者Bは、暗号文を**共通鍵**で復号する。

セッション鍵暗号方式では、通信のセッション番号や乱数をもとに**ハッシュ関数**（入力値から固定長の一意の出力値を得る関数）などを利用して共通鍵を生成する。共通鍵の生成は暗号化通信のセッションごとに行われ、公開鍵暗号方式により受け渡される。そのため、万一、共通鍵が漏えいしても被害を抑えられる。また、通常時是一組の公開鍵／秘密鍵を管理するだけなので鍵の管理も容易であり、暗号化通信は共通鍵暗号方式で高速に行える。つまり、共通鍵暗号方式と公開鍵暗号方式の長所を組み合わせた方式といえる。ただし、なりすましの可能性については、セッション鍵暗号方式でも対処できない。

【ブロック暗号利用モード】

ブロック暗号利用モードは、固定長のブロック単位で暗号化する**ブロック暗号**の使い方を定義したものである（ビット単位やバイト単位で暗号化する方式は、**ストリーム暗号**という）。例えば、共通鍵暗号方式のAESは128ビットのブロック暗号、IDEAは64ビットのブロック暗号を採用している。

ブロック暗号利用モードには、メッセージ秘匿のためのモード（暗号化方式）と、メッセージ認証のためのモード（メッセージ認証コード）などがある。例えば、ECBモードはメッセージ秘匿のための利用モードである。このモードでは、複数ブロックを暗号化すると、同じ平文から同じ暗号文が生成されて内容を推測される可能性がある。そのため、同じ平文を暗号化した場合でも、異なる暗号文が生成されるCBCモードなどが規定されている。

1-2-2 認証技術

(1) 利用者認証

利用者認証とは、利用者が本人であることを確認する、利用者確認のための認証技術である。悪意のある第三者が正当な利用者を装うなりすまし対策として用いられる。

利用者認証には、目的や用途に応じてさまざまな方法がある。また、複数の認証情報（知識[パスワードなど]、所持[IC カードなど]、生体[身体的特徴など]）を組み合わせた**多要素認証**（パスワードと生体情報の**2要素認証**など）や、複数回にわたって同じ方法（又は異なる方法）で認証を行う**多段階認証**を用いる場合もある。

① ユーザ ID／パスワード

ユーザ ID（利用者 ID）は利用者に割り振られた識別子、**パスワード**はあらかじめ登録しておく文字列（キーワード）である。ログイン時に、利用者が入力したユーザ ID から登録済みのパスワードを検索し、利用者が入力したパスワードと一致すれば、パスワードを知っている利用者本人であると認証する技術である。パスワードの不正利用を防止するため、利用者にパスワードの管理・要件などを周知徹底する。

【パスワード管理の注意点】

- ・英数字、記号などを組み合わせた、意味のない文字列にする。
- ・必要十分な長さにする（最低でも 6 ～ 8 文字以上にする）。
- ・再利用しない（同じパスワードを複数の認証で利用しない）。
- ・人目に触れるところに記録しない（メモなどの管理を徹底する）。
- ・他人と共有したり、他人に教えたりしない。
- ・忘れた場合、漏えいした場合、不要になった場合などは、速やかに管理者に連絡し、然るべき処置（無効化など）を実施する。
- ・**パスワードリマインダ**（パスワードを忘れたとき、質問の回答や合言葉で本人を確認する仕組み）の回答は推測されにくいものにする。

② IC カード

プラスチック製のカードに、情報を記録できる IC チップを埋め込んだ IC カードを使用して、利用者を認証する技術である。住民基本台帳カード、クレジットカード、定期券（Suica、PASMO など）、社員証などで利用されている。利用時に IC カードに設定された **PIN**（Personal Identification Number；個人識別番号／暗証番号）を入力させることで、正当な保有者であるかを確認することもできる。

IC カードは、磁気カードに比べて 100 倍以上の記録容量をもっており、データの暗号化も可能なので、偽造にも強い。また、IC カード内部に書き込まれた情報への不正アクセスや改ざんを防止できる**耐タンパ性**にも優れている。

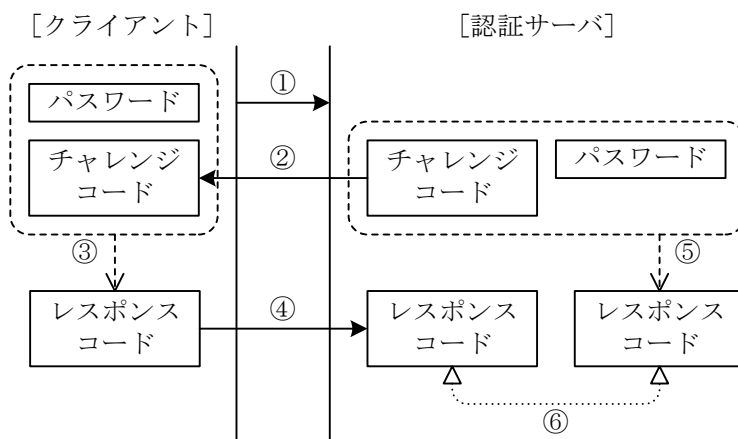
③ ワンタイムパスワード (OTP : One-Time Password)

主に、通信回線に接続されているコンピュータで行われる利用者認証技術である。**セキュリティトークン**と呼ばれる機器（デバイス）などから得られる、ログインごとに異なるパスワードを使用することで、パスワード漏えいなどの危険性を回避する。ワンタイムパスワードの代表的な方式（技術）は、次のとおりである。

・チャレンジレスポンス認証

通信回線上で直接、パスワードを送受信しない方式である（この場合、秘密情報を送受信しないで、秘密情報（パスワードなど）を知っていることを証明する。これを、**ゼロ知識証明**という）。パスワードの代わりに、アクセス要求ごとに異なるチャレンジコード（乱数）とパスワードから生成したレスポンスコードを使用して利用者を認証する。

IC カードにワンタイムパスワードの生成機能を記録して、PIN による認証と併用する場合もある。



- ① クライアントがユーザ ID を送信して、アクセスを要求する。
- ② 認証サーバは、チャレンジコードを生成して送信する。
- ③ クライアントは、自分のパスワードとチャレンジコードからレスポンスコードを生成する。
- ④ クライアントは、レスポンスコードを認証サーバに送信する。
- ⑤ 認証サーバは、ユーザ ID から求めた利用者のパスワードと、送信したチャレンジコードからレスポンスコードを生成する。
- ⑥ 認証サーバは、受信したレスポンスコードと⑤で生成したレスポンスコードを比較し、一致すればクライアントを認証する。

・タイムシンクロナス認証

キャッシュカードサイズの専用ハードウェアに、一定時間ごとに異なるコード（PIN ; 暗証番号）が表示され、そのコードを時刻やユーザ ID の後に付けて送信する方式である。専用ハードウェアと認証サーバは同期がとれているため、送信コードが一致すればログインが許可される。

④ 生体認証（バイオメトリクス認証）

個々人に固有の身体的・行動的情報を利用することで、個人（利用者）を特定する認証技術である。認証システムにあらかじめ登録しておいた身体的情報や、行動的情報（署名をするときの速度や筆圧などによる署名認証）によって個人を特定する。セキュリティ区画への入退室管理のほか、銀行のATMや個人用PCでも利用されている。

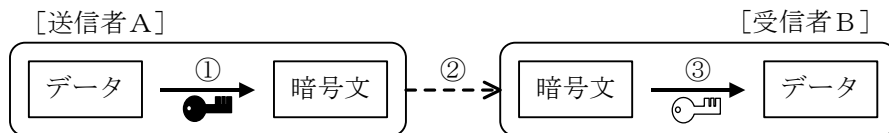
なりすましの防止に効果的であるが、特別な認証システムが必要であること、認証精度（本人拒否率と他人受入率の関係）、代替が難しいこと、身体の経年変化への対応、個人情報保護などの面で、問題が残されている。

名称	認証情報
顔認証	顔の特徴（目、鼻、口の位置関係など）
虹彩認証	目の虹彩（どう孔から発生するしわ）パターンの形状や濃淡など
網膜認証	目の網膜パターン（毛細血管など）
声紋認証	声の波形の特徴など
掌認証	掌（てのひら）の幅や指の長さなど
静脈認証	静脈の分岐角度や長さなど ※掌認証に含まれる場合もある。
指紋認証	指紋（隆線によって形作られる文様）の特徴点（マニューシャ）

⑤ 公開鍵暗号方式

公開鍵暗号方式で利用される秘密鍵は、所有者本人が秘匿していることを利用する認証技術である。

【公開鍵暗号方式による認証手順】



- ① 送信者Aは、**送信者Aの秘密鍵**でデータを暗号化する。
- ② 送信者Aは、暗号文（暗号化したデータ）を受信者Bに送信する。
- ③ 受信者Bは、暗号文を**送信者Aの公開鍵**で復号する。
→ 正しく復号できた場合、送信者をAであると認証する。

利用者認証のポイントとなるのは、手順③である。受信者Bが送信者Aの公開鍵で暗号文を復号できたのは、暗号文がAの秘密鍵で暗号化されたものであることを意味する（一方の鍵で平文を暗号化した場合、もう一方の対になる鍵でだけ復号できる）。Aの秘密鍵はAだけが所有（秘匿）しているので、この暗号文はA以外には作成することができない。つまり、送信者は間違いなくAであると認証できるのである。

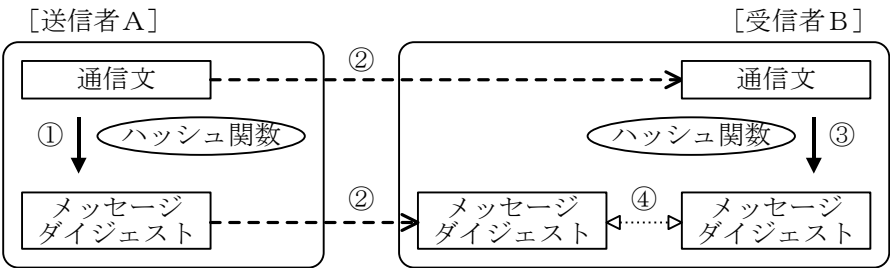
ただし、この認証を成立させるためには、受信者Bが所有する“Aの公開鍵”が間違いなくAのものであり、対になる秘密鍵を間違いなくAだけが所有していることが前提となることに注意する必要がある。

(2) メッセージ認証

メッセージ認証とは、データが不正に書き換えられていないことを確認する認証技術である。データが不正に書き換えられる改ざん防止対策として用いられる。

データ通信時に利用される代表的なメッセージ認証方法として、通信文（データ）から**ハッシュ関数**によってメッセージダイジェストを求める方法がある。

【メッセージ認証の手順】



- ① 送信者Aは、通信文からメッセージダイジェストを求める。
- ② 送信者Aは、通信文とメッセージダイジェストを受信者Bに送信する。
- ③ 受信者Bは、受信した通信文からメッセージダイジェストを求める。
- ④ 受信者Bは、受信したメッセージダイジェストと③で生成したメッセージダイジェストを比較し、一致すれば改ざんされていないことを認証する。

・ハッシュ関数

任意長の入力値から、固定長の出力値（ハッシュ値）を得る関数である。同じ入力値からは同じ出力値が得られ、出力値から入力値を得ることは困難である（一方向性）という特性をもっている。SHA-1、MD5 というハッシュ関数も存在したが、現在は使用が推奨されていない。また、SHA-2 の後継として SHA-3 が発表されたが、元は Keccak という全く別系統のハッシュ関数である。

ハッシュ関数名		ハッシュ値の長さ	規格化／標準化
SHA-2	SHA-256	256 ビット	SHA-1 の後継標準として NIST により規格化
	SHA-512	512 ビット	

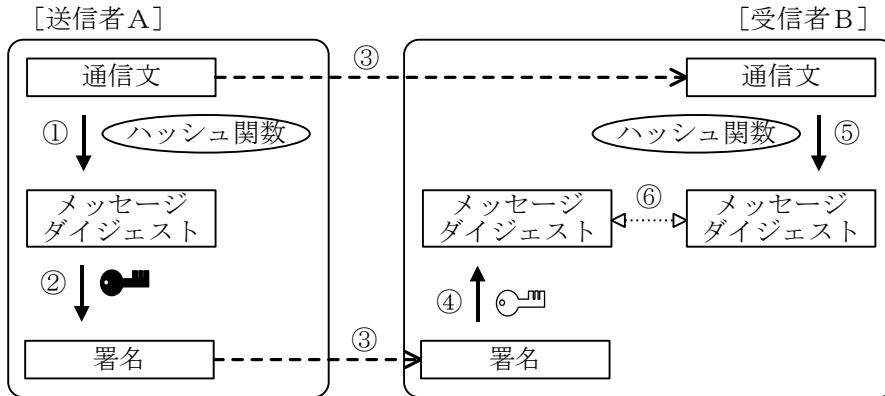
ハッシュ関数の特性により、送信時と受信時の通信文にわずかでも違いがあれば、受信側でメッセージダイジェストが一致しないことから改ざんがあったと検出できる。

ただし、既知のハッシュ関数を使用すると、通信文とメッセージダイジェストの両方が改ざんされ、改ざんを検出できなくなる可能性がある。この問題を解決する技術として、**MAC** (Message Authentication Code ; **メッセージ認証符号**) がある。この方式は、送信者と受信者が共有する鍵（共通鍵）を用いて、“通信文＋共通鍵” から求めた MAC 値をメッセージダイジェストとする（MAC 値を求める方法としては、ブロック暗号利用モードやハッシュ関数が利用される）。この方式の場合、共通鍵がわからない第三者は正しい MAC 値を求めることができないので、安全性が高くなる。

(3) デジタル署名

デジタル署名とは、公開鍵暗号方式を用いた利用者認証とメッセージ認証を組み合わせた認証技術である（デジタル署名のような電子的な署名を、一般の署名（サイン）と区別するために**電子署名**という場合もある）。

【デジタル署名の手順】



- ① 送信者 A は、通信文からメッセージダイジェストを求める。
- ② 送信者 A は、署名鍵（**送信者 A の秘密鍵**）で署名（メッセージダイジェストを暗号化）する。
- ③ 送信者 A は、通信文と署名を受信者 B に送信する。
- ④ 受信者 B は、署名を検証鍵（**送信者 A の公開鍵**）で検証（復号）する。
- ⑤ 受信者 B は、受信した通信文からメッセージダイジェストを求める。
- ⑥ 受信者 B は、④で復号したメッセージダイジェストと、⑤で生成したメッセージダイジェストを比較する。一致すれば、送信者が A であること、及び通信文が改ざんされていないことを認証する。

代表的なデジタル署名の方式として、NIST（アメリカ国立標準技術研究所）が政府標準として開発した **DSA** (Digital Signature Algorithm) がある。DSA は、公開鍵暗号方式の一つであるエルガマル暗号方式を用いたエルガマル署名を改良したもので、メッセージダイジェストを SHA-1 で生成していた。しかし、現在はハッシュ関数が SHA-1 から SHA-2 (SHA-256, SHA-512 など) に移行されている。

また、XML 文書に対するデジタル署名の構文を規定した、W3C 勧告による **XML デジタル署名** もある。XML デジタル署名では、通信文に付ける通常の署名のほかに、指定したエレメントやコンテンツなどに対しても署名を付けることができる。

デジタル署名では、公開鍵暗号方式による利用者認証で前提となる、公開鍵・秘密鍵ペアの正当性を認証機関と呼ばれる第三者（民間企業）が保証する。このような認証機関は、デジタル署名以外（例えば、公開鍵暗号方式など）でも利用されている。

認証機関は、**認証局**（CA : Certification Authority）、**登録局**（RA : Registration Authority）および**検証局**（VA : Validation Authority）などからなる第三者機関である。利用者や下位の認証局に対して**デジタル証明書**を発行し、公開鍵やデジタル署名の正当性を保証する。また、認証機関自身の正当性を証明して公開鍵を配布するために、自らの秘密鍵で署名した**ルート証明書**を発行する。

- **デジタル証明書**（電子証明書又は**公開鍵証明書**）

認証局が発行する、個人や組織に関する電子的な証明書である。デジタル証明書には、署名アルゴリズム（ハッシュ関数）、暗号方式、有効期限、被証明者の公開鍵などが記載されている。デジタル証明書は、認証局が発行したものであることを確認できるように、認証局の秘密鍵で署名されている。

デジタル証明書の復号鍵（公開鍵）は、認証局が公開しているルート証明書から入手できる。ITU-T（国際電気通信連合－電気通信標準化部門）の X. 509 で公開されているデジタル証明書の標準仕様は、次のとおりである。

項目名	内容
バージョン情報	v1～v3
シリアル番号	証明書番号
署名アルゴリズム	SHA-2（ハッシュ関数）、RSA（公開鍵）など、署名に使用されるアルゴリズム
発行者名	発行機関（通常は認証局名）
有効期間	開始日時と終了日時（数秒から数百年まで）
証明者名	主体名（証明書の保持者）
公開鍵情報	証明者の公開鍵情報

- **CRL**（Certificate Revocation List ; **証明書失効リスト**）

有効期間内であるにもかかわらず、鍵の漏えい・紛失などで失効させる必要のあるデジタル証明書のリストである。CRL には、デジタル証明書のシリアル番号と失効日が記載され、一般に認証局のリポジトリで公開される。CRL を調べることにより、デジタル証明書の有効性を確認できる。

- **OCSP**（On-line Certificate Status Protocol）

デジタル証明書の有効性を、リアルタイムで確認するプロトコルである。OCSP は、IETF（Internet Engineering Task Force）が RFC 2560 として規定している。OCSP サーバは、認証局（CA）と CRL を集中管理する検証局（VA）が運営する。OCSP クライアントは、デジタル証明書を OCSP サーバに確認させることで、CRL の取得・照合の手間を省くことができる。ただし、OCSP はデジタル証明書の失効を確認するだけなので、有効期間などはクライアント側で確認しなければならない。

また、認証局（CA）には、対外的なやり取りを行う公的な**パブリック CA**と、特定の組織内などの閉じた環境内で機能する私的な**プライベート CA**がある。

(4) その他の認証技術

① 時刻認証（タイムスタンプ認証）

電子文書法に基づき、第三者機関である時刻印業者（TS 業者；認証機関の場合もある）が、電子文書の存在と真正であることを保証する認証技術である。時刻印業者が電子文書から生成されたメッセージダイジェストに時刻情報を付加した TS トークンを作成して保管することで、電子文書がその時刻に存在していたこと、およびその時刻から改ざんされていないことを保証する。なお、通常は、作成者のデジタル署名とデジタル証明書も電子文書に添付する。

② IEEE 802.1X

LAN で利用されるクライアント認証技術である。**RADIUS サーバ**などの認証サーバと認証用のソフトウェア（**サブリカント**）などで実現される。

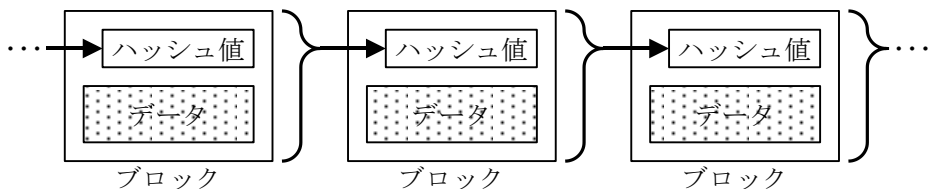


③ CAPTCHA（画像認証）

画面に表示された歪んだ文字や数字を入力させて、利用者が人間であることを認証する技術である。プログラムによる自動投稿などを防止する目的で利用される。

④ ブロックチェーン

暗号資産などで用いられている技術で、改ざん防止（メッセージ認証）の効果があ。特定ブロックのハッシュ値を引き継ぎながら、チェーンのようにつなげることで、1か所でも改ざんすると全体の整合性がとれなくなるようにしている。



1-2-3 PKI（Public Key Infrastructure；公開鍵基盤）

PKI（公開鍵基盤）とは、公開鍵暗号技術、認証機関、デジタル署名を用いたセキュリティ基盤である。電子商取引（EC：Electronic Commerce）や、セキュアプロトコル **SSL**（Secure Sockets Layer）などで利用されている。また、PKIを利用した、行政機関への申請・届出などを行うための **GPKI**（Government PKI；**政府認証基盤**）という認証基盤もある。GPKIでは、各府省の認証局と民間の認証局の相互認証が必要となるため、両者の認証局を仲介する **ブリッジ認証局**（**BCA**：Bridge CA）が必要となる。

1-3 情報セキュリティ管理

情報セキュリティ管理とは、情報資産（物理的資産、ソフトウェア資産、無形情報資産など）に関わるさまざまなリスクを分析・評価し、適切なセキュリティ対策を立案・実行することである。ここでは、企業などの組織における情報セキュリティマネジメント、情報セキュリティの評価・認証制度、およびリスクマネジメントについて説明する。

1-3-1 情報セキュリティマネジメント

情報セキュリティマネジメントとは、組織における情報セキュリティに対する考え方を明文化し、具体的に実施・管理していくことである。

(1) 情報セキュリティポリシー

情報セキュリティポリシーとは、情報セキュリティに対する組織としての考え方を明文化したものである。情報セキュリティポリシーは、次のように体系化して管理する。

名称	役割
情報セキュリティポリシー	情報セキュリティに対する考え方
情報セキュリティ基本方針	組織の統一かつ基本的な考え方や理念
情報セキュリティ対策基準	基本方針を実践するための遵守事項や基準
情報セキュリティ対策実施手順 など	具体的な実施手順 など

情報セキュリティ基本方針は、組織として保護する情報資産及び保護理由を明確にし、情報セキュリティに関する取組み（考え方や理念）を文書化したものである。情報セキュリティのための経営陣の方向性および支持を規定する目的で策定される。

情報セキュリティ対策基準は情報セキュリティ基本方針をもとに、組織として統一した情報セキュリティに関して遵守すべき行為や判断の基準などを、策定したものである。

情報セキュリティ対策実施手順は、情報セキュリティ対策の具体的な実施手順についてまとめたものである。情報セキュリティの管理機能（予防、検知、最小化、復旧）の観点から、対象（ハードウェア製品、ソフトウェア製品など）ごとに、具体的なセキュリティ対策（セキュリティコントロール）を規定する。特に事業の継続が困難になるような状況への対応策を、事前に規定しておくことが重要となる（**情報セキュリティ継続**）。

・ **コンティンジェンシープラン**（緊急時対応計画）

主に緊急事態（非常事態）が発生したときの対応策（最小化、復旧）を規定したものである。災害を原因として発生した緊急事態からの復旧対応を、特に**ディザスタリカバリ**（災害復旧）と呼ぶ場合もある。

企業などの組織においては、情報セキュリティポリシーの中で、さまざまなセキュリティの規程を作成する必要がある。代表的な**情報セキュリティ諸規程**には、次のようなものがある。これらの規程に関しては、基本的に経営者層が承認する。

【企業活動の情報セキュリティ諸規程】

- ・雇用契約に関する規程
- ・機密管理規程
- ・情報管理規程
- ・情報セキュリティインシデント対応規程
(コンピュータウイルス感染時の対応など)
- ・情報セキュリティ教育の規程
- ・対外説明の規程
- ・職務規程
- ・文書管理規程
- ・プライバシーポリシー (個人情報保護方針)
- ・罰則の規程
- ・規則更新の規程

これらの規程のうち、近年、重要視されているものが**プライバシーポリシー (個人情報保護方針)**である。プライバシーポリシー (個人情報保護方針) は、氏名や生年月日など個人を特定できる情報 (個人情報) を保護するための取組みに関する方針である。この方針に基づいて、体系的に統合された**コンプライアンスプログラム** (企業における法令遵守のための計画) の一環として、個人情報保護が実施される。

個人情報保護に関する取組みの一つとして、**JIS Q 15001** (個人情報保護マネジメントシステム—要求事項) に基づき、個人情報の取扱いについて適切な保護措置を講じる体制を整備している民間事業者などを認定する**プライバシーマーク制度**の活用がある。この制度で認定された事業者に対しては、**プライバシーマーク** (Pマーク) が付与される。

(2) ISMS (Information Security Management System)

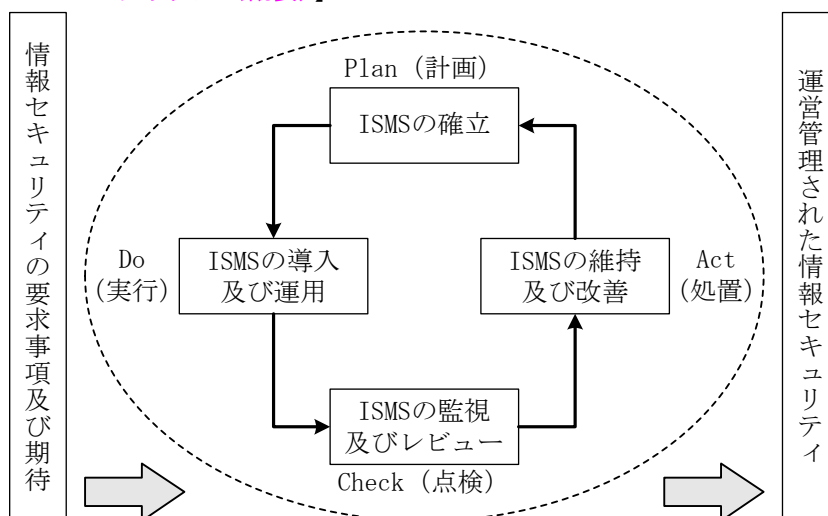
ISMS (情報セキュリティマネジメントシステム) は、組織の情報資産全体の正しい運用管理、およびセキュリティの確保・維持を目的とするマネジメントシステムである。ISMSは、英国規格協会の BS7799-1, BS7799-2 をベースに構築され、現在では ISO/IEC 27001, ISO/IEC 27002 (JIS Q 27000 シリーズ) として標準化されている。

ISO/IEC 規格	JIS 規格	概要
ISO/IEC 27001	JIS Q 27001	情報セキュリティマネジメントシステム—要求事項
ISO/IEC 27002 (ISO/IEC 17799)	JIS Q 27002 (JIS X 5080)	情報セキュリティマネジメントの実践のための規範 (ISO/IEC 17799 が改訂され、ISO/IEC 27002 に改称された)

ISMS は、組織の責任者 (経営陣) および要員が、効果的な情報セキュリティマネジメントシステムを構築・運用する上でのモデルを提供する。また、責任者に対して、ISMS の確立、導入・運用、監視・レビュー、維持・改善に関するコミットメント (公的な関係およびその承認) を要求する。

ISMS は、次に示す **PDCA サイクル** (Plan ; 計画, Do ; 実行, Check ; 点検, Act ; 処置) で運用される。

【ISMS の PDCA サイクル (概要)】



① Plan (計画) : ISMS の確立

- 1) 情報セキュリティ推進体制の構築
- 2) ISMS 適用範囲及び境界の定義・決定
- 3) ISMS 基本方針 (情報セキュリティ基本方針を包含) の策定
- 4) リスクアセスメント (リスクの特定・分析・評価) の実施
- 5) リスク対応のための管理目的及び管理策の選択
- 6) ISMS 導入及び運用の承認
- 7) 適用宣言書の作成

② Do (実行) : ISMS の導入及び運用

- 1) リスク対応計画の策定と実施
- 2) リスク対応のための管理目的を満たす管理策の実施
- 3) 管理策の有効性の測定方法の定義
- 4) 教育・訓練及び意識向上のためのプログラムの実施
- 5) ISMS の運用及び ISMS のための経営資源の管理

③ Check (点検) : ISMS の監視及びレビュー

- 1) ISMS の有効性に関する定期的なレビューの実施
- 2) 管理策の有効性の測定
- 3) ISMS に関する内部監査の実施

④ Act (処置) : ISMS の維持及び改善

- 1) 特定した改善策の導入
- 2) 適切な是正措置及び予防処置の実施

ISMS を第三者機関が評価・認証する制度として、一般財団法人 日本情報経済社会推進協会（JIPDEC）の **ISMS 適合性評価制度**がある。ISMS 認証を取得するためには、JIPDEC が ISMS 認証機関として認定した組織に対して申請する。ISMS 認証機関は、ISO/IEC 27001（JIS Q 27001）の管理目的及び管理策を基準として、認証申請者が構築した ISMS の適合性を評価する。

【ISO/IEC 27001:2013（JIS Q 27001:2014）における管理目的及び管理策】

1. 情報セキュリティのための方針群
情報セキュリティのための経営陣の方向性
2. 情報セキュリティのための組織
内部組織，モバイル機器及びテレワーキング
3. 人的資源のセキュリティ
雇用前，雇用期間中（情報セキュリティの教育及び訓練），雇用の終了及び変更
4. 資産の管理
資産に対する責任，情報分類，媒体の取扱い
5. アクセス制御
アクセス制御に対する業務上の要求事項，利用者アクセスの管理，利用者の責任，システム及びアプリケーションのアクセス制御
6. 暗号
暗号による管理策
7. 物理的及び環境的セキュリティ
セキュリティを保つべき領域，装置
8. 運用のセキュリティ
運用の手順及び責任，マルウェアからの保護，バックアップ，ログ取得及び監視，運用ソフトウェアの管理，技術的ぜい弱性管理，運用システムの監査に対する考慮事項
9. 通信のセキュリティ
ネットワークセキュリティ管理，情報の転送
10. システムの取得，開発及び保守
情報システムのセキュリティ要求事項，開発及びサポートプロセスにおけるセキュリティ，試験データ
11. 供給者関係
供給者関係における情報セキュリティ，供給者のサービス提供の管理
12. 情報セキュリティインシデント管理
情報セキュリティインシデントの管理及びその改善
13. 事業継続マネジメントにおける情報セキュリティの側面
情報セキュリティ継続，冗長性
14. 順守
法的及び契約上の要求事項の順守，情報セキュリティのレビュー

1-3-2 リスクマネジメント

JIS Q 31000:2019 では、**リスクマネジメント**を「リスクについて、組織を指揮統制するための調整された活動」と規定している。

リスクとは「目的に対する不確かさの影響」（JIS Q 31000:2019）である。ここでいう影響とは、期待されていることから好ましい方向又は好ましくない方向に乖離（かいり）することである。例えば、投資などは好ましい又は好ましくない方向に乖離する可能性がある（経営主体の管理下で発生するこのようなリスクを、**投機的リスク**という）。一方、災害などは好ましくない方向にだけ乖離する可能性がある（経営主体の管理外で発生するこのようなリスクを、**純粋リスク**という）。なお、**情報セキュリティリスク**は、「脅威が情報資産の脆弱性又は情報資産グループの脆弱性につけ込み、その結果、組織に損害を与える可能性に伴って生じる」（JIS Q 27000:2019）と規定された純粋リスクである。

情報セキュリティのリスクマネジメントでは、情報セキュリティリスクにあらかじめ組織的な対応を決めて管理する（ISMS の一環としても実施される）。

【リスクマネジメントの手順】

① 組織の状況の確認（情報資産の調査・分類）

リスク分析の対象となる情報資産を洗い出し、機密性、完全性、可用性の側面から重要性を検討して分類する。この結果により、情報資産を保護するための判断基準（要求される情報セキュリティの水準）を決定する。

② **リスクアセスメント**の実施（情報セキュリティリスクアセスメント）

1) リスク特定（リスク識別）

情報資産に対する脅威や脆弱性などから、リスクを特定する。

2) リスク分析

リスクの発生頻度や発生時の影響（被害・損失）の大きさ（**リスクレベル**）などを分析する。リスク分析では、**ペリル**（損失原因）と**ハザード**（危険状態）に着目する。ペリルはリスクにつながる原因であるが、ハザードがなければリスクにつながりにくい（脅威＝ペリル、脆弱性＝ハザードと考えてもよい）。また、**モラルハザード**（リスクに対する保険などの補償によって、危機感が低下してリスクを促進する状態）にも注意する必要がある。

3) リスク評価

リスク分析の結果と**リスク受容基準**から、それぞれのリスクに対して**リスク受容**が可能（リスクレベルが許容範囲内）かを評価する。

③ リスクへの対応（**リスク対応**）

受容できないリスクに対し、要求される情報セキュリティの水準を勘案したリスク対応策（**リスク対応計画**など）を適切に決定・実施する。受容するリスクおよびリスク対応計画は、リスク所有者から承認を得る。

(1) リスクの種類

リスク特定では、情報資産ごとにリスクを識別する。リスクの種類は、ハードウェアなどの物理的資産であれば故障、破壊、盗難など、ソフトウェア資産であればエラーや誤動作など、対象となる情報資産によって異なるので漏れないようにする。

また、リスク分析では、リスクが与える損失の種類によってリスクを分類する場合もある。リスクが与える損失には、次のような種類がある。なお、一般的なリスク分析では、純粹リスク以外に、利益・損失の両方が生じる投機的リスクも対象とする。

- ・ **財産損失** : 財産の喪失、財産価値の低下などによって生じる損失
- ・ **責任損失** : 責任を負って支払う賠償金、罰金などによって生じる損失
- ・ **純収益損失** : 販売機会の喪失などで収益が減少することによって生じる損失
- ・ **人的損失** : 人的資源に生じる損失、又は人的能力の低下によって生じる損失

(2) リスク対策

リスク対応で実施が検討されるリスク対策の考え方には、次のような種類がある。なお、ISMSではセキュリティリスクへの対応を、**リスク回避**、**リスク低減**（リスク予防に該当）、**リスク移転**、**リスク受容**（何もしない）の四つとしている。

リスク対策名	内容
リスクコントロール	リスクの発生を予防し、損失を軽減する。
リスク回避	資産の使用、活動を止めるか、代替する。
リスク予防	脆弱性を改善し、脅威の発生頻度を低減する。
リスク分離	資産を分離して、リスクの影響を低減する。
リスク集中	リスクの発生源を集めて、集中管理する。
リスク移転	レンタル会社などとの契約によって、リスクを移転する。
リスク最適化	リスク分離などにより、リスクを受容可能な状態にする。
リスクファイナンス	リスクが発生した場合に被る損害の資金手当を行う。
リスク保有	リスクを保有したまま、経常費や準備金で損失に備える。
リスク移転	保険をかけて、リスク発生時の損失を他社と分担する。

注記：JIS Q 27000:2019では「好ましくない結果に対処するリスク対応は、リスク軽減、リスク排除、リスク予防及びリスク低減と呼ばれることがある」と規定されている。

リスク対策は、情報資産の価値又は発生する損失と対策費用のバランス（費用対効果）を検討して選択される。そのため、リスク対応後に**残留リスク**が生じることもあるので、受容したリスクも含めて調査（モニタリング、レビューなど）を継続的に実施する。

また、リスク対応では、複数のリスクが同時に発生したとき、どのリスクの対応を優先させるかも決めておく必要がある。

1-4 情報セキュリティ機関・評価基準

情報セキュリティの概要のまとめとして、情報セキュリティに関連する機関および代表的な情報セキュリティ評価基準について説明する。

1-4-1 情報セキュリティ機関

① 情報セキュリティ委員会

最高情報セキュリティ責任者(CISO)をはじめとした経営層の意思決定機関である。

② SOC (Security Operation Center ; セキュリティ・オペレーション・センター)

セキュリティインシデントの検知を主目的として、セキュリティ機器やネットワークなどを監視・分析する拠点(組織・機関又はサービス・役割など)である。

③ CSIRT (Computer Security Incident Response Team)

セキュリティインシデントに関する情報を収集・監視するとともに、問題が発生したときには原因や影響範囲を調査し、対応活動を行う組織の総称である。

組織内 CSIRT	組織にかかわるインシデントに対応する。
国際連携 CSIRT	国の窓口としてインシデントに対応する。
コーディネーションセンター	対応の中で、CSIRT 間の情報連携、調整を行う。
分析センター	インシデントの傾向分析やマルウェアの解析、攻撃の痕跡の分析を行う(注意喚起を含む)。
ベンダチーム	自社製品の脆弱性に対して、パッチ作成や注意喚起を行う。
インシデントレスポンス プロバイダ	組織内 CSIRT の機能の一部又は全部を、サービスプロバイダとして有償で請け負う。

④ JPCERT/CC (Japan Computer Emergency Response Team/Coordination Center)

日本国内のセキュリティインシデントに関する情報を、収集・分析・発信する一般社団法人である。技術的な立場における日本の CSIRT でもある。

・情報セキュリティ早期警戒パートナーシップ

国内におけるソフトウェアなどの脆弱性関連情報を、適切に流通させることを目的に作られた枠組みで、JPCERT/CC が調整機関という役割を担っている。

・JVN (Japan Vulnerability Notes)

ソフトウェアなどの脆弱性関連情報を提供している脆弱性対策情報ポータルサイトで、IPA(独立行政法人 情報処理推進機構)と共同で運営している。

・CSIRT マテリアル

組織的なインシデント対応体制(組織内 CSIRT)の構築を支援する目的で作成された、CSIRT 構築・運用に関するガイドラインである。構想・構築・運用の3段階のフェーズで構成され、運用フェーズには CSIRT ガイドやインシデントハンドリングマニュアルが含まれている。

- ⑤ **NISC** (National center of Incident readiness and Strategy for Cybersecurity)
日本の情報セキュリティ政策を統一的かつ効率的に遂行するために、内閣官房に設置されている**内閣サイバーセキュリティセンター**である。なお、同じ目的をもつ機関として、内閣には**サイバーセキュリティ戦略本部**が設置されている。
- ⑥ **IPA セキュリティセンター**
関係機関と連携を図り、民間では困難な情報収集・分析と、それらの知見の一般化を行う、IPA 内の組織である。コンピュータ不正アクセス届出制度、コンピュータウイルス届出制度などの受付も行っている。
- ・ **サイバー情報共有イニシアティブ (J-CSIP)**
公的機関である IPA を情報ハブ（集約点）とし、参加組織間で情報を共有し、高度なサイバー攻撃対策につなげていく取組みである。
 - ・ **組織における内部不正防止ガイドライン**
組織において必要な内部不正対策を効果的に実施できるように、IPA セキュリティセンターの委員会において作成されたガイドラインである。
 - ・ **サイバーセキュリティ経営ガイドライン**
サイバー攻撃から企業を守る観点から、経営者が認識する必要がある3原則、および経営者がサイバーセキュリティ対策実施責任者（CISO など）に指示すべき重要10項目を、IPA と経済産業省が策定したガイドラインである。
- ⑦ **CRYPTREC** (Cryptography Research and Evaluation Committees)
電子政府推奨暗号の安全性を評価・監視し、暗号技術の適切な実装法・運用法を調査・検討するプロジェクトおよび機関のことである。安全性および実装性能が確認された暗号技術は、**CRYPTREC 暗号リスト**（電子政府推奨リスト、推奨候補リスト）として公開される。

1-4-2 情報セキュリティ評価基準

- ① **コモンクライテリア** (ISO/IEC 15408)
IPA が運用する「IT セキュリティ評価及び認証制度 (**JISEC**)」において、認証機関が公開する評価基準の規格である。セキュリティ機能要件やセキュリティ保証要件について規定し、評価保証レベル (**EAL**: Evaluation Assurance Level) の基準とされる。
- ② **JCMVP** (暗号モジュール試験及び認証制度)
暗号化や署名などの機能を実装した暗号モジュールが、内部の重要情報を適切に保護していることを認証するための評価基準および制度である。
- ③ **PCI DSS** (Payment Card Industry Data Security Standard)
クレジットカード情報および取引情報を保護するためのセキュリティ基準（認定評価基準）である。**ペネトレーションテスト**（擬似侵入テスト）の実施や、使用するカードの耐タンパ性などについても規定している。法令により、クレジットカード事業者や加盟店は PCI DSS 準拠（又は加盟店の場合は非保持化）が義務付けられている。

2 情報セキュリティ対策

情報セキュリティ対策は、情報システムだけでなく、人間をとりまく環境を含めて総合的に実施する必要がある。ISMS では、情報セキュリティ対策の観点として、物理的セキュリティ、人的セキュリティ、技術的セキュリティの三つをあげている。

2-1 物理的セキュリティ対策

物理的セキュリティとは、施設・設備などの環境管理や、文書・記録媒体などの管理に関わるセキュリティである。物理的セキュリティ対策とは、RASIS の S (Security ; 機密性) を高めることであるが、RAS (信頼性, 可用性, 保守性) を高める **RAS 技術** (二重化技術, ミラーリング, 障害抑止) の導入も、物理的セキュリティの一環として考える。

物理的セキュリティ対策の代表的な種類および関連技術は、次のとおりである。

- ・ **ゾーニング** (zoning ; 施設、設備のセキュリティ区画の設定)
セキュリティ区画として管理すべき場所を設定し、耐震・耐火設備などを導入して災害などの物理的脅威に備える。
- ・ **入退室管理**
セキュリティ区画への入退室者や外来者を管理する。ID カードなどを用いて入室を制限し、入室者の氏名・入退室日時・入室目的などを記録する。セキュリティ区画は**施錠管理**し、入退室者記録用の**監視カメラ**を設置する。
- ・ **遠隔バックアップ**
災害などに備え、遠隔地にバックアップを保管する。
- ・ **セキュリティワイヤ**
機器を柱などにつなぎ、盗難を防止する装置である。
- ・ **認証デバイス**
入退室管理の利用者認証などに使用する認証用機器で、利用環境などに応じて導入を検討し、セキュリティゲートなどに設置する。
- ・ **USB キー**
固有の認証情報が記録され、機器に差し込むだけで利用者認証ができる認証デバイスである。LAN で利用される IEEE 802.1X のクライアント認証などに使用されるが、内蔵メモリの容量などに関する規定はない。
- ・ **廃棄**
物理的破壊やランダムなビット列での複数回の上書きなどにより、廃棄した機器からの情報漏えいを防止する。

2-2 人的セキュリティ対策

人的セキュリティとは、人的な手続・管理・運用ルールなどに対するセキュリティであり、関係者によるセキュリティ違反などの事故・事件を防止することが目的である。

組織に対しては、情報セキュリティに関して権限・責任をもつ、最高情報セキュリティ責任者（**CISO**: Chief Information Security Officer）、部・課単位の情報セキュリティ担当官、情報システム管理者を置いて、管理・監視する。一方、組織の要員に対しては、**情報セキュリティポリシー**として、**組織における内部不正防止ガイドライン**などを参考に、次のような項目を含む**社内規程**を策定し、管理・監視する。

- ・ 要員の採用・派遣・委託契約を行うときの審査
- ・ 機密保持契約／守秘義務
- ・ 情報セキュリティ教育／訓練
- ・ セキュリティ事故・事件（インシデント）が発生したときの対処／連絡
- ・ セキュリティ違反に対する懲戒手続
- ・ 情報システムの取扱い
- ・ アカウント管理／パスワード管理

人的セキュリティのPDCAサイクルでは、ISMSの計画（Plan）段階で情報セキュリティポリシーに規定し、実行（Do）段階でセキュリティ教育・訓練を実施する。さらに、要員のセキュリティ遵守状況を**ログ管理**や監視によって点検（Check）段階でチェックし、不備や不正があった場合には処置（Act）段階で対処・是正する。

また、人的セキュリティ対策で最も重点を置くべきことは、関係者からの情報漏えいである。人的セキュリティにおける具体的な**情報漏えい対策**には、次のようなものがある。ここであげている対策は、主に**不正のトライアングル**（機会・動機・正当化）の機会を減らす（重要な情報を手に入れにくくする）ものである。

- ・ 情報セキュリティ啓発（教育、訓練、資料配布、メディア活用）により、正規社員や派遣社員など、すべての関係者のセキュリティ意識を向上させる。
- ・ 重要情報の伝達手順・廃棄手順などの社内規程を定め、遵守させる。また、遵守状況を監視し、改善が必要な点があれば適切に指導する。
- ・ “need-to-know（必要な人だけに情報を開示する）”の原則に従って、利用者ごとに適切なアクセス権を設定する（最小権限）。
- ・ あらゆる作業（データアクセス、プログラムインストールなど）が許可される**特権的アクセス権**（管理者権限）については、割当て・管理を厳重に行う。
- ・ **クリアデスク**（書類などを机上にし出したままにしないこと）／**クリアスクリーン**（PCなどを使用できる状態で放置しないこと）を徹底する。
- ・ 夜間や休日など、相互監視ができない環境における単独作業を制限する。

2-3 技術的セキュリティ対策

技術的セキュリティとは、ハードウェア、ソフトウェア、ネットワークなどの技術を利用するセキュリティである。ここでは、技術的セキュリティ対策の種類および関連技術について説明する（実装技術の詳細は「2-4 セキュリティ実装技術」で説明する）。

・不正アクセス対策

アクセス権のない利用者が、システムやデータを不正に使用するのを防ぐ対策である。なりすましを防ぐ**パスワードクラック対策**や、他人のPCやシステムなどに侵入するのを防ぐ**クラッキング対策**などがある。

・パスワードクラック対策

ほとんどのパスワードクラックの攻撃は、利用者のパスワードが要件を満たしていれば防げる。そのためには、パスワードを登録するときに、要件を満たしているかを検査する仕組みなどが必要である。

攻撃名	パスワード要件
総当たり攻撃	必要十分な長さにする。
逆総当たり攻撃	英数字、記号などを組み合わせた、意味のない文字列にする。
辞書攻撃、類推攻撃	
パスワードリスト攻撃	再利用しない。

また、パスワードを保存するときは盗用防止のため、暗号化又はハッシュ値に変更して記録する。このとき、公知のハッシュ関数を使用する場合は事前計算攻撃への対策として、次のような方法を用いる。

・ソルトを用いる方式

利用者ごとにランダムな文字列（ソルト）を割り当て、パスワードとソルトを連結した文字列からハッシュ値を求める。利用者ごとにソルトが異なるため、攻撃者が一つのパスワード候補から求めるハッシュ値の数が膨大になる。

・ストレッチング

ハッシュ値を求める処理を、指定回数だけ繰り返す方式である。ハッシュ化を何回も行うため、攻撃者が一つのパスワード候補からハッシュ値を求めるのに時間がかかる。

・クラッキング対策

クラッキングでは、技術的な手法で侵入を試みる前に、アドレススキャンやポートスキャンを行って調査することが多い。その対策として、ポートスキャナなどで、悪用されそうなポート（サービス）を事前に検査し、対処しておく必要がある（ポートスキャナは、不要なサービスの検査にも使用されるソフトウェア）。また、アクセスログなどを定期的に確認（解析）し、攻撃の痕跡を見つけることも有効である。

- ・ **マルウェア対策／コンピュータウイルス対策**

マルウェア（コンピュータウイルス）に対する、感染の予防・検知・被害の拡大防止・復旧を技術的に行う対策である。この対策には、ウイルス対策やスパイウェア対策など、何種類かのセキュリティ対策機能をもつ**セキュリティ対策ソフト**を利用する場合が多い。また、感染経路は電子メールの添付ファイルなど、ネットワーク経由の場合が多いため、ネットワークセキュリティを実現するソフトウェアなどと併用すると効果的である。

- ・ **ウイルス対策ソフト（ワクチンソフト）**

コンピュータウイルスの検出・駆除などを行うソフトウェアである。コンピュータウイルスの代表的な検出手法として、次の二つがある。

- ・ **パターンマッチング方式**

ウイルスの特徴的な部分を“シグネチャコード（パターン）”としてデータベース（パターンファイル、定義ファイル）化し、検査対象のプログラムコードと照合（マッチング）する方式である。この方式では、登録されていないウイルスは検出できないため、常に最新のパターンファイルに更新しておく必要がある。

- ・ **ルールベース方式**

あらかじめ定めたルールベースで、プログラムの挙動を解析してウイルスを検出する方式であり、次の二つの方法がある。一つは、バイナリコードから逆アセンブルすることなどで得られるソースコードを静的に解析する**ヒューリスティック法**である。もう一つは、**サンドボックス**（不審なプログラムを実行する仮想環境、又は仮想環境を利用したセキュリティ機構）などで検査対象を実行して、ウイルスと判断されるような挙動を検出（動的に解析）する**ビヘイビア法**である。

なお、ウイルス感染時の初動措置（ネットワークからの切断、管理者への連絡など）は、情報セキュリティポリシーに規定して人的セキュリティ対策として実施する。このとき、発見者自身がウイルスを駆除したり、メモリなどを初期化したりすることは、原則、行ってはならない。その理由は、感染したウイルスの種類や感染経路、被害状況などを調査・分析できなくなるからである。

- ・ **アップデート**

メーカーやソフトウェアベンダが提供する、ソフトウェアのセキュリティなどの不具合に対応する修正プログラム（**パッチプログラム**）により、OSやアプリケーションの脆弱性をなくしていく対策である。新たに判明したマルウェアやコンピュータウイルスの感染を防止する効果もある。なお、アップデートには最新のものを使用することが望ましいが、**OS アップデート**などの場合、他のプログラムの動作に影響を与えることもあるため、事前の確認が大切である。

- ・ **セキュアブート**

PC 起動時に OS やドライバのデジタル署名を検証することで、OS 起動前のマルウェア（許可のないソフトウェア）の実行を防ぐ技術である。

- ・ **SPAM 対策（スパムメール対策）**

SPAM を受け取らない対策と、SPAM の不正中継の踏み台として利用されない対策がある。なお、SPAM を受け取ってしまった場合は、何もしない（開封／返信しない）で削除するのが最適な対策である。

- ・ **ブラックリスト／ホワイトリスト**

受信拒否アドレスが登録されたブラックリストや、受信許可アドレスが登録されたホワイトリストを使用する方式である。

- ・ **SPF** (Sender Policy Framework)

あるドメインの電子メールを送信できる正規メールサーバの IP アドレスをリストで管理する、送信元ドメイン認証方式である。ドメインと無関係なメールサーバから送られてきた電子メールを、自動的に受信拒否できる。SMTP の拡張仕様で、RFC 4408 として規定されている。

- ・ **OP25B** (Outbound Port 25 Blocking)

ISP などが、外部ネットワークとの境界にあるルータなどによって、内部ネットワークから外部への TCP ポート 25 番 (SMTP) の通信を禁止する方式である。OP25B が有効になっていると、SMTP 通信がブロックされ、SPAM の不正中継などを防止できる。

- ・ **メールヘッダインジェクション対策**

電子メールのヘッダ情報（宛先など）を不正に書き換え、不正中継の踏み台とするメールヘッダインジェクション攻撃を防止する。

- ・ **DKIM** (DomainKeys Identified Mail)

受信した電子メールが、正当な送信者からのもので、改ざんされていないかどうかを調べる、デジタル署名方式の送信ドメイン認証技術である。送信時に、秘密鍵によって生成した署名情報を送信電子メールのヘッダに記述し、受信時に署名ドメインの DNS サーバ上の公開鍵によって署名を検証する。これにより、電子メールの送信ドメインと本文の正当性（送信途中で改ざんされていないこと）の両方を確認できる。

- ・ **コンテンツフィルタリング**

Web サイトなどに公開されているコンテンツの閲覧を、不適切なキーワードが含まれているかなどのフィルタリングによって規制する技術である。有害情報を含むコンテンツの閲覧を禁止する場合と、安全が確認できないコンテンツへのアクセスを禁止するセキュリティ的な目的で利用する場合がある。

- ・ **URL フィルタリング**

指定された URL のコンテンツ閲覧を禁止する技術である。閲覧禁止の URL リスト（フィルタ）を、最初から用意してある製品も多い。

- ・ **ハニーポット**

不正侵入の手段や経路、ウイルスの挙動などを調査・分析する目的で、インターネット上に設置される“おとり”のサーバや機器である。ハニーポットからは、外部攻撃ができない（被害が拡大しない）ように設定されている。

- ・ **電子透かし**

文書、静止画、動画などのコンテンツ中に、通常の閲覧・再生では知覚できないデータを埋め込む技術である。専用の検出ソフトに読み込ませると情報が表示されるので、不正コピーやデータの改ざんなどを検出できる。なお、秘密にしたい情報を他者に気付かれることなく埋め込む（情報を隠蔽する）技術を、**ステガノグラフィ**という。

- ・ **携帯端末セキュリティ**

携帯端末には、電話帳などの個人情報や機密情報などが記録されているため、セキュリティ対策が必要である。携帯端末は持ち運びが簡単なので、機器本体や**SIMカード**（加入者を特定するための固有の番号が記録されたICカード）などの紛失・盗難に備えることが重要となる。

- ・ **携帯電話／スマートフォン**

一定時間操作しないとキーロックがかかる設定を利用する。ロックの解除方法には、パスワード方式のほかに指紋認証や顔認証などもある。また、紛失・盗難時には、遠隔操作でキー操作を無効にする機能や、端末自体を初期化する機能なども活用できる。

- ・ **タブレット端末**

タブレット端末には、ノートPCと同様のセキュリティ対策が必要である。タブレット端末を狙ったウイルスなどに対しては、ウイルス対策ソフトなどのインストールも必須である。なお、ノートPCやタブレット端末はログインパスワードを設定できるが、紛失・盗難時にはハードディスク装置を取り外すことができるため、ハードディスクパスワードやディスク暗号化などの対策も必要である。

- ・ **MDM** (Mobile Device Management)

従業員が個人で所有する携帯端末（情報機器）を業務に使用する **BYOD** (Bring Your Own Device) は、セキュリティ設定の不備などによる情報セキュリティリスクが増大する。そこで、企業が貸与するスマートフォンなどを含め、MDMによりセキュリティポリシーに従って一元的に設定したり、利用状況を一元管理したりする。

- ・ **デジタルフォレンジックス**

フォレンジックス (forensics) とは、法医学、鑑識、科学捜査という意味であり、デジタル鑑識のことである。不正アクセス、情報漏えいなどのコンピュータ犯罪や法的紛争の原因究明・捜査に必要な機器、データ、電子記録を収集・分析し、法的な根拠を明らかにする技術の総称である。

2-4 セキュリティ実装技術

セキュリティ実装技術とは、攻撃（脅威）に対してネットワークやデータベースなどの攻撃対象ごとに実装される技術的対策のことである。それぞれの攻撃に対して、もっとも適正なセキュリティ技術で対応することが重要である。

2-4-1 セキュアプロトコル

セキュアプロトコルとは、通信データの盗聴や不正接続などを防ぐためのプロトコルである。TCP/IP ネットワークでは、さまざまなセキュアプロトコルが用意されている。

- **IPsec** (Security Architecture for Internet Protocol)

IPv6 の標準仕様となっているセキュアプロトコルである。インターネット層（ネットワーク層）で“暗号ペイロード (ESP)”と“認証ヘッダ (AH)”によるパケット暗号化や認証を行うので、データの秘匿性／正当性が向上する。

- **SSL** (Secure Sockets Layer)/**TLS** (Transport Layer Security)

アプリケーション層と TCP 層の間で、クライアントとサーバ間の認証（デジタル署名）と、通信内容の暗号化（セッション鍵暗号方式）を行うセキュアプロトコルである。SSL をもとに IETF で標準化が行われ、RFC 2246 として TLS が策定されたため、現在では **SSL/TLS** 又は単に TLS と表記される場合が多い。

元のプロトコル	SSL/TLS と組み合わせたセキュア通信
HTTP	HTTP over TLS (HTTPS)
FTP	FTP over TLS (FTPS)
SMTP	SMTP over TLS (SMTPS)
POP3	POP3 over TLS (POP3S)

- **SSL/TLS アクセラレータ**

サーバの暗号処理負荷を軽減し、高速化を図るハードウェアである。

- **SSH** (Secure SHell)

TCP 層とアプリケーション層で、パスワードやデータの暗号化を行う。UNIX 系のコマンド (rsh) のセキュリティを強化したプロトコルである。

- **S/MIME** (Secure / Multipurpose Internet Mail Extensions)

公開鍵暗号方式を用いて、電子メールの暗号化とデジタル署名による認証を行うセキュアプロトコルである。

- **PGP** (Pretty Good Privacy)

セッション鍵暗号方式（共通鍵：IDEA，公開鍵：RSA）を用いて電子メール本文を暗号化する，個人向けのセキュアプロトコルである。

- **APOP** (Authenticated POP)

POP にセキュリティ機能を付加したプロトコルで，メールサーバへのログイン時にパスワードを暗号化する。ただし，電子メール本文は暗号化されない。

- **SET** (Secure Electronic Transactions)

暗号化技術とデジタル署名を用いた，クレジットカード決済プロトコルである。販売店とカード会社で参照できる情報が限定され，機密性が高い。

また、「2-3 技術的セキュリティ対策」で説明した SPF や DKIM で利用されるプロトコルを**認証プロトコル**という。認証プロトコルは、なりすましによる不正接続やサービスの不正利用を防止する目的で利用される、セキュアプロトコルの一種といえる。

- **SMTP-AUTH**

電子メール送信時に、メール送信サーバによって利用者の認証を行うプロトコルである。SMTP は利用者認証の仕組みがないため、メールサーバを不正利用した迷惑メールの増加が問題となっていた。SMTP-AUTH では、あらかじめユーザ ID とパスワードを指定しておくことにより、利用者認証を行う。

- **OAuth**

あらかじめ信頼関係を構築したサービス（アプリケーション）間で、利用者の同意のもとに、利用者の権限受渡しを行うときの認証用プロトコルである。OAuth では、利用者はサービスにパスワードを教えないでアクセス権を委譲し、適用範囲を指定したり、有効期限を設定したりすることができる。

- **DNSSEC** (DNS SECurity Extensions)

DNS のセキュリティを向上させるための拡張仕様である。通常の DNS サーバは通信相手を認証することができないが、DNSSEC はデータ作成元やデータの完全性をデジタル署名によって確認できる。応答を送信する DNS サーバは秘密鍵を使って署名し、受信者は公開鍵で検証する。

- **EAP** (PPP Extensible Authentication Protocol ; PPP 拡張認証プロトコル)

リモートアクセスによる利用者認証に用いられるプロトコルである。PPP に認証機能を追加した EAP による認証方法には、次のような方式がある。

- **EAP-TLS** (EAP Transport Layer Security)

サーバとクライアントの双方でデジタル証明書（サーバ側は**サーバ証明書**、クライアント側は**クライアント証明書**）を利用する。IETF によって、RFC 2716 として定義されている。

- **PEAP** (Protected EAP) / **EAP-TTLS** (EAP Tunnel mode TLS)

どちらも、サーバの認証には**サーバ証明書**を利用し、クライアントの認証にはユーザ ID とパスワードを利用する。PEAP は Microsoft, Cisco Systems, RSA Security の 3 社で開発された。

- **CHAP** (Challenge Handshake Authentication Protocol)

チャレンジレスポンス方式を用いて、ネットワーク上にパスワードそのものを流さずに認証するためのプロトコルである。

なお、利用者認証を目的とする認証プロトコルに対して、利用者を特定しないでサービスを提供する **anonymous FTP** というプロトコルがある。このプロトコルは、ユーザ ID に“anonymous（匿名）”と入力することで、誰でも FTP サービスを利用することができる。ユーザ ID の登録・管理などの手間が省けるため利便性は高いが、クラッカによる攻撃対象となりやすいので十分なセキュリティ対策が必要となる。

2-4-2 ネットワークセキュリティ

ネットワークセキュリティとは、ネットワークを利用した攻撃に対するセキュリティの総称である。ネットワークセキュリティでは、内部ネットワークへの不正侵入（不正アクセス）を防止するネットワークアクセス制御が重要となってくる。

(1) ファイアウォール

ファイアウォールは、LAN などの内部ネットワークとその他のネットワークの境界に設置して、内部ネットワークを不正アクセスから保護する仕組みである。ファイアウォールによって内部セグメントと隔離されたセグメントを、**DMZ** (DeMilitarized Zone；**非武装セグメント**) という。外部に公開するサーバを DMZ に設置することで、ファイアウォールによって外部からの不正なアクセスを排除でき、万一、公開サーバが攻撃されても、内部セグメントへの被害を防止できる。ただし、メールサーバなどの公開サーバに十分な対策を施しておかないと、第三者中継で“踏み台”にされることがある。

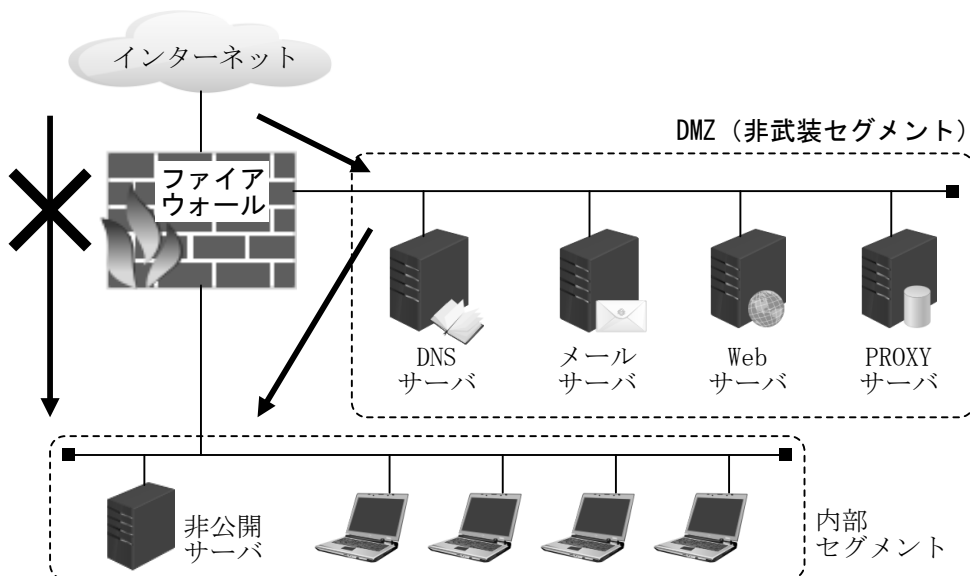


図 6-2 ファイアウォールの構成図（例）

代表的なファイアウォール技術として、**パケットフィルタリング**方式がある。パケットフィルタリング方式では、パケット中の送信元の IP アドレス／ポート番号、宛先の IP アドレス／ポート番号などの設定ルールに基づいて、パケット通過の許可／拒否を行う。主に、外部からの不要な通信を制御するために IP、TCP、UDP、ICMP などのヘッダに含まれる情報をもとに制御を行う。

例えば、図 6-2 のメールサーバに関連する設定ルールは、次のようになる。通常は、送信元と宛先には IP アドレスを使用するが、ここではわかりやすいように機器名やネットワーク名を使用している。

	送信元	宛先	宛先ポート番号	動作
①	インターネット	メールサーバ	25 (SMTP)	許可
②	メールサーバ	インターネット	25 (SMTP)	許可
③	内部セグメント	メールサーバ	25 (SMTP)	許可
④	内部セグメント	メールサーバ	110 (POP3)	許可
⑤	すべて	すべて	すべて	拒否

ファイアウォールはパケットを受信すると、合致するルールを設定ルールの順に確認し、合致したルールの動作を実行する。図 6-2 の上記の設定ルールでは、インターネット側からメールサーバに電子メール（以下、メールと略記）が送られてくると、メール送信のプロトコルは SMTP（ポート番号 25）なので①のルールに合致する。合致したルールの動作は“許可”なので、このパケットはファイアウォールの通過を許可され、メールサーバに届く。しかし、インターネット側からメールサーバに HTTP 要求パケットが送られてくると、HTTP 通信のプロトコルは HTTP（ポート番号 80）なので①～④のいずれにも該当せず、⑤のルールに合致する。合致したルールの動作は“拒否”なので、このパケットはファイアウォールの通過を拒否され、メールサーバには届かない（破棄される）。また、④のルールで内部セグメントの PC からメールサーバにメール受信要求パケットが通過すると、メールサーバはメールを送り返してくる（これをリプライパケットという）。このリプライパケットがファイアウォールを通過できるルールは、**動的パケットフィルタリング**によって自動的に追加される（**静的パケットフィルタリング**では、リプライパケットのルール設定も必要になる）。

パケットフィルタリング方式では、送信するパケットの内容までは検査できないので、**アプリケーションゲートウェイ**方式を利用することがある。この方式は、ネットワークの境界に配置されるゲートウェイでアプリケーションを中継するプログラム（プロキシプログラム）によって、内部・外部間の通信を制御（許可／拒否）する。中継プログラムは、アプリケーションの種類ごとに用意する必要があるが、アプリケーションのコマンドやデータについて詳細に制御し、高いセキュリティを実現することができる。

〔リバースプロキシ〕

リバースプロキシは、特定サーバの代わりにクライアントからの要求を受け付け、特定サーバに中継する。また、中継時にパケットの内容などをスキャンする機能を組み込むことで、特定サーバへのアクセスを制限するセキュリティ対策にも利用できる（本来は、クライアントからの要求への応答代行によるサーバの負担軽減や、コンテンツのキャッシュ保存による応答高速化の目的で利用される）。

(2) IDS (Intrusion Detection System ; 侵入検知システム)

IDS (侵入検知システム) は、ネットワーク接続機器などへの不正侵入を検知し、管理者への通知やログの収集・解析を行うシステムである。あらかじめ登録してある侵入パターンに該当するアクセスパターンや、通常の運用におけるアクセスパターンと異なるものなどを検知する。

- ・ **ホスト型 IDS (HIDS)**

特定サーバやクライアントのアクセス状況を監視し、不正侵入を検知する。

- ・ **ネットワーク型 IDS (NIDS)**

ネットワークを通過するパケットを監視し、不正侵入を検知する。

なお、IDS の機能を拡張し、検知した不正侵入の接続を遮断することなどで、不正侵入を防止する **IPS** (Intrusion Prevention System ; 侵入防止システム) もある。

(3) 検疫ネットワーク

検疫ネットワーク は、内部ネットワークに接続するノート PC の安全性確認のために、独立したネットワーク (検疫ネットワーク) にいったん接続して検査し、問題があれば対処する仕組みである。一般的には認証サーバと連動して用いられ、“認証→検疫→接続”という流れで安全性を確保する。

- ・ **認証サーバ**

認証処理を専門に行うサーバで、システム全体の認証を一元化する **RADIUS サーバ** (RADIUS : Remote Authentication Dial-In User Service) などがある。この認証サーバを利用することで、一度のログイン (認証) で複数のシステム (サービス) を利用できる **シングルサインオン** なども実現できる。

(4) コールバック (call back)

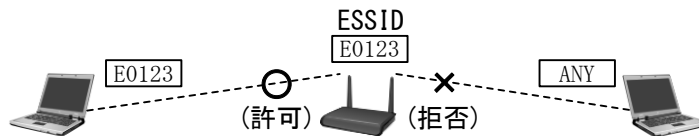
コールバック は、内部ネットワークなどに一般電話回線などの公衆回線経由で接続するときに使用される利用者確認技術である。RAS サーバ (RAS : Remote Access Service) に公衆回線経由でダイヤルアップアクセスすると、RAS サーバは発信側の呼をいったん切断し、あらかじめ RAS サーバに登録してある発信者の電話番号又は発信者 ID から利用者を特定し、再び呼を返すことで利用者確認を行う方式である。ただし、発信側の電話などの通信機器を特定するだけなので、第三者がそれらの機器を不正に使用した場合は、検出できない。そのため、一般には、ユーザ ID / パスワードによる確認など、ほかの利用者確認方法と組み合わせて利用する場合が多い。

(5) 無線 LAN セキュリティ

無線 LAN は、電波が届く範囲内であれば簡単に LAN と接続できるため、不正接続を防止するためのセキュリティ対策が重要である。

① ANY 接続拒否機能

アクセスポイントに設定した ESSID（又は SSID）と端末側の ESSID の値が一致しない（空白や“ANY”が設定されている）場合に、接続要求を拒否する機能である。



・ステルス機能（ESSID ステルス）

アクセスポイントが定期的に発信するビーコン信号を停止して、ESSID を隠蔽する機能である。ANY 接続拒否機能と併用されることが多い。

② MAC アドレスフィルタリング機能

接続を許可する端末の MAC アドレスをアクセスポイントに登録しておき、登録されていない MAC アドレスをもつ端末からの接続を拒否する機能である。



③ プライバシーパレータ機能

アクセスポイント経由での無断アクセスを防止するために、同一アクセスポイントに接続している端末同士の通信をアクセスポイントで遮断する機能である。

④ 暗号化

初期の無線 LAN では、WEP (Wired Equivalent Privacy) や WPA (Wi-Fi Protected Access) などの暗号方式が利用されていた。しかし、どちらも鍵の解析が容易な共通鍵暗号方式 RC4 を用いているため、現在は推奨されていない。

・WPA2 (Wi-Fi Protected Access 2)

共通鍵暗号方式 AES を利用した暗号化プロトコル CCMP (Counter mode with CBC-MAC Protocol) を用いる方式である。認証方式と併せ、WPA2 パーソナル (WPA2+PSK) や WPA2 エンタープライズ (WPA2+EAP-TLS) と設定されることがある (PSK はアクセスポイントと端末が事前共有鍵 (Pre-Shared Key) を用いて相互認証を行う)。

・WPA3 (Wi-Fi Protected Access 3)

WPA2 を改善し、安全性をより強固にした方式である。WPA2 で発見された脆弱性は改善されているが、新たな脆弱性も指摘されている。

(6) その他のネットワークセキュリティ

- ・ **NAT/NAPT (IP マスカレード)**

プライベートIPアドレスを、グローバルIPアドレスに変換する機能である。プライベートIPアドレスが秘匿されるため、若干のセキュリティ効果もある。

- ・ **VPN (Virtual Private Network ; 仮想私設通信網)**

複数の利用者が、それぞれ独自の専用線を保有しているかのように利用できるサービスである。窃取などを防止する効果が期待できる。

- ・ **アクセスログ解析**

アクセスログを解析することで、**アドレススキャン** (ping コマンドを繰り返し発行して接続可能な IP アドレスを探す攻撃) や、**ポートスキャン** (ポート番号を変えながらアクセスし、侵入可能なサービスを探す攻撃) の痕跡を見つける方法である。即時性はないが、効果は比較的高い。

- ・ **SIEM (Security Information and Event Management)**

ファイアウォール、プロキシサーバ/リバースプロキシサーバ、IDS 又は IPS、検疫ネットワークなどのセキュリティ機器や、Web サーバ、メールサーバなどの各種サーバからログを収集して一元的に管理し、各ログを統合的に分析して不正を検知するシステムである。

- ・ **UTM (Unified Threat Management ; 統合脅威管理)**

複数の異なるセキュリティ機能を一つのハードウェアに統合して、集中的にネットワーク管理を行う方式である。

- ・ **ペネトレーションテスト (擬似侵入テスト)**

システムへの不正アクセス (不正侵入) を実際に試みて、セキュリティ対策が有効であるかを検証する手法である。

- ・ **ファジング (fuzzing)**

ソフトウェア製品の不具合や脆弱性を発見するテスト手法である。予測不可能な入力データや問題を引き起こしそうな多様なデータを自動生成し、ソフトウェアに入力したときのソフトウェアの応答や挙動から脆弱性を検出する。

- ・ **多層防御 (多段階防御, 階層的防御)**

セキュリティ技術やセキュリティ製品を多層的に複数組み合わせることで活用することにより、情報の安全性を高める手法である。情報保証戦略の一環として、情報セキュリティ資源の配分などまで含めることもある。

- ・ **入口対策 / 出口対策**

標的型攻撃対策などで用いられる多層防御の考え方である。標的型攻撃の初期潜入段階でマルウェアの侵入を防ぐことなどが入口対策に該当し、目的遂行段階で機密情報が外部に送信されないようにすることなどが出口対策に該当する。なお、物理的セキュリティ対策 (入退室管理) で、入口 / 出口に関する対策という意味で用いられる場合もある。

- **DLP** (Data Loss Prevention)

機密データの漏えいや紛失を防止するための製品やサービスのことである。機密データの利用を監視し、不正な利用（例えば、外部への電子メール送信など）に対して警告を発したり、処理を中断させたりする。

- **ID 連携トラストフレームワーク**

現在、インターネットなどで進められている **OpenID** (OpenID 財団) や **SAML** (標準化団体 OASIS) などの **アイデンティティ連携** (複数サービス間で利用者のアイデンティティ (属性情報) を連携していく考え方。これにより、同じ ID で複数のサービスを利用できることになるため、利便性が高くなる) では、異なる組織 (サービスを提供する企業や団体など) 間でアイデンティティを相互交換することにより生じる、個人情報悪用の悪用・漏えいというリスクが問題視されている。そこで、経済産業省が提案した仕組みが ID 連携トラストフレームワークである。このフレームワークでは、どんな組織でも連携させるのではなく、ポリシーやルールを明確にした上で信頼 (トラスト) できる組織を認定し、それらを連携させることを目的としている。

- **セキュリティ対策ソフト**

PC などにインストールすることで、ウイルス対策、迷惑メール対策、パーソナルファイアウォール、コンテンツフィルタリングなどのセキュリティ機能を利用できるパッケージソフトウェアである。なお、**偽セキュリティ対策ソフト** などもあるので注意が必要である。

- **クラウドサービスのセキュリティ**

クラウドサービス (SaaS, PaaS, IaaS) では、サービスの種類によって利用者と管理者のセキュリティ対策に対する責務範囲が異なる。

	SaaS	PaaS	IaaS
アプリケーション	管理者 (責務範囲)	利用者 (責務範囲)	利用者
データ		管理者	
OS／DBMS			管理者
ハードウェア			

【情報セキュリティ訓練】

セキュリティ対策としては、実際のサイバー攻撃を模した擬似攻撃を社員などに仕掛け、対処法を実地で習得させる **情報セキュリティ訓練** も有効である。擬似的な不審メールを訓練対象者に送信する **標的型攻撃メールに対する訓練**、セキュリティの専門家で攻撃チーム (レッドチーム) を作り、現実に近い各種攻撃を仕掛けてセキュリティ対策の実効性を検証する **レッドチーム演習** などがある。

2-4-3 データベースセキュリティ

データベースセキュリティとは、データベースへの攻撃に対するセキュリティの総称である。基本的に、DBMS が提供する**機密保護機能**を利用する。これと併せて、外部媒体の利用制御や、不正アクセスの検知なども実施すると、さらに安全性が高くなる。

機密保護機能	保護内容
暗号化	データベースに記録してある内容を暗号化する。
アクセス権の設定	データベースに対するアクセス権を設定する。
パスワードの設定	利用者をユーザ ID とパスワードによって認証する。
ログファイルへの記録	利用状況などを記録し、不正使用の有無を確認する。

2-4-4 アプリケーションセキュリティ

アプリケーションセキュリティとは、アプリケーションソフトウェアの脆弱性（セキュリティホール）を突いた攻撃に対するセキュリティの総称である。情報セキュリティを企画・設計段階から確保するための方策である**セキュリティバイデザイン**や、個人情報保護を企画・設計段階から確保するための方策である**プライバシーバイデザイン**、さらにプログラミング段階でセキュリティ対策を実装する**セキュアプログラミング**などが該当する。

- ・ **WAF** (Web Application Firewall)

Web アプリケーションのセキュリティホールなどに起因する脆弱性への攻撃を遮断するための装置やソフトウェアである。問題がある通信データパターンを定義したブラックリストに該当する通信を遮断する方式などがある。

- ・ **Web システムのセキュリティ対策**

Web システムに対する攻撃への対策としては、次のような方法がある。

- ・ SQL インジェクション対策：**エスケープ処理**や**プレースホルダ**
- ・ XSS 対策：危険なスクリプトを無害化する**サニタイジング**
- ・ CSRF 対策：処理実行前にパスワード入力を促す**再認証**
- ・ バッファオーバーフロー対策：バッファ領域へのデータ出力制限

2-4-5 セキュア OS

セキュア OSとは、強制アクセス制御と呼ばれるアクセス機構及び最少権限という考え方から作られた OS で、セキュリティ機能が十分に実装されている。

- ・ **強制アクセス制御** (MAC : Mandatory Access Control)

システム管理者が設定したアクセス権を変更できない仕組みである。

- ・ **最少権限**

細かく分割された特権を、必要に応じて与える仕組みである。

第6部 演習問題

問1

JIS Q 27000:2019 の情報セキュリティにおける可用性に該当するものはどれか。

- ア 情報及び処理方法が，正確であること及び完全であることを確実にする。
- イ 情報システムの利用者が正しい利用者であることを確実にする。
- ウ 第三者に情報が漏えいしないことを確実にする。
- エ 利用者が必要なときに情報資産を利用できることを確実にする。

問2

ソーシャルエンジニアリングに該当する行為はどれか。

- ア OS のセキュリティホールを突いた攻撃を行う。
- イ ウイルスに感染したコンピュータを外部から操作する。
- ウ プログラムで解析した暗証番号を用いてコンピュータ室に侵入する。
- エ 本人を装って電話をかけ，パスワードを聞き出す。

問3

マルウェアについて，トロイの木馬とワームを比較したとき，ワームの特徴はどれか。

- ア 勝手にファイルを暗号化して正常に読めなくする。
- イ 単独のプログラムとして不正な動作を行う。
- ウ 特定の条件になるまで活動せずに待機する。
- エ ネットワークやリムーバブルメディアを媒介として，自ら感染を広げる。

問4

Web アプリケーションにスクリプトを埋め込むことが可能な脆弱性があるとき，これを悪用して，該当する Web アプリケーションの利用者ブラウザ上で不正なスクリプトを実行させる攻撃はどれか。

ア DoS 攻撃	イ SQL インジェクション
ウ クロスサイトスクリプティング	エ フィッシング

問5

共通鍵暗号の鍵を見つけ出そうとする，ブルートフォース攻撃に該当するものはどれか。

- ア 一組の平文と暗号文に対し，すべての鍵候補を一つずつ試して，鍵を見つけ出す。
- イ 平文・暗号文と鍵の関係を表す代数式を手掛かりに，鍵を見つけ出す。
- ウ 平文の一部の情報と，暗号文の一部の情報との統計的相関を手掛かりに，鍵を見つけ出す。
- エ 平文を一定量変化させたときの暗号文の変化から，鍵を見つけ出す。

問6

ある商店が，顧客からネットワークを通じて注文（メッセージ）を受信するとき，公開鍵暗号方式を利用して，注文の内容が第三者にわからないようにしたい。このとき，顧客が暗号化に使用する鍵はどれか。

ア 顧客の公開鍵	イ 顧客の秘密鍵
ウ 商店の公開鍵	エ 商店の秘密鍵

問7

パスワードを忘れてしまった社内のユーザに対して，本人確認を行った後，セキュリティ管理者がとるべき対応として，最も適切なものはどれか。

- ア 暗号化して管理しているパスワードを復号し，ユーザに電子メールで伝える。
- イ 暗号化して管理しているパスワードを復号し，ユーザに電話で伝える。
- ウ 自分のPCで管理しているパスワードをハッシュ値に変換し，秘密扱いの社内文書を使ってユーザに伝える。
- エ パスワードを初期化し，ユーザに新しいパスワードを設定させる。

問8

人間の目から取得できる情報を利用した生体認証はどれか。

ア 虹彩認証	イ 指紋認証
ウ 声紋認証	エ 掌認証

問9

メッセージ認証におけるメッセージダイジェストの利用目的はどれか。

- ア メッセージが改ざんされていないことを確認する。
- イ メッセージの暗号化方式を確認する。
- ウ メッセージの概要を確認する。
- エ メッセージの秘匿性を確保する。

問10

インターネットでソフトウェアを公開する際に、ソフトウェアの開発者がデジタル署名を添付する目的はどれか。

- ア ソフトウェアの開発者が保守に責任をもつことを保証する。
- イ ソフトウェアの使用を特定の利用者に制限する。
- ウ ソフトウェアの著作権が開発者にあることを明示する。
- エ ソフトウェアの内容が不正に変更されていないことを保証する。

問11

情報セキュリティポリシーに関する記述のうち、適切なものはどれか。

- ア 企業のセキュリティポリシーは、各セキュリティシステムに設定すべき内容の定義を目的としているので、導入するセキュリティ関連製品により内容が異なる。
- イ 企業のセキュリティポリシーは、遵守すべき行為及び判断の基準となるもので、セキュリティの取組みに対する姿勢や方向性は含まれない。
- ウ 企業のトップが、セキュリティポリシー策定の要因となっている情報システムの脆弱性を対外的に公表することが望ましい。
- エ 目標とするセキュリティレベルを達成するために、遵守すべき行為及び判断についての考え方を明確にすることが必要である。

問12

ISMS プロセスのPDCA モデルにおいて、Plan で実施するものはどれか。

ア 運用状況の管理	イ 改善策の実施
ウ 実施状況に対するレビュー	エ 情報資産のリスクアセスメント

問13

リスクアセスメントに関する記述のうち、適切なものはどれか。

- ア 考えられる全てのリスクに対処することは時間と費用がかかりすぎるので、損失額と発生確率を予測し、リスクの大きさに従って優先順位を付けるべきである。
- イ リスク分析によって評価されたリスクに対し、全ての対策が完了しないうちに、繰り返しリスク分析を実施することは避けるべきである。
- ウ リスク分析は、将来の損失を防ぐことが目的であるから、過去の類似プロジェクトで蓄積されたデータを参照することは避けるべきである。
- エ リスク分析は、リスクの発生による損失額を知ることが目的であり、その損失額が大きいものに対して最優先で対策を検討すべきである。

問14

JPCERT/CC に関する説明として、適切なものはどれか。

- ア 暗号技術の適切な実装法・運用法を調査するプロジェクトである。
- イ 情報セキュリティ早期警戒パートナーシップの調整機関である。
- ウ 独立行政法人 情報処理推進機構の管轄下にあるセキュリティセンターである。
- エ 内閣官房に設置された内閣サイバーセキュリティセンターである。

問15

ウイルス対策ソフトに関する記述のうち、適切なものはどれか。

- ア ウイルス対策ソフトの定義ファイルは、各種ウイルスの先頭から 16 バイト又は 32 バイトのコードをデータベース化したものである。
- イ ウイルス対策ソフトの定義ファイルを用いた検知処理は、既知のウイルスを検出し、ウイルス名を特定するのに有効な手法である。
- ウ ウイルスに感染したファイルの大きさが感染前の大きさと同じならば、ウイルス駆除によって感染前のファイルに戻すことができる。
- エ 不正な動作を識別してウイルスを検知する方式では、動作特性からウイルス名を特定する。

問16

OP25B に関する説明として、適切なものはどれか。

- ア インターネット上に脆弱性があるように見せかけたサーバを設置して、不正侵入の手段や侵入経路の情報を収集する。
- イ 電子メールのヘッダに署名情報を記録することで、電子メールの送信元ドメインを認証する。
- ウ 内部ネットワークから Web サイトへのアクセスを、コンテンツをフィルタリングすることで制限する。
- エ 内部ネットワークから外部メールサーバへの電子メール送信を、ポート番号によってフィルタリングする。

問17

クライアントとサーバ間の認証と、通信データの暗号化機能を兼ね備えたセキュアプロトコルはどれか。

- ア APOP イ EAP ウ OAuth エ SSL

問18

リバースプロキシの役割として、適切なものはどれか。

- ア クライアントの代わりに、サーバへのアクセス要求を送信する。
- イ サーバの代わりに、クライアントからのアクセス要求を受け付ける。
- ウ ネットワークに接続するPCの安全性を確保する。
- エ ネットワークへの不正侵入を検知する。

問19

WPA2 に関する説明として、適切なものはどれか。

- ア Web アプリケーションの脆弱性への攻撃を遮断するソフトウェアである。
- イ 内部アドレスを外部アドレスに 1 対 1 で変換する方式である。
- ウ 複数の異なるセキュリティ機能を統合して、ネットワークを集中管理する方式である。
- エ 無線 LAN の暗号化方式である。

第7部

データ構造とアルゴリズム

1 データ構造

コンピュータでデータを処理するには、プログラムで指示を与える必要がある。このとき、処理内容に応じてデータの取扱いを決めておくと効率が良い。この章では、データを記録する構成（形態）であるデータ構造について学習する。

1-1 配列

配列（array；**アレイ**）は、複数の同一形式のデータを、まとめて取り扱う場合に使用されるデータ構造である。一つひとつのデータ（要素）を、番号（**添字**又は**指標**）で区別する。

【配列要素の表記方法】

- ・添字は、0から始まる場合と1から始まる場合がある。本書では、見た目のわかりやすさを考慮して、1から始まる表記で統一する。
- ・添字の囲み方には、（ ）や[]などがある。本書では、（ ）で統一する。

例1． 配列を使用しないで、3科目の得点を0にする。

英語	国語	数学

<処理>

- ① 0 → 英語
- ② 0 → 国語
- ③ 0 → 数学

例2． 配列を使用して、3科目の得点を0にする。

得点	得点(1)	得点(2)	得点(3)

<処理>

- ① 0 → 得点(1)
- ② 0 → 得点(2)
- ③ 0 → 得点(3)

配列を使用するメリットは、複数のデータに対して同一処理を行う場合、添字の値を変更するだけで処理を記述できるという点である。例えば、例2の処理は、添字を変数Iで表すことにより、次のように記述することができる。

変数Iを1～3に変えながら、次の処理を繰り返す。

0 → 得点(I)

このように、配列構造を使用すると、処理手順を簡単に表記できる。この例では、処理対象が少ない（3科目）ので違いをあまり感じられないかもしれない。しかし、もっと大量のデータ（例えば、20科目）になると、効率の良さがよくわかるはずである。

(1) 1次元配列

1次元配列は、同一形式のデータを一行に並べて扱うデータ構造である。一般的に配列という場合は、1次元配列のことを意味する。

例. 4科目の試験の結果（得点）を記録する1次元配列

	1	2	3	4
得点	得点(1)	得点(2)	得点(3)	得点(4)

(2) 多次元配列

多次元配列は、配列要素を複数の関連で取り扱うデータ構造である。例えば、2次元配列なら、要素を縦と横の二つの関連で取り扱うことになる。したがって、要素を指定する添字の数も複数（次元数と同じ数）必要になる。

例. 4科目の試験を受けた3人の結果（得点）を記録する2次元配列

得点	1	2	3	4	
1	得点(1, 1)	得点(1, 2)	得点(1, 3)	得点(1, 4)	: 1人目
2	得点(2, 1)	得点(2, 2)	得点(2, 3)	得点(2, 4)	: 2人目
3	得点(3, 1)	得点(3, 2)	得点(3, 3)	得点(3, 4)	: 3人目

(3) 構造体配列

構造体配列は、配列要素が異なる形式のデータの集合になっているデータ構造である。ファイルに記録するレコードの考え方と同じであるため、**レコード型配列**ともいう。

例. 3人の受験者の受験番号、氏名、合計点を記録する構造体配列

	受験番号	氏名	合計点	
1	受験番号(1)	氏名(1)	合計点(1)	: 1人目
2	受験番号(2)	氏名(2)	合計点(2)	: 2人目
3	受験番号(3)	氏名(3)	合計点(3)	: 3人目

一般的な配列は、メモリ（主記憶）上の連続した領域に固定要素数で確保される（これを**静的配列**という）。ただし、プログラム言語によっては、配列の要素数を処理中に変動させることが可能な**動的配列**を利用できるものもある。

また、配列は、添字を使用した直接アクセスが可能なデータ構造であるため、格納するデータから格納場所（添字）を求める**ハッシュ表**として利用することもできる。格納場所（添字）の求め方としては、直接編成ファイルでレコードキーからレコードアドレスを求めるときに使用する、**除算法**、**重合せ法**、**基数変換法**が応用できる（専用の**ハッシュ関数**も使用できる）。このとき、シノニムの対処法についても検討する必要がある。

1-2 リスト

リスト（連結リスト）は、データの並びをポインタ（要素が記録されている位置＝アドレス）によって決定するデータ構造である。

TOP…先頭要素へのポインタ

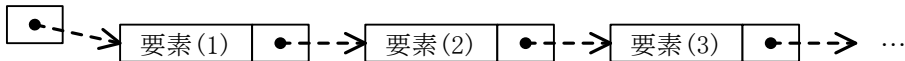


図 7-1 リストのイメージ

リストは、配列を利用して表現することもできる。この場合、ポインタには、次の要素を示す添字を記録する。配列を利用したリストの例は、図 7-2 のとおりである。

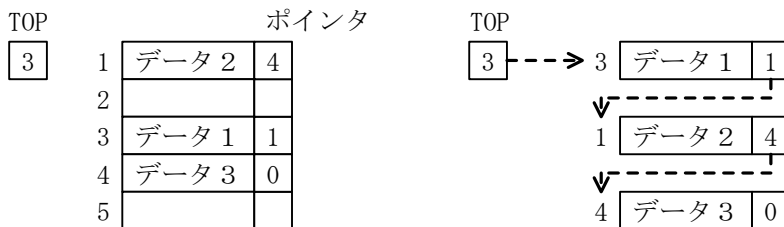


図 7-2 配列を利用したリストの例

配列はデータの記録順が意味をもつデータ構造であるため、データの挿入や削除が配列全体の操作となって効率が悪い。これに対して、データの記録順に意味がないリストの場合、ポインタを付け替えるだけでデータの挿入や削除を簡単に行うことができる。

例えば、図 7-2 のリストのデータ 1 とデータ 2 の間に、新たに要素(5)に追加したデータ X を挿入するには、図 7-3 のようにデータ 1（挿入データの直前にあるデータ）のポインタとデータ X（挿入データ）のポインタを付け替えればよい。

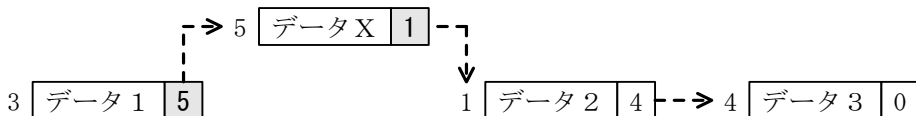


図 7-3 リストにデータを挿入する例

一方、図 7-2 のリストからデータ 2 を削除するには、図 7-4 のようにデータ 1（削除データの直前にあるデータ）のポインタを付け替えればよい。ただし、データ 2 はリストから論理的には削除されるが、データ自体は残っているので別途、物理的に削除する必要がある。

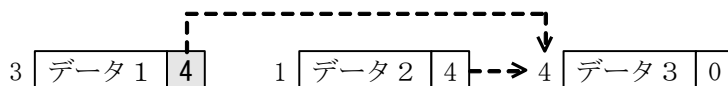


図 7-4 リストからデータを削除する例

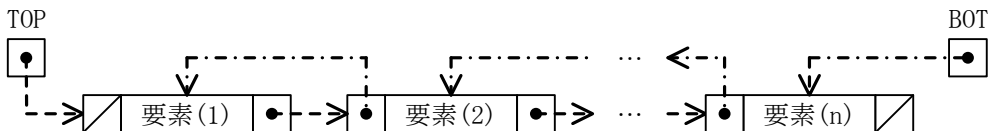
(1) 単方向リスト

単方向リストは、データを片方向にしかたどれないリストである。**線形リスト**，単リスト，片方向リストともいう。各要素は，次の要素の記録位置を示すポインタをもっているだけなのでデータ量は少なくて済むが，逆方向にたどれないため用途は限定される。リストの最後尾要素のポインタには，終わりを表す値（NULL や 0 など）が記録される。



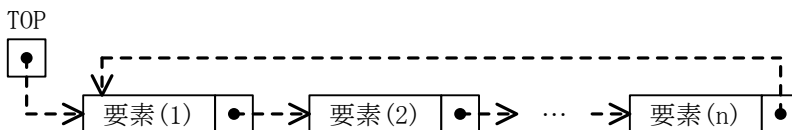
(2) 双方向リスト

双方向リストは，次の要素の記録位置を示す後ポインタと前の要素の記録位置を示す前ポインタをもつことで，データを両方向にたどれるリストである。双リスト，両方向リストともいう。要素を両方向にたどれるため，データの利用方法は広がるが，ポインタが増えるためデータ量が多くなり，処理（ポインタの付替え）も複雑になる。



(3) 環状リスト

環状リストは，全要素が環状につながっているリストである。環状リストを単方向リストで実現する場合は，最後尾要素のポインタに先頭要素へのポインタを記録する。



一般的なリストは，必要に応じて記録領域を動的に確保すればよいので，扱うデータ数を予測できないときにも利用しやすいデータ構造である。しかし，要素をたどる順次アクセスしかできないので，特定の要素に対する処理が中心となる場合には適さない。なお，単方向リストや環状リストでも，リストの末尾要素に対する処理（例えば，リストの末尾へのデータの追加など）が多い場合は，双方向リストのように末尾ポインタを利用することもある。ただし，末尾要素から前にたどることはできないので，大幅に効率が良くなるわけではない（リストの末尾要素の削除などは，ほとんど効率が変わらない）。

1-3 スタックとキュー

スタックと**キュー**は、データの利用方法をモデル化するデータ構造である。**問題解決型データ構造**とも呼ばれ、配列などで実現される。

1-3-1 スタック

スタックは、後から記録したデータを先に取り出す **LIFO** (Last-In First-Out ; 後入れ先出し) 方式のデータ構造である。箱にデータを積んだり、箱からデータを取り出したりするデータ構造と考えればよい。データを出し入れする側が同じなので、直前に格納したデータが先に取り出される。スタックにデータを格納することを**プッシュ** (PUSH)、スタックからデータを取り出すことを**ポップ** (POP) という。

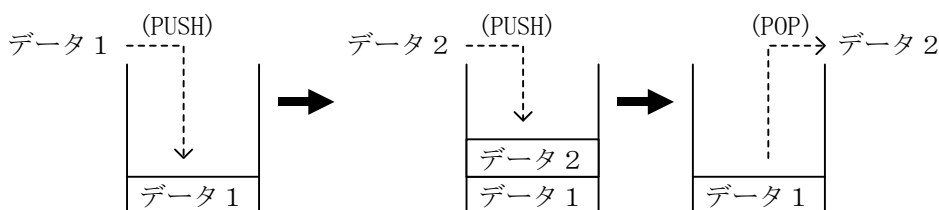
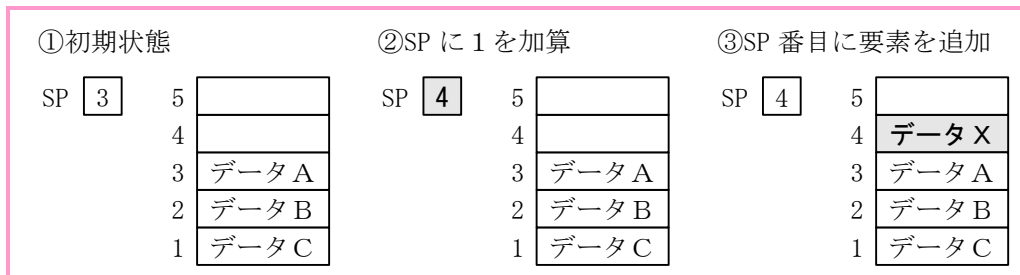


図 7-5 スタックのイメージ

スタックを実現するには、**スタックポインタ** (SP : Stack Pointer) を使用する方法が一般的である。スタックポインタは、現時点でのスタックの最上位要素の位置 (配列構造なら添字) を表している。データを格納する場合は、スタックポインタの値を更新 (一般的には+1) した位置に記録する。逆に、データを取り出す場合は、スタックポインタの位置からデータを取り出した後、スタックポインタの値を更新 (一般的には-1) する。話をわかりやすくするために、配列を逆にした図でデータを格納 (プッシュ) する処理を表現すると、次のようになる。



スタックは、関数呼出しや**再帰**などで、戻り番地や処理途中のデータを一時的に保存するのに適している。また、OS やアプリケーションプログラムなどは、メモリ (主記憶) にスタック領域を確保して利用している。

1-3-2 キュー

キューは、先に記録したデータを先に取り出す、**FIFO** (First-In First-Out ; 先入れ先出し) 方式のデータ構造である。データの入口と出口が異なる、筒のようなデータ構造と考えればよい。キューは、行列に並ぶイメージになることから、**待ち行列**とも呼ばれる。キューにデータを格納することを**エンキュー** (enq), キューからデータを取り出すことを**デキュー** (deq) という。

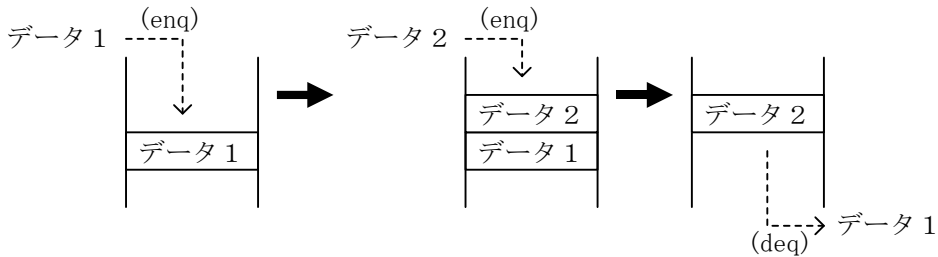
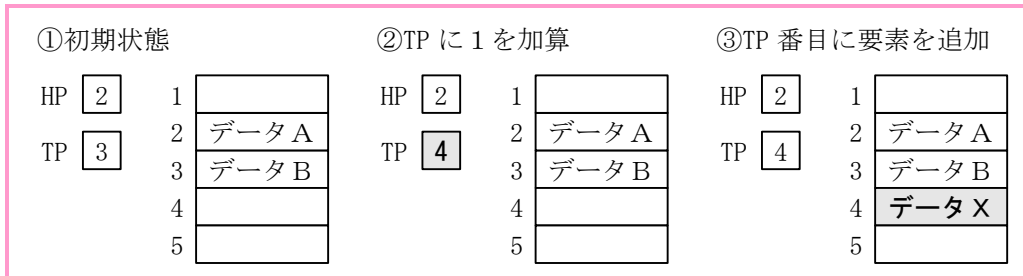


図 7-6 キューのイメージ

キューを実現するには、先頭要素を表すポインタ (HP : Head Pointer) と最後尾要素を表すポインタ (TP : Tail Pointer) を使用する。データを格納する場合は、ポインタ TP の値を更新 (一般的に+1) した位置に記録する。逆に、データを取り出す場合は、ポインタ HP の位置から取り出した後、ポインタ HP の値を更新 (一般的には+1) しておく。キューにデータを格納 (エンキュー) する処理を表現すると、次のようになる。



なお、キューを配列で表す場合、二つのポインタ (HP, TP) はどちらも 1 ずつ加算されていくため、そのままでは配列要素が無限に必要となってしまう。そのため、実際の処理では、ポインタの値が配列要素数より大きくなった場合は 1 に戻すという考え方が利用される。この考え方により配列要素を循環して利用することになり、有限個の要素数の配列でキューを実現することができる。

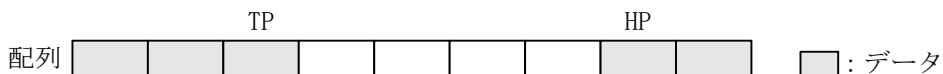
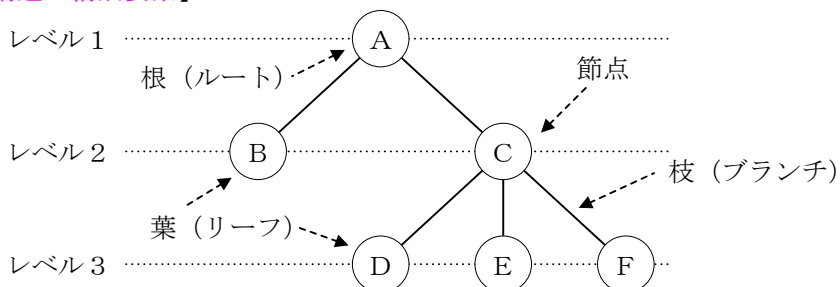


図 7-7 配列の循環利用のイメージ

1-4 木構造（ツリー構造）

木構造（ツリー構造）は、親に対して複数の子が存在する1対多の階層構造を表すためのデータ構造である。

【木構造の構成要素】



- ・ **ノード** : 各要素（データ）。**節点**と訳されるが、根でも葉でもないノードだけを節点と表す場合もある。
- ・ **根（ルート）** : 最上位のノード。木構造に一つだけ存在する。
- ・ **葉（リーフ）** : 最下位のノード
- ・ **枝（ブランチ）** : 各ノード（ルート、リーフを含む）を関係付ける線
- ・ **レベル** : 木構造の階層の深さ

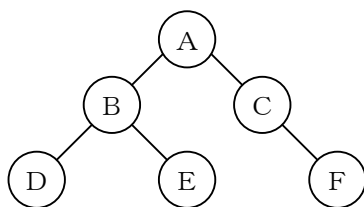
木構造を配列で実現するには、親ノード又は子ノードとの関連をポインタ（添字）で表現する。例えば、上の図の木構造を、親ノードへのポインタ（親 PT）を用いて配列で表現すると、次のようになる。なお、根の親 PT は、0（つながらない）としている。

	1	2	3	4	5	6
ノード値	A	B	C	D	E	F
親 PT	0	1	1	3	3	3

一方、子ノードとの関連で表す場合は、子ノードの数が決まっていなくて扱いにくい。そこで、子ノードの数を2個以下に限定した**2分木**が使用される。なお、子ノードの数が2個より多いn個の木構造を、**多分木**又は**n進木**と呼んで区別することもある。

2分木を配列で実現するには、左の子ノードにつながる左ポインタ（左 PT）と、右の子ノードにつながる右ポインタ（右 PT）を用いて表現する。

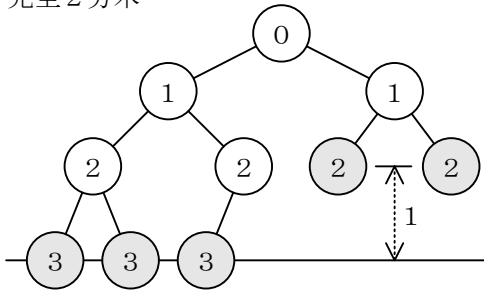
【2分木の配列表現】



	左 PT	ノード値	右 PT
1	2	A	3
2	4	B	5
3	0	C	6
4	0	D	0
5	0	E	0
6	0	F	0

2分木のうち、最終レベルの葉が左詰めになっていて、根からすべての葉までのレベルが等しい、又は高々1しか変わらない木を、特に**完全2分木**という。

・完全2分木



・完全2分木でない木

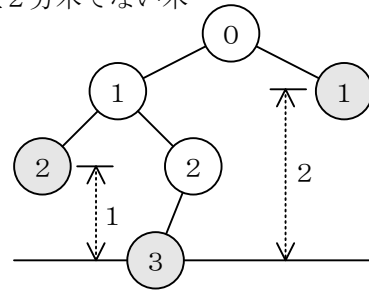


図 7-8 完全2分木と完全2分木でない木の例

また、木構造からノード値（データ）を取り出していく方法を、**木の巡回法**という。木の巡回法には、次のような種類がある。

・**幅優先探索**

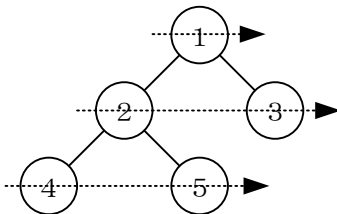
根から順に、同一レベルの左から右にノード値を取り出していく。

・**深さ優先探索**

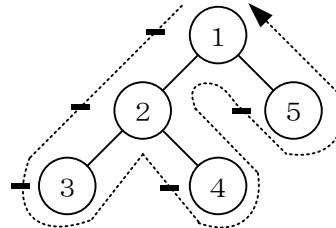
根から順に、左部分木から右部分木まで外周をたどりながら、ノード値を順に取り出していく。

- ・**先行順**：節点→左部分木→右部分木の順に取り出す（行きがけのなぞり）。
- ・**中間順**：左部分木→節点→右部分木の順に取り出す（通りがけのなぞり）。
- ・**後行順**：左部分木→右部分木→節点の順に取り出す（帰りがけのなぞり）。

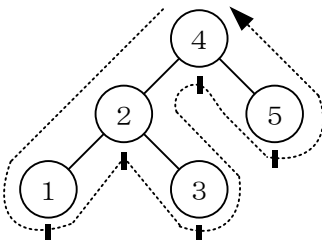
①幅優先探索



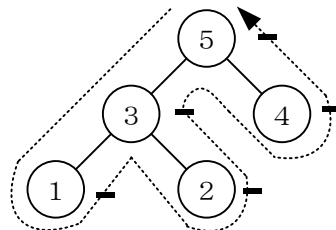
②深さ優先探索（先行順）



③深さ優先探索（中間順）



④深さ優先探索（後行順）



1-4-1 2分木の活用

(1) 2分木による式の表記

計算式は次の法則によって、2分木で表現することができる。計算式を2分木で表現したとき、深さ優先探索（先行順）でノード値を取り出すと**ポーランド記法**、深さ優先探索（後行順）でノード値を取り出すと**逆ポーランド記法**の式を得ることができる。

法則1：演算子を節点に、演算子の両側（変数や式）を左の子と右の子にする。

法則2：複数の演算子がある場合は、後に行う演算の演算子を上位にする。

法則3：()は演算順序を変えるためだけに使用し、ノードには記述しない。

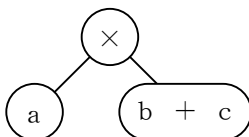
例．計算式“ $a \times (b + c)$ ”を逆ポーランド記法で表現する。

① 計算式を2分木で表現する。

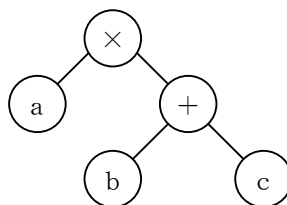
1) 後に行う演算の演算子“ $\times$ ”を節点にした2分木を作る。

2) 右の子“ $b + c$ ”を、演算子“ $+$ ”を節点にした2分木にする。

1)



2)

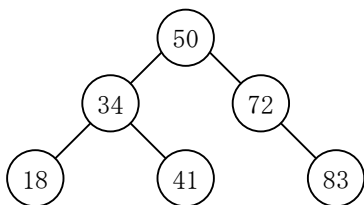


② 深さ優先探索（後行順）でノード値を取り出す。

逆ポーランド記法： $a \ b \ c \ + \ \times$

(2) 2分探索木

2分探索木は、すべてのノードにおいて“左部分木 < 節点 < 右部分木”の関係が成立する2分木のことである。2分探索木は、その名が示すとおり、データを探索するのに適した木構造である（このような木構造のことを**探索木**という）。



左部分木	節点	右部分木
18, 34, 41	50	72, 83
18	34	41
—	72	83

※すべての節点において、
左部分木 < 節点 < 右部分木

図 7-9 2分探索木の例

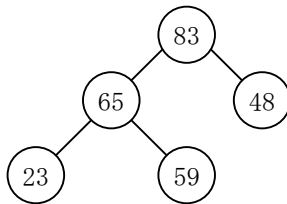
【2分探索木のデータ探索手順】

- 1) 根（ルート）を最初の探索ノードとする。
- 2) 探索ノードがなくなるか、目的値が見つかるまで次の処理を繰り返す。
 - ・探索ノードのノード値と目的値を比較する。
 - “ノード値 > 目的値”：左の子ノードを次の探索ノードにする。
 - “ノード値 < 目的値”：右の子ノードを次の探索ノードにする。

(3) ヒープ

ヒープは、親ノードと子ノードの間に一定の大小関係が成立する完全2分木である。親子の大小関係だけが一定であり、兄弟間では一定の大小関係は成立しない（2分探索木やヒープのように、ノードに一定の順序関係がある木構造を**順序木**という）。

・ヒープ



・ヒープでない木

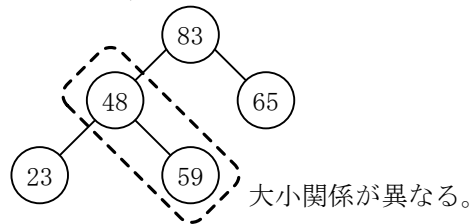


図 7-10 ヒープとヒープでない木の例

ヒープは、完全2分木の構成を保つために**再編成**が行われる。また、ヒープの根には全ノードの最大値（最小値）が記録される。この性質を利用して、根のデータを順次取り出すことで、データを整列することができる（この整列方法を**ヒープソート**という）。

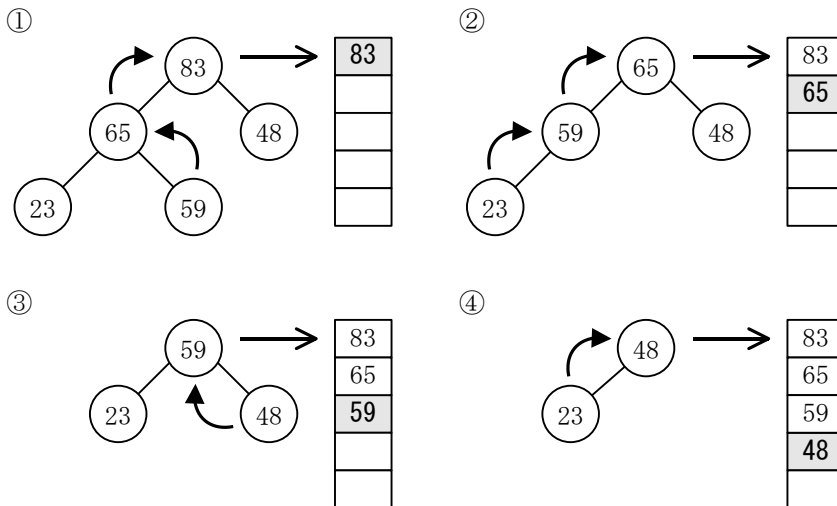


図 7-11 ヒープソートのイメージ

1-4-2 バランス木

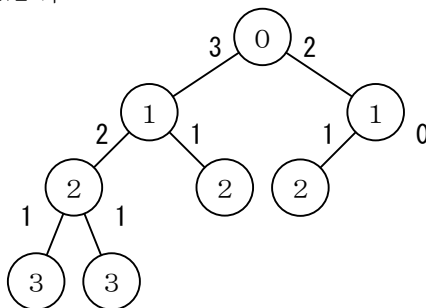
バランス木は、データの追加や削除によって、ある葉のレベルだけが深くなってノードへのアクセス効率が低下することを防止するために、**再編成**を行う木構造である（ヒープもバランス木の一つである）。

代表的なバランス木として、次の二つの木構造がある。

(1) AVL 木

AVL 木は、各ノードにおいて左右の葉までの深さの差が1以内の2分木である。

・ AVL 木



・ AVL 木でない木

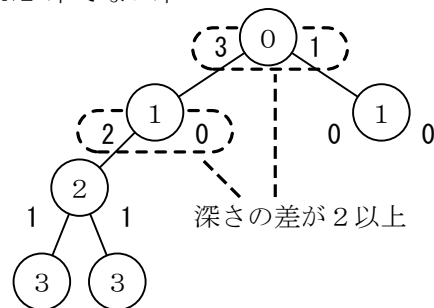


図 7-12 AVL 木と AVL 木でない木の例

(2) B 木

B 木は、2分木の考え方を多分木に発展させたバランス木である。B木は、次のような特徴をもつデータ構造である。

- ・すべての葉（リーフ）が同一レベルにある。
- ・根（ルート）以外の各ノードは、 n 個以上 $2n$ 個以下の要素をもつ。
- ・各ノード（根を含む）は、“要素数+1” 個の子ノードをもつ。
- ・ノードに含まれる各要素は、整列されている。

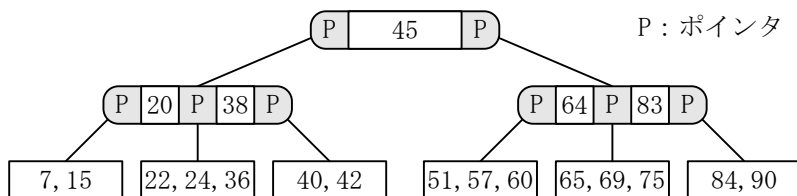


図 7-13 B木のイメージ

2 基本アルゴリズム

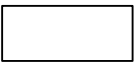
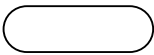
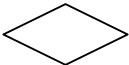
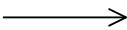
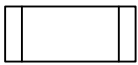
アルゴリズムとは、問題を解決する手順やコンピュータで処理を行うときの手順のことである。アルゴリズムはできるだけ処理効率の良い、わかりやすいものを作成するべきである。この章では、アルゴリズムを表記するフローチャートの書き方と、代表的なアルゴリズムについて学習する。

2-1 フローチャート

フローチャート（**流れ図**）は、一般的に使用されているアルゴリズムの表記法である。処理手順を記号の組合せで表現するので、視覚的にわかりやすいという利点がある。

2-1-1 代表的な記号

フローチャートに使用する代表的な記号は、次のとおりである。

記号	記号名	意味
	処理	さまざまな処理（代入、計算など）を表す。
	端子	フローチャートの開始／終了を表す。開始の端子記号には名称を付ける場合もある。
	判断	条件により処理を分岐させる。
	線	各記号を結び、実行順序を表す。
	ループ端	繰返し処理（ループ）の開始／終了を表す。 “変数名：初期値，増分，終値”の形式で記述することもできる。
	定義済み処理	定義済み処理を呼び出す（同名の端子記号から始まるフローチャートを実行する）。
	データ	データの入力，出力を表す（処理記号で代用されることもある）。

【構造化チャート】

アルゴリズムを表現する**構造化チャート**には、フローチャート以外に PAD や、NS チャートなどがある。

2-1-2 三つの基本構造

三つの基本構造は、わかりやすく処理効率の良いアルゴリズムを作成するための**構造化定理**（一つの入口と一つの出口をもつプログラムであれば、基本的な三つの構造単位の組合せで表現することができる）で利用される構造単位である。構造化定理の考え方によって処理手順を考え、プログラムを作成することを**構造化プログラミング**という。

(1) 順次

“**順次**（SEQUENCE 型）”とは、処理を上から下へ順に実行する構造である。もっとも基本的な構造であり、他の構造単位も“順次”で組み合わせていくことになる。

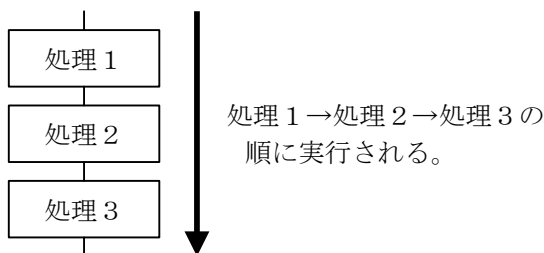


図 7-14 順次

(2) 選択（判定）

“**選択（判定）**”とは、条件により処理を分岐させる構造である。条件によって処理を二つに分岐させる“**双岐選択**（IF THEN ELSE 型）”，処理を複数（通常は三つ以上）に分岐させる“**多岐選択**（CASE 型）”がある（一般的には，“選択”とは“双岐選択”を意味することが多い）。

“双岐選択”は、条件の真偽によって、どちらか一方の処理を実行する。一方，“多岐選択”は、複数の条件の中で成立した条件に該当する処理を実行する。ただし、どちらの場合でも、処理が並列的に同時に実行されることはない。

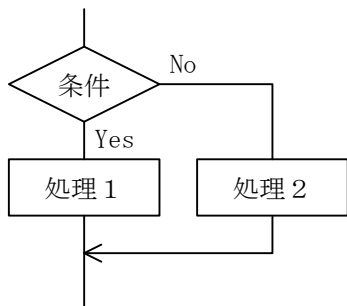


図 7-15 双岐選択

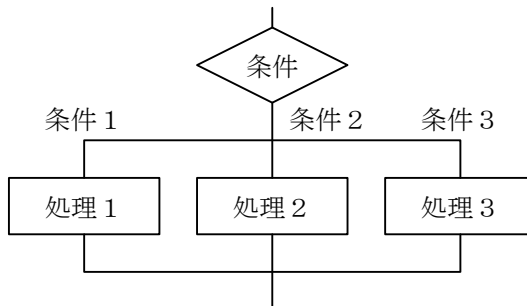


図 7-16 多岐選択

(3) 繰返し（反復）

“**繰返し（反復）**”は、条件が成立している間（又は条件が成立するまでの間）処理を繰り返す構造である。“繰返し”で使われる条件のうち、成立している間、繰返し処理を続ける条件を**継続条件**、成立した時点で繰返し処理を終了する条件を**終了条件**という。

“繰返し”には、繰返し処理の前に条件判定する“**前判定繰返し**（DO WHILE 型）”と、繰返し処理の後に条件判定する“**後判定繰返し**（REPEAT UNTIL 型）”がある。“前判定繰返し”は、繰返し処理を一度も実行しないことがあるが、“後判定繰返し”は繰返し処理を少なくとも1回は実行する。

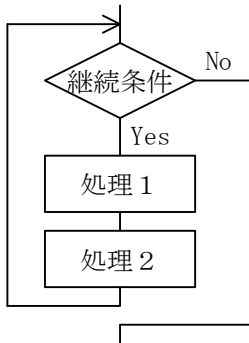


図 7-17 前判定繰返し

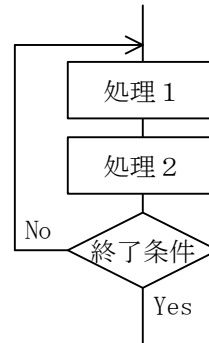


図 7-18 後判定繰返し

なお、ループ端記号では“繰返し”の条件として、原則、終了条件を使用する。また、終了条件以外に、変数の値がどのように変化するかを“変数名：初期値，増分，終値”の形式で記述することもできる。

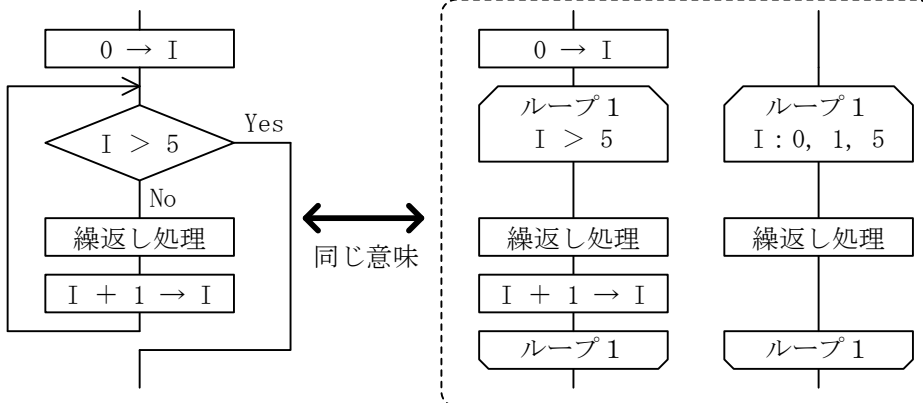


図 7-19 ループ端記号の繰返し

三つの基本構造は、どのように組み合わせて利用してもよい（“選択”の処理に“繰返し”を，“繰返し”の中に“選択”や“繰返し”を入れてよい）。ただし，“選択”の処理に“選択”がある**入れ子構造**（ネスト）は、可能ならば避けたほうがわかりやすい。

2-2 データ探索処理

記録されているデータの中から、目的となるデータを探し出す処理を**探索**（サーチ）という。代表的な探索アルゴリズムとしては、**線形探索法**と**2分探索法**がある。

2-2-1 線形探索法

線形探索法（順次探索法又は逐次探索法）は、先頭要素から順に目的値を探索していくアルゴリズムである。

データが記録された要素数 N 個の配列 T の中から、入力された目的値 X を探索し、記録されている位置（添字）を出力する線形探索法のアルゴリズムについて考える。目的値 X と一致する要素が存在しない場合は、添字の代わりに (-1) を出力する。

	1	2	3	...	N	
T	35	67	21	...	54	X ...処理中で入力

線形探索法では、 $T(1)$ 、 $T(2)$ 、 $T(3)$ 、 $\dots$ 、 $T(N)$ と先頭要素から順に、目的値 X と一致するかを比較する。そのため、添字 I を1、2、3、 $\dots$ 、 N としながら、“ $T(I) = X$ ”となる要素を探しているのが図7-20である（条件“ $I > N$ ”は $T(1) \sim T(N)$ に X と一致する要素が存在しないときに成立する）。ただし、図7-20は基本構造の組合せになっていない（繰返しの出口が2か所ある）ので、これを書き直したものが図7-21である。

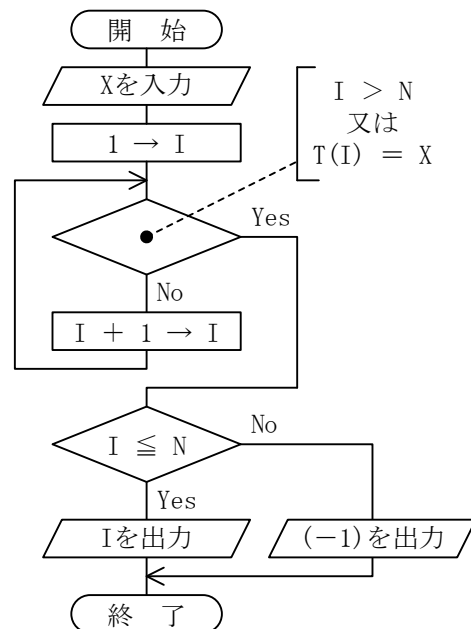
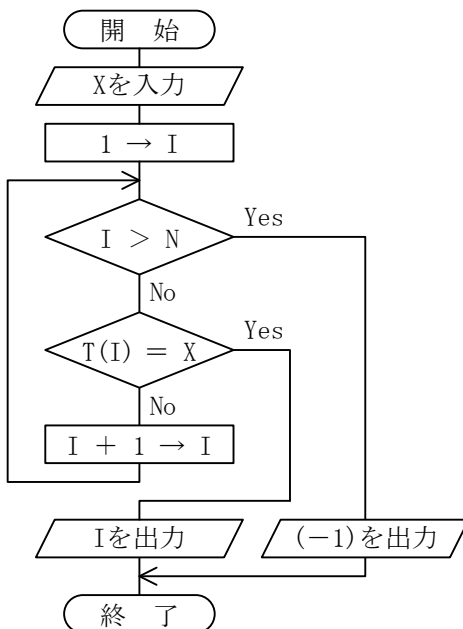


図 7-20 及び図 7-21 では、1 回の繰返しごとに“ $I > N$ ”と“ $T(I) = X$ ”の二つの条件を比較している。この条件比較（判断処理）を減らして、処理効率を向上させる**番兵法**という考え方がある。番兵法は、探索の前に全要素の末尾に目的値を“番兵”として格納する方法である。“番兵”を格納すると、目的値は必ず見つかるようになるので、繰返しの終了条件を“目的値が見つかる（ $T(I) = X$ ）”だけにすることができる。

次の配列 T を用いた番兵法のアルゴリズムは、図 7-22 のようになる。

	1	2	3	...	N	N+1	
T	35	67	21	...	54		...T(N+1) : 番兵を格納する要素

また、線形探索法はリストでも利用できる。リストを使用する場合は、先頭要素からポインタを利用して要素をたどりながら順番に調べていくようになる。

次の配列を用いたリストに記録されたデータの中から、入力された目的値 X を探索し、記録されている位置（添字）を出力するアルゴリズムは図 7-23 のようになる。

TOP		D	PT
	1		
	2		
	3		
	⋮	⋮	⋮
	N		

- ・ TOP : 先頭要素へのポインタ（添字）
- ・ D : 探索対象となるデータ
- ・ PT : 次要素へのポインタ（添字）
（次要素がない場合は 0）

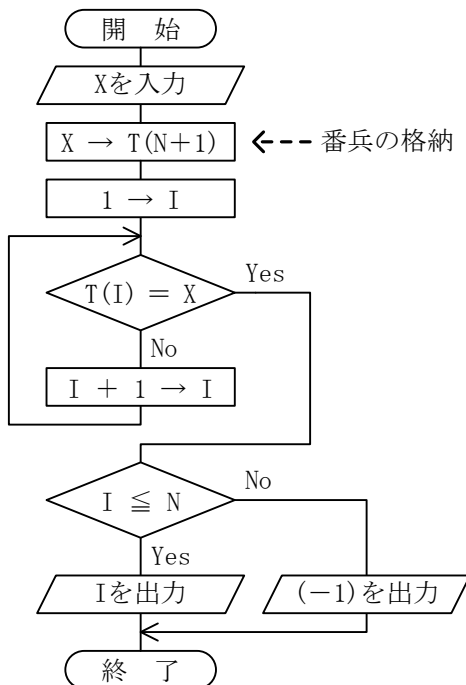


図 7-22 線形探索法（番兵法）

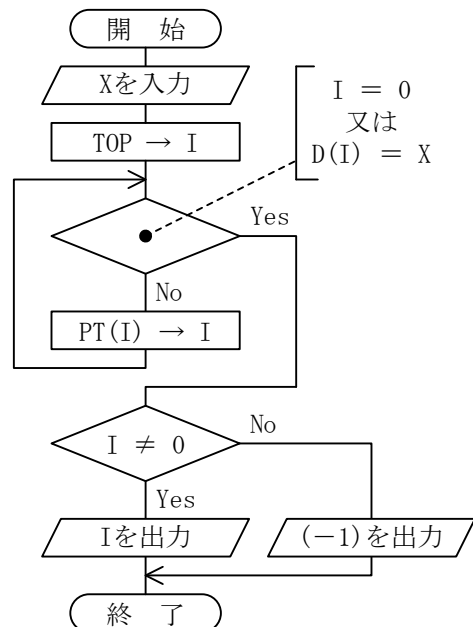


図 7-23 線形探索法（リスト）

2-2-2 2分探索法

2分探索法（バイナリサーチ法）は、データが昇順（又は降順）に記録されているときだけ使用できる探索アルゴリズムである。探索範囲の中央に位置するデータと目的値を比較し、大小関係により探索範囲を狭めていく方式である。

データが昇順に記録された要素数9個の配列Tの中から、入力された目的値Xを探索し、記録されている位置（添字）を出力する2分探索法のアルゴリズムについて考える。目的値Xと一致する要素が存在しない場合は、添字の代わりに(-1)を出力する。

	1	2	3	4	5	6	7	8	9	
T	12	28	34	45	57	60	71	83	96	X

目的値Xに34が入力された場合の探索手順は、次のようになる。

- 1) 最初の探索範囲として、配列全体を指定する。探索範囲を表す小さいほうの添字をL，大きいほうの添字をHに格納する。

	L								H
T	12	28	34	45	57	60	71	83	96

- 2) 探索範囲の中央（M=5）に位置する要素T(5)と目的値Xを比較する。
→ T(M) > X なので、T(M)の後にXはない（HをMの一つ前に移動）。

	L			H	M				
T	12	28	34	45	57	60	71	83	96

- 3) 探索範囲の中央（M=2）に位置する要素T(2)と目的値Xを比較する。
→ T(M) < X なので、T(M)の前にXはない（LをMの一つ後に移動）。

		M	L		H				
T	12	28	34	45	57	60	71	83	96

- 4) 探索範囲の中央（M=3）に位置する要素T(3)と目的値Xを比較する。
→ T(M) = X なので、Mを出力して探索を終了する。

			L, M	H					
T	12	28	34	45	57	60	71	83	96

※探索範囲の中央にある要素の添字Mは“(L+H)÷2”で求める。このとき、計算結果が小数になる場合は小数点以下を切り捨てる。

※処理を繰り返している途中で探索範囲がなくなった（L>Hになった）場合、目的値Xと一致する要素は存在しないことになる。

この考え方でフローチャートを作成すると、図 7-24 のようになる。このフローチャートでは、探索中の状態を変数 FLG の値（0：見つかっていない，1：見つかった）で確認している（このように、状態を表す変数のことを**フラグ**という）。なお、破線で囲んだ入れ子構造の部分は、多岐選択で書き換えることもできる。

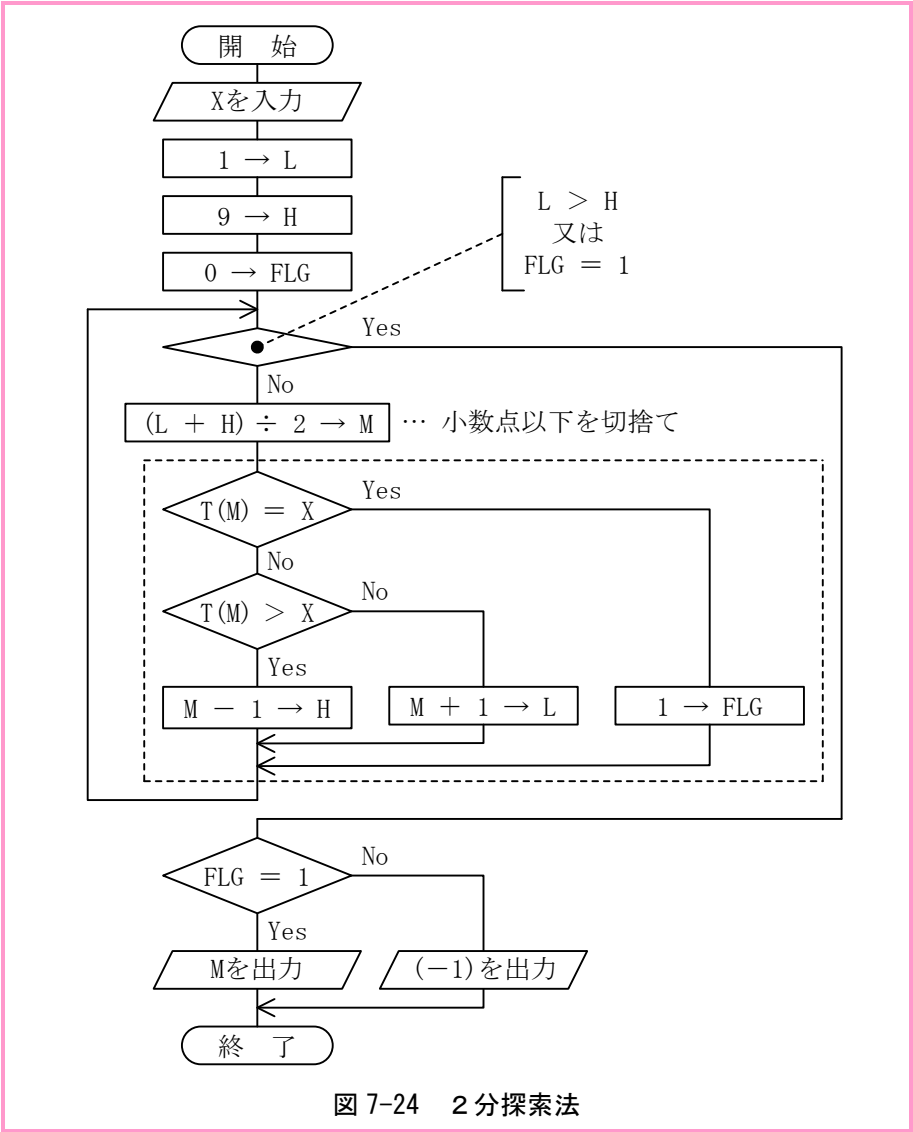


図 7-24 2分探索法

また、**2分探索木**を利用する探索も2分探索法の一つといえる。2分探索木は、すべてのノードにおいて“左部分木 < 節点 < 右部分木”の関係が成立するため、節点のノード値と目的値の大小関係により、次に探索する節点を決定する。

【2分探索木を利用する探索（目的値に41が入力された場合の探索手順）】

	左 PT	ノード値	右 PT
1	2	50	3
2	4	34	5
3	0	72	6
4	0	18	0
5	0	41	0
6	0	83	0

- 1) 根の添字 (1) を I に格納
- 2) ノード値(I) > 目的値
⇒ 左 PT(I) → I
- 3) ノード値(I) < 目的値
⇒ 右 PT(I) → I
- 4) ノード値(I) = 目的値
⇒ I を出力して探索終了

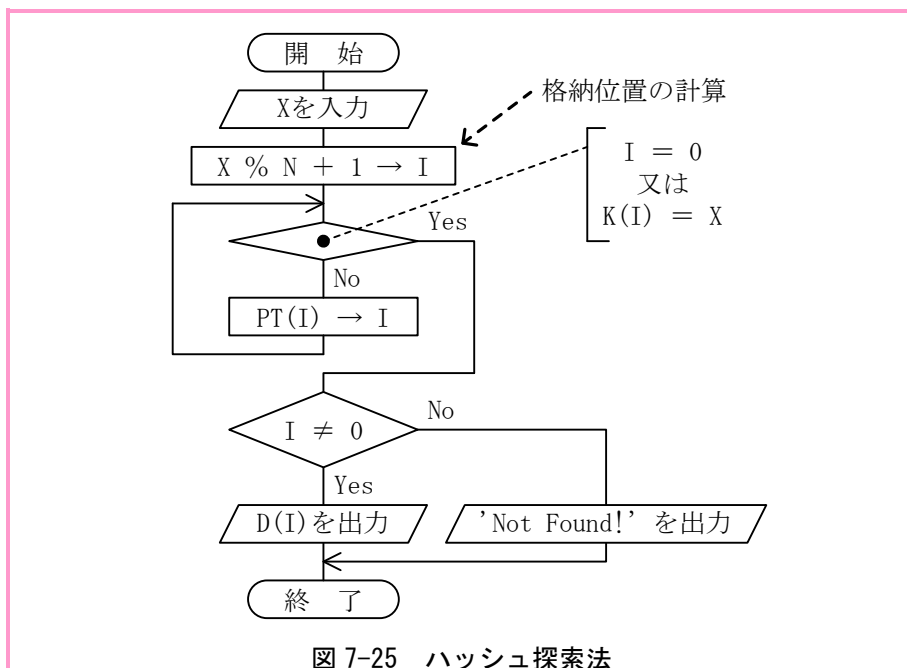
2-2-3 ハッシュ探索法

ハッシュ探索法（ハッシュ表探索法）とは、格納するデータから格納場所（添字）を求める**ハッシュ表**を利用した探索アルゴリズムである。目的値から、データの格納場所を計算（**ハッシュ関数**）によって求められるため、基本的には1回の探索で目的値を見つけることができる。ポイントとなるのは、**シノニム**が発生した場合の処理である。

次のハッシュ表（配列）に記録されたデータの中から、入力された目的値Xと一致するキーを探索し、そのキーに対応するデータを出力するアルゴリズムについて考える。目的値Xと一致するキーが存在しない場合は、メッセージ“Not Found!”を出力する。

	K	D	PT	
1				• K : 格納位置（添字）を決定するキー
2				格納位置（添字）= $K \% N + 1$
3				• D : キーに対応するデータ
⋮	⋮	⋮	⋮	• PT : シノニムレコードへのポインタ（添字）
N				（シノニムレコードがない場合は0）
⋮	⋮	⋮	⋮	} シノニム領域（シノニムレコードを格納する）
M				

このハッシュ表では、シノニム領域を確保してシノニムレコードを格納し、ポインタでつないでいく**チェーン法**を利用している。したがって、目的値から求めた格納場所のキーが目的値と一致しなかった場合は、ポインタによりシノニムレコードをたどっていくことになる。このフローチャートは、図 7-25 のようになる。



2-2-4 計算量

計算量とは、アルゴリズムを評価する指標の一つである。処理するデータ量に応じた処理時間の増加量を表す**時間計算量**や、メモリの使用領域の増加量を表す**空間計算量**（領域計算量）などがある。計算量の表記には、アルゴリズムの計算量の上限を O （オーダ）で表す**オーダ記法**などがある。オーダ記法では、全体に対して最も影響を与える部分の計算量が、データ数 N に比例するなら $O(N)$ 、 N の2乗に比例するなら $O(N^2)$ と表記する。

三つの探索アルゴリズムについて、計算量（時間計算量）を比較してみる。探索アルゴリズムでは、目的値が見つかるまで比較処理（判定処理）を繰り返すので、比較回数（探索回数）がアルゴリズム又は実装したプログラムの処理時間を決定する。

各探索アルゴリズムの比較回数を、探索対象となるデータ数を N 件としてまとめると、次のようになる。

	最小比較回数	最大比較回数	平均比較回数
線形探索法	1 回	N 回	$N \div 2$ 回
2 分探索法	1 回	$\lceil \log_2 N \rceil + 1$ 回	$\lceil \log_2 N \rceil$ 回
ハッシュ探索法	1 回	$1 + \alpha$ 回	1 回※

・ $\lceil \]$ は、 $\lceil \]$ 内の数値を超えない最大の整数を表す（ガウス記号）。

・ $\log_2 N$ は、 $2^x = N$ となる x を表す（例えば、 $\log_2 16 = 4$ となる）。

※格納場所（ハッシュ値）が**一様分布**（事象の発生確率がほぼ等しい分布）で近似されて、シノニムが発生する確率が無視できるほど小さい場合

例. データ数が 100 件から 200 件に増加したとき、線形探索法と 2 分探索法では平均比較回数がそれぞれ何回増えるか。

- ① 線形探索法の平均比較回数の増加回数
 = データ数 200 件の平均比較回数 - データ数 100 件の平均比較回数
 = $(200 \div 2)$ 回 - $(100 \div 2)$ 回 = 100 回 - 50 回 = 50 回
- ② 2 分探索法の平均比較回数の増加回数
 = データ数 200 件の平均比較回数 - データ数 100 件の平均比較回数
 = $\lceil \log_2 200 \rceil$ 回 - $\lceil \log_2 100 \rceil$ 回 = 7 回 - 6 回 = 1 回

したがって、線形探索法の計算量は $O(N)$ 、2 分探索法の計算量は $O(\log_2 N)$ 、ハッシュ探索法の計算量は $O(1)$ となる。

なお、オーダ記法の計算量は指標（目安）であり、実際の処理時間とは必ずしも一致しない。例えば、目的値が見つからない場合は最大比較回数の比較が行われるので、探索成功と探索失敗の確率により平均探索時間は変動する。また、線形探索で昇順（又は降順）に記録されたデータを用いて、目的値が存在しないことがわかった時点で終了するように改良すると、探索失敗時の比較回数を少なくできるが、オーダは変わらない。

2-3 データ整列処理

記録されているデータを、指定した項目の昇順又は降順に並べ替える処理を**整列**（ソート）という。整列は、大きく二つに分類される。

・内部整列

整列対象のデータが、主記憶装置に記録されている場合の整列である。

基本選択法、基本交換法、基本挿入法、クイックソート、

ヒープソート、シェーカーソート、シェルソート

・外部整列

整列対象のデータが、補助記憶装置に記録されている場合の整列である。

マージソート

2-3-1 基本選択法

基本選択法（選択ソート）は、先頭要素から順番に一つずつ要素を決定していく整列アルゴリズムである。

データが記録された要素数 N 個の配列 S を、昇順に整列する基本選択法のアルゴリズムについて考える。問題を簡単にするために、次の配列 S （ $N=5$ ）を使用する。

	1	2	3	4	5		
S	28	84	73	16	51	N	5 …要素数

配列 S を基本選択法で昇順に整列する手順は、次のようになる。

- 1) 全要素の中から最小値を選択し、先頭要素 $S(1)$ とする。

S

16	84	73	28	51

- 2) 残りの要素の中から最小値を選択し、2番目の要素 $S(2)$ とする。

S

16	28	73	84	51

- 3) 残りの要素の中から最小値を選択し、3番目の要素 $S(3)$ とする。

S

16	28	51	84	73

- 4) 残りの要素の中から最小値を選択し、4番目の要素 $S(4)$ とする。

→ 5番目の要素 $S(5)$ は、自動的に決定する。（整列終了）

S

16	28	51	73	84

この考え方を基に作成したフローチャートが、図 7-26 と図 7-27 である。図 7-26 では比較をするたびに要素を交換しているが、図 7-27 では比較が終わって交換する要素が決定してから 1 回だけ交換している（上の手順は図 7-27 に該当する）。どちらの方法でも $S(I)$ に格納する要素を決定するために、 $S(I+1) \sim S(N)$ と比較を繰り返している。

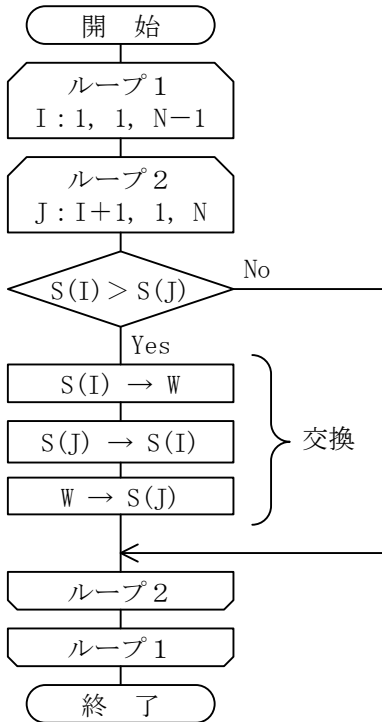


図 7-26 基本選択法①

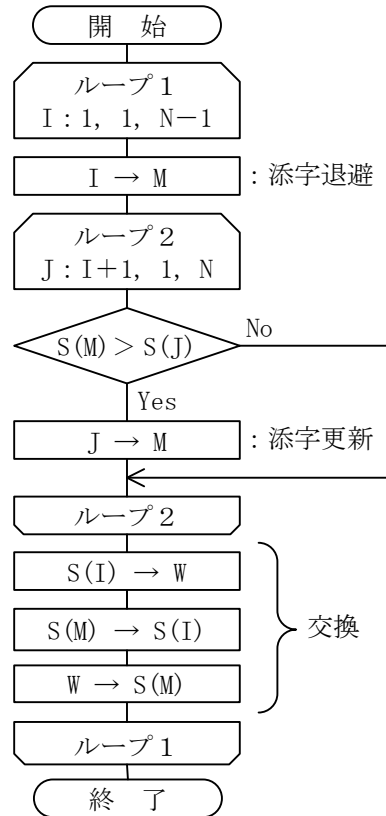
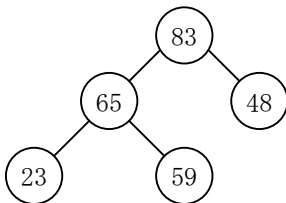


図 7-27 基本選択法②

【ヒープソート】

ヒープソートは、基本選択法を改良した整列アルゴリズムである。整列対象となるデータをヒープで表現し、根のデータを順次取り出していく（ヒープ及びヒープソートの詳細については、P. 361 を参照）。

ヒープの配列表現では、根を配列の 1 番目に置き、 I 番目の要素の左の子ノードを $(2 \times I)$ 番目、右の子ノードを $(2 \times I + 1)$ 番目に置く。この関係を利用して、親ノードと子ノードの要素を比較／交換しながらヒープは再編成される。



	1	2	3	4	5	6
H	83	65	48	23	59	
	↓ 根 (83) を取り出して再編成					
H	65	59	48	23		
	↓ 根 (65) を取り出して再編成					
H	59	23	48			

2-3-2 基本交換法

基本交換法（バブルソート）は、隣り合う要素を比較し、大小関係が逆（逆順）ならば交換して正しい関係（正順）にする操作を、先頭要素から順に行っていく整列アルゴリズムである。バブルソートという名称は、最大値又は最小値が徐々に後方に移動する動きが水中に発生した泡の動きに似ていることから付けられた。

基本選択法で使った配列 S（N=5）を用いて、データが記録された要素数 N 個の配列 S を、昇順に整列する基本交換法のアルゴリズムについて考える。

	1	2	3	4	5		
S	28	84	73	16	51	N	5 …要素数

配列 S を基本交換法で昇順に整列する手順は、次のようになる。なお、先頭から末尾の要素までの比較／交換を繰り返すことを、1 回のパスということもある。

- 1) 1 回目の比較／交換を、先頭要素から順に行う。
→ 5 番目の要素 S(5) が決定する。
S

28	73	16	51	84
- 2) 2 回目の比較／交換を、先頭要素から順に行う。
→ 4 番目の要素 S(4) が決定する。
S

28	16	51	73	84
- 3) 3 回目の比較／交換を、先頭要素から順に行う。
→ 3 番目の要素 S(3) が決定する。
S

16	28	51	73	84
- 4) 4 回目の比較／交換を、先頭要素から順に行う。
→ 2 番目の要素 S(2) が決定する。
→ 1 番目の要素 S(1) は、自動的に決定する。（整列終了）
S

16	28	51	73	84

この説明では、末尾の要素を一つずつ決定しながら整列対象から外していき、全要素の位置が決定するまでパスを繰り返している。図 7-28 では、要素数 N を 1 回のパスごとに 1 ずつ減らしていくことで、このアルゴリズムを実現している。

一方、1 回のパスの中で交換が一度も起こらなければ、すべての要素は正順に並んでいるので、整列を終了するという考え方もある。図 7-29 は、変数 FLG の値（0：初期状態，1：交換が発生した）を利用し、交換が発生しなければ処理を終了する方法である。

なお、効率を重視する場合、二つの方法を組み合わせて使用することもできる。どのようなフローチャートになるかは、考えてみて欲しい。

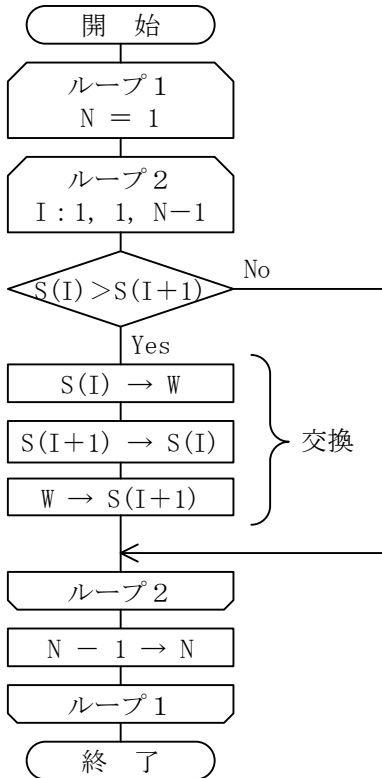


図 7-28 基本交換法①

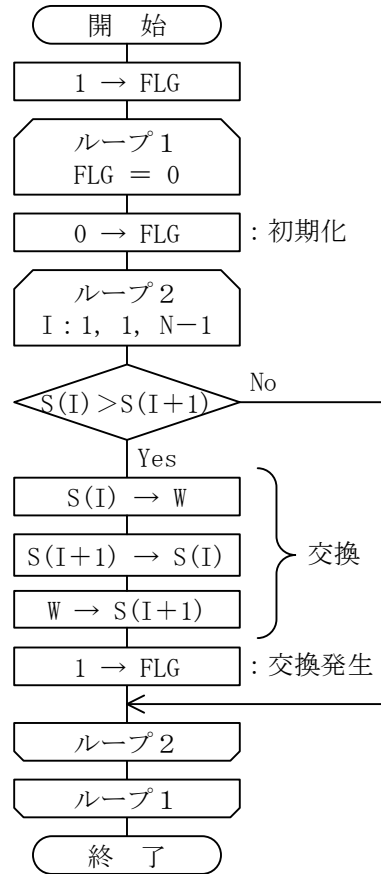
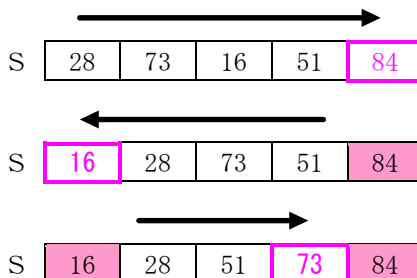


図 7-29 基本交換法②

〔シェーカーソート〕

シェーカーソートは、基本交換法を改良した整列アルゴリズムである。基本交換法は、要素番号の小さいほうから順に隣同士を比較／交換していくが、シェーカーソートは要素番号の小さいほうから順に隣同士を比較／交換した後、今度は逆に要素番号の大きいほうから順に比較／交換していく。小さいほうから大きいほうへ、大きいほうから小さいほうへと交互に要素が動く様子から、シェーカーソートと呼ばれる。ただし、効率はそれほど向上しない。



2-3-3 基本挿入法

基本挿入法（挿入ソート）は、データを整列済みの部分と整列前の要素に分割し、整列済みの部分の順序を壊さないように、データを挿入していく整列アルゴリズムである。

基本選択法で使った配列S（N=5）を用いて、データが記録された要素数N個の配列Sを、昇順に整列する基本挿入法のアルゴリズムについて考える。

	1	2	3	4	5
S	28	84	73	16	51

N	5

 …要素数

配列Sを基本挿入法で昇順に整列する手順は、次のようになる。

- 1) 1番目の要素が整列済みと考える。
S

28

84	73	16	51
- 2) 整列済みの部分（1～1）に、整列前の要素から“84”を挿入する。
S

28	84

73	16	51
- 3) 整列済みの部分（1～2）に、整列前の要素から“73”を挿入する。
S

28	73	84

16	51
- 4) 整列済みの部分（1～3）に、整列前の要素から“16”を挿入する。
S

16	28	73	84

51
- 5) 整列済みの部分（1～4）に、整列前の要素から“51”を挿入する。
→ 整列前の要素がなくなる。（整列終了）
S

16	28	51	73	84

基本挿入法で、昇順に整列済みの部分にデータを挿入するには、整列済み部分の末尾の要素から順に比較して挿入データより小さい値が見つかるか、比較要素がなくなる（先頭要素まで交換する）まで交換を繰り返していけばよい。上記の5)を例に、要素“51”を挿入する処理を表すと、次のようになる。

☆整列済みの部分（1～4）に、整列前の要素から“51”を挿入する。

S

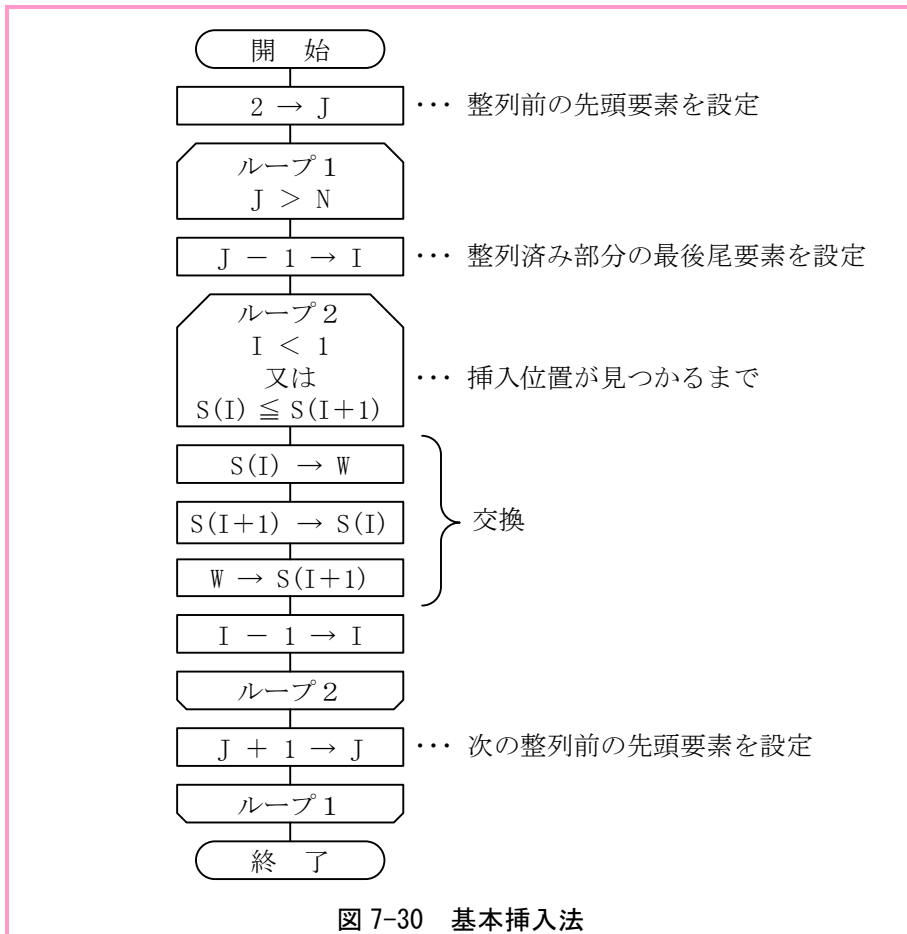
16	28	73	84	51

- 1) “84>51”なので、二つの要素を交換する。
S

16	28	73	51	84
- 2) “73>51”なので、二つの要素を交換する。
S

16	28	51	73	84
- 3) “28<51”なので、挿入位置が決定する。

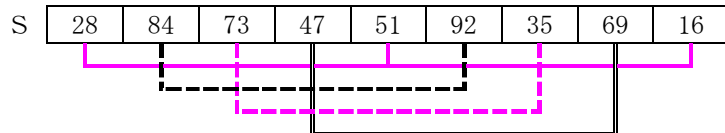
この考え方を基に作成したフローチャートが、図 7-30 である。図 7-30 では、整列前の先頭要素（挿入するデータ）を変数Jで管理している。



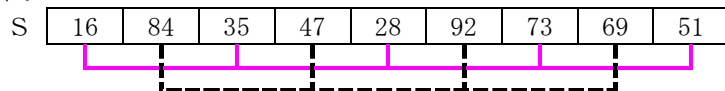
〔シェルソート〕

シェルソートは、基本挿入法を改良した整列アルゴリズムである。整列するデータを一定間隔（ギャップ）ごとに抽出して、そのデータを挿入法で整列する。このギャップの大きさを徐々に小さくしていき、1になるまで繰り返していく。大幅なデータの移動が少なくなる、効率の良いアルゴリズムである。

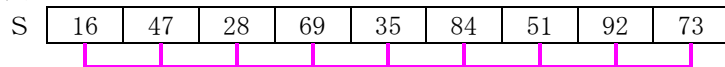
1) ギャップ=4



2) ギャップ=2



3) ギャップ=1



2-3-4 クイックソート

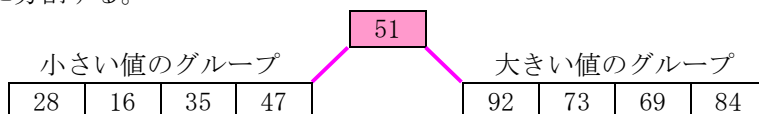
クイックソートは、高速な整列アルゴリズムである。整列したいデータ群から基準値を選択し、その基準値より小さい値のグループと大きい値のグループに分割する。次に、それぞれのグループで基準値を選択して、同様に分割する。この操作を、全グループの要素が一つになるまで、自分自身を呼び出して繰り返す**再帰的アルゴリズム**である。

次の配列 S (N=9) を利用して、クイックソートのアルゴリズムについて考える。

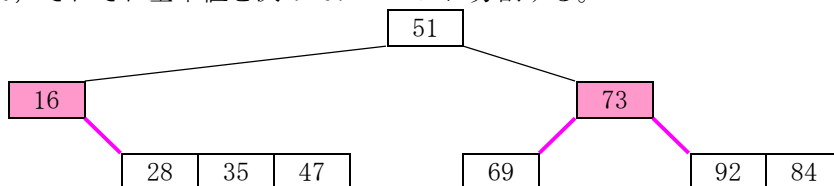
	1	2	3	4	5	6	7	8	9	
S	28	84	73	47	51	92	35	69	16	N 9

配列 S をクイックソートで昇順に整列する手順のイメージは、次のようになる。なお、基準値としては、対象データの中央にある要素を利用する。

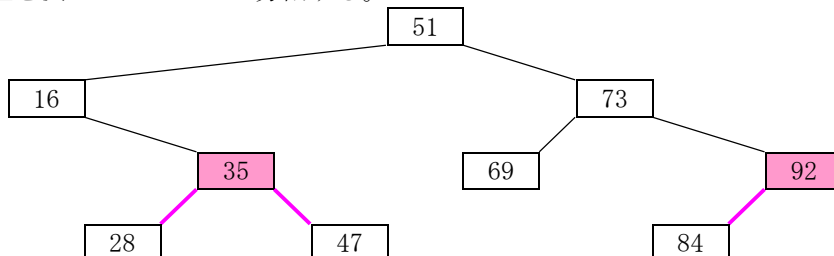
- 1) S(1)～S(9) を、基準値 S(5) より小さい値のグループと大きい値のグループに分割する。



- 2) 小さい値のグループ {S(1)～S(4)} と大きい値のグループ {S(6)～S(9)} で、それぞれ基準値を決めてグループに分割する。



- 3) 要素が一つでないグループ {S(2)～S(4), S(8)～S(9)} で、それぞれ基準値を決めてグループに分割する。



- 4) すべてのグループの要素が一つになったら、整列を終了する。

16	28	35	47	51	69	73	84	92

この考え方を基に作成したフローチャートが、図 7-31 である。図 7-31 では、クイックソートの処理を“QSORT(左端の添字, 右端の添字)”で再帰的に呼び出している。

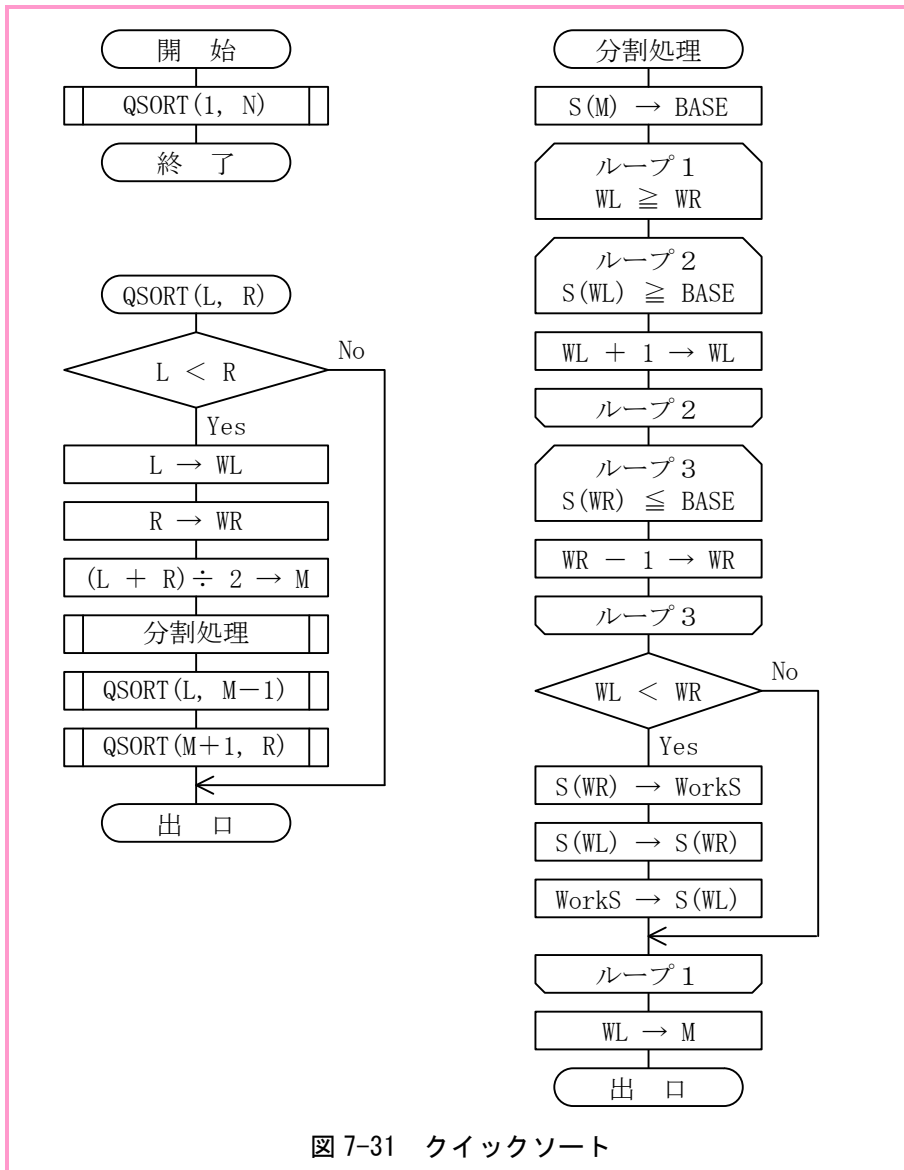
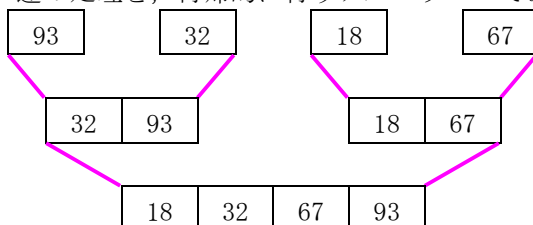


図 7-31 クイックソート

【マージソート】

マージソートは、整列された二つのデータを、順序を乱さずに一つにまとめる**併合**（マージ）を利用する整列アルゴリズムである。データの分割，併合という一連の処理を，再帰的に行うアルゴリズムである。



: 1 回目の併合（マージ）

: 2 回目の併合（マージ）

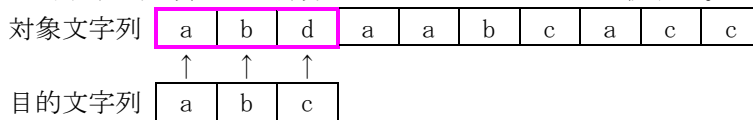
2-4 その他のアルゴリズム

2-4-1 文字列処理

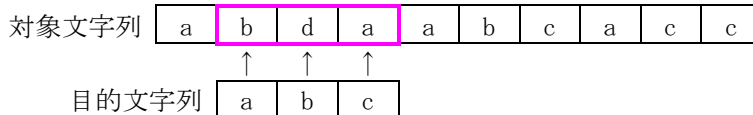
文字列処理とは、配列の要素として文字を利用するアルゴリズムのことである。代表的な文字列処理として、配列に記録された文字列の中から目的文字列を探索する**文字列照合**がある。基本的な考え方は、配列の要素に数値が記録されている場合と変わりはないが、目的値が文字列となるため、複数の要素が同時に一致するかを確認しなければならない。

文字列照合の最も簡単な考え方としては、配列要素中の部分文字列（目的文字列と同じ文字数の文字列）を、先頭から1文字ずつずらしながら照合する方法がある。

- 1) 配列要素中の先頭にある部分文字列と目的文字列を比較する。

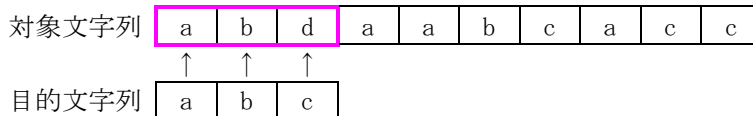


- 2) 部分文字列が一致しない場合、開始位置を1文字ずらして比較する。

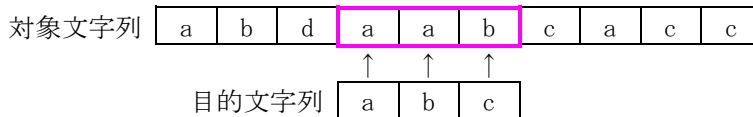


この手順に誤りはないが、効率はあまり良くない。そこで、文字列の効果的な照合方法として、次の**ポイヤ・ムーア法**が考えられた。

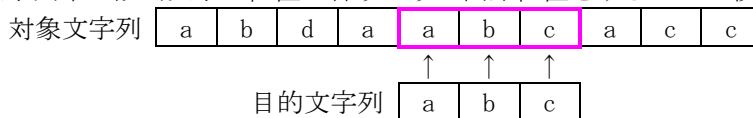
- 1) 配列要素中の先頭にある部分文字列と目的文字列を比較する。



- 2) 部分文字列の末尾の文字が目的文字列中に存在しない場合、開始位置を目的文字列の文字数分ずらして比較する。



- 3) 部分文字列の末尾の文字が目的文字列中に存在する場合、末尾の文字と目的文字列中の該当文字の位置が合うように開始位置をずらして比較する。



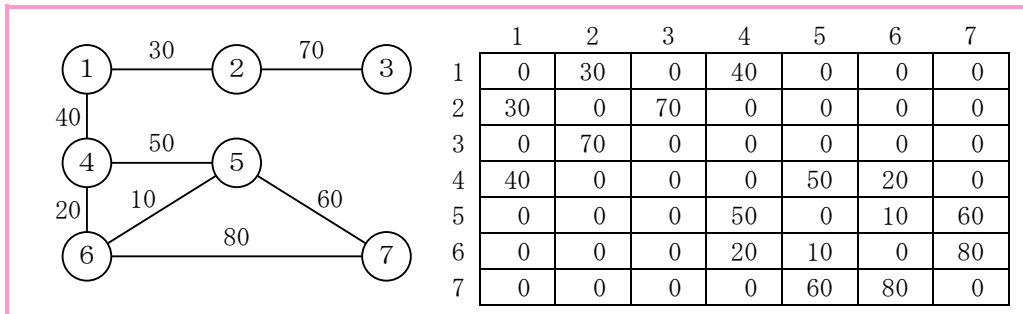
このような文字列照合以外に、文字列処理には**ランレングス法**などに代表される文字列の圧縮／解凍アルゴリズムなどがある。

2-4-2 グラフ処理

グラフ処理とは、グラフを利用するアルゴリズムのことである。**グラフ**とは、節点と枝（辺）からなる図形である（木構造もグラフの一つである）。

グラフには、状態遷移図のように枝に方向がある**有向グラフ**と、木構造やネットワーク図のように枝に方向がない**無向グラフ**がある。

グラフは、行列、配列、リストなどで表現することができる。アルゴリズムでは、節点の隣接（連結）状態を表した**隣接行列**（配列）を用いることが多い。無向グラフ（ネットワーク図）を表した隣接行列の例は、次のようになる。この例では、節点を結ぶ枝のコスト（距離など）を記録して、枝があることを表している（枝がない場合は0とする）。



代表的なグラフ処理として、出発地点（節点）から目的地点（節点）までの経路を見つける**経路探索**がある。経路探索の代表的な方法には、次の二つがある。

- ・ **幅優先探索**

経路を階層的に探していく方法である。それぞれの節点到隣接する1階層下の節点を求める処理を、目的地点が出現するまで繰り返す。

- ・ **深さ優先探索**

一つの経路を、最後（行止まり又は目的地点）まで探していく方法である。解の経路でないと判明した（①→②→③となった）場合は、手前に戻って別の経路を探索する**バックトラック法**（**バックトラッキング**）である。

ただし、この二つの方法では、見つかった経路が最短経路（コスト最小の経路：①→④→⑥→⑤→⑦）であるとは限らない。そこで、**最短経路探索**の方法として、最低コストの節点を一つずつ順番に確定していく**ダイクストラ法**が考えられた。

【**ダイクストラ法の一般的な手順**】

- 1) 出発地点に隣接する節点で最低コストの節点Xを確定する。
- 2) 確定した節点Xに隣接する節点も含めて、その中で最低コストの節点を一つ確定する。
- 3) 2)の処理を、すべての節点の最低コストが確定するまで繰り返し行う。
⇒ 目的地点までの最低コスト（経路）が決定する。

2-4-3 数値処理

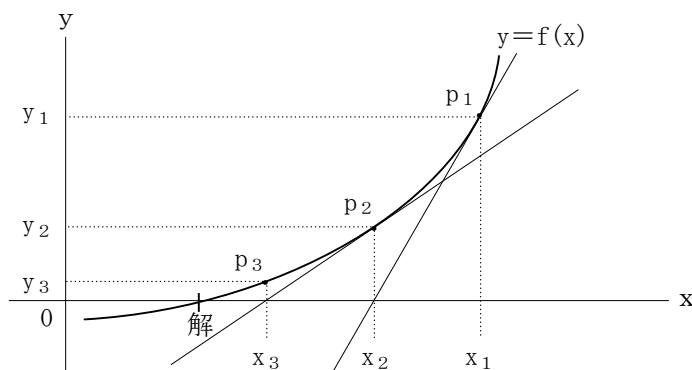
数値処理とは、近似解を求める数値計算などを利用するアルゴリズムのことである。

(1) ニュートン法

ニュートン法は、次数の大きい n 次方程式の近似解がわかっているときに、その近似解を修正していきながら真の解の近似値を求める方法である。

【ニュートン法のアルゴリズム】

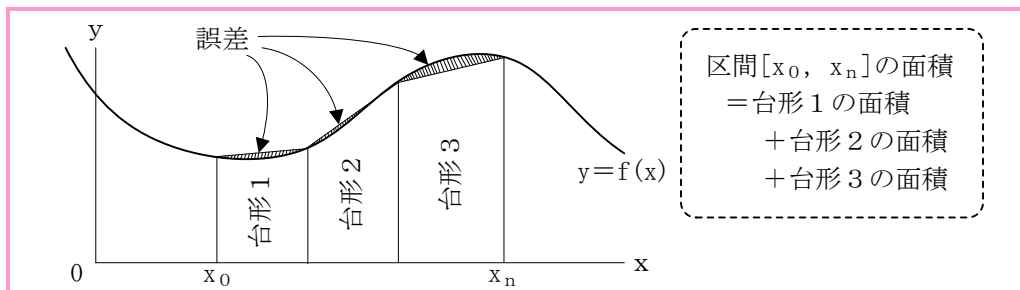
- 1) 点 $p_1(x_1, y_1)$ における $y=f(x)$ の接線と x 軸との交点を、 x_2 とする。
- 2) 点 $p_2(x_2, y_2)$ における $y=f(x)$ の接線と x 軸との交点を、 x_3 とする。
- 3) 以下同様に、 $x_4, x_5 \dots$ と求めていき、 x_n と x_{n+1} の差がある一定値以下に収束したとき、 x_{n+1} を解とする。



ニュートン法は、関数 $f(x)$ の接線を求める必要があるため、関数 $f(x)$ が微分可能でなければ利用できない。また、初期値として曲線上の1点の x 座標を与えればよいが、初期値の与え方によっては解が収束しない（解の近似値が得られない）場合もある。

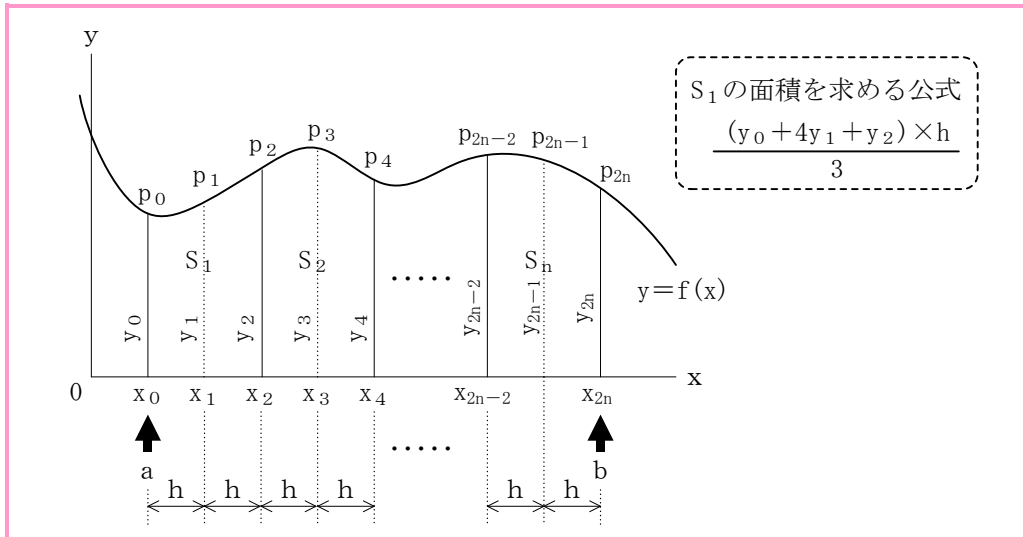
(2) シンプソン法

曲線と x 軸の区間で囲まれた部分の面積の近似値を求める方法の一つとして、**台形公式**がある。台形公式では、求める面積の部分を台形に分割して、その合計を求める。



しかし、台形公式は曲線を直線に近似するため、誤差が大きくなるという欠点がある。誤差を少なくするためには分割する台形の数多くするしかないが、分割する台形の数が多くなれば、それだけアルゴリズムが複雑になり、計算量が増す。

そこで、**シンプソン法**では、分割して面積を求めるときに、 $y=f(x)$ の辺に該当する部分を放物線で近似する。具体的には、次のように分割した S_1 部分の面積を、点 p_0 と p_2 を直線で結ばずに、点 p_0 、 p_1 、 p_2 を通る放物線と考えて公式で求める。



台形公式では求める面積の部分を n 等分するが、シンプソン法では $2n$ （偶数）等分することが特徴的である。

2-4-4 ファイル処理

ファイル処理とは、レコードが記録されたファイルを利用するアルゴリズムである。基本的なファイル処理では、“入力ファイルからレコードを読み込む”、“入力レコードを処理（計算処理、編集処理など）する”、“出力ファイルにレコードを書き出す”という一連の流れを、入力ファイル中のレコードがなくなるまで繰り返し行うことになる。

・ 整列処理／併合処理

ファイルに記録されたレコードの整列や、整列された二つのファイルを整列順を乱さずに一つのファイルにまとめる併合を行う処理である。レコードの内容をいったん配列に格納して内部処理で実現する方法と、プログラム言語の命令（SORT 命令、MERGE 命令）やサービスプログラムを利用する方法がある。

・ コントロールブレイク処理

ファイルのレコードがキー項目で整列されている場合、キー項目の値が変わるまでを一つのグループとして扱う処理である。グループごとの合計を求める処理（グループトータル処理）などに利用される。

2-5 アルゴリズム設計

アルゴリズム設計とは、問題分析を行ってアルゴリズムを作成することである。

問題分析では、入力データ／出力データの種類やデータ構造、各データ間の関連などを明確にし、複雑な処理条件を**決定表（デシジョンテーブル）**などにまとめる。

・決定表（デシジョンテーブル）

条件に応じた行動や処理を表形式でまとめたものである。

条件表題欄	条件記入欄 (Y/N/-)	雨が降っている	Y	N	N
行動表題欄	行動記入欄 (X/-)	降水確率が 30%以上	-	Y	N
		長い傘を持っていく	X	-	-
		折り畳み傘を持っていく	-	X	-
		傘は持っていない	-	-	X

アルゴリズムの作成に当たって、問題が大きくて複雑な場合は、小さな問題に分割してそれぞれを解決することで全体の解を求める**分割統治法**を利用することもある。分割統治法を用いると、アルゴリズムは**再帰**になることが多い（クイックソートなど）。例えば、1～N（ $N > 2$ ）の階乗を求める図 7-32 のアルゴリズムを、分割統治法の考え方（Nに対して何をすればよいかだけ考える）で作成すると、図 7-33 のような再帰になる。

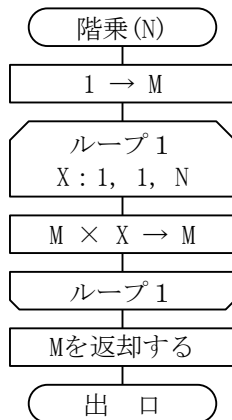


図 7-32 階乗の求め方①

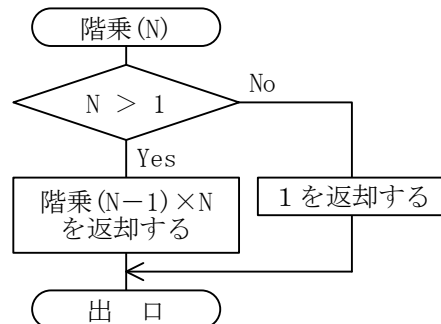


図 7-33 階乗の求め方②

また、上図のアルゴリズムは、次のような**擬似言語**を用いて表現することもできる。

```

○整数型: 階乗(整数型: N)
  整数型: M, X
  M ← 1
  for (X を 1 から N まで 1 ずつ増やす)
    M ← M × X
  endfor
  return M
  
```

```

○整数型: 階乗(整数型: N)
  if (N が 1 より大きい)
    return 階乗(N - 1) × N
  else
    return 1
  endif
  
```

第7部 演習問題

問1

図は単方向リストを表している。“東京”がリストの先頭であり、そのポインタには次のデータのアドレスが入っている。また、“名古屋”はリストの最後であり、そのポインタには0が入っている。アドレス150に置かれた“静岡”を、“熱海”と“浜松”の間に挿入する処理はどれか。

先頭へのポインタ	アドレス	データ	ポインタ
10	10	東京	50
	30	名古屋	0
	50	新横浜	90
	70	浜松	30
	90	熱海	70
	150	静岡	

- ア 静岡のポインタを50とし、浜松のポインタを150とする。
- イ 静岡のポインタを70とし、熱海のポインタを150とする。
- ウ 静岡のポインタを90とし、浜松のポインタを150とする。
- エ 静岡のポインタを150とし、熱海のポインタを90とする。

問2

スタックに関する記述として、適切なものはどれか。

- ア 最後に格納したデータを最初に取り出すことができる。
- イ 最初に格納したデータを最初に取り出すことができる。
- ウ 先頭要素からたどっていくことで、データを順番に取り出すことができる。
- エ 探索キーをアドレスに変換することによって、データを取り出すことができる。

問3

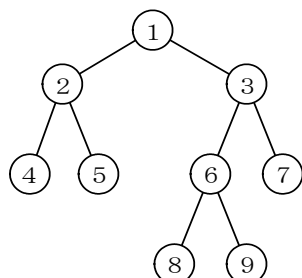
逆ポーランド記法では、計算式“ $X = (A - B) \times C$ ”を“ $XAB - C \times =$ ”と表現する。計算式“ $X = (A + B) \times (C - D \div E)$ ”を逆ポーランド記法で表現したものはどれか。

- ア $XAB + CDE \div - \times =$
- イ $XAB + C - DE \div \times =$
- ウ $XAB + EDC \div - \times =$
- エ $XBA + CD - E \div \times =$

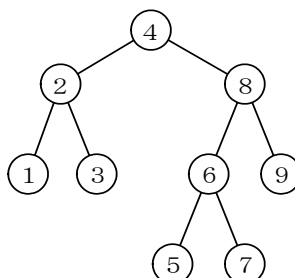
問4

2分探索木として適切なものはどれか。ここで、1～9の数字は、各ノード（節点）の値を表す。

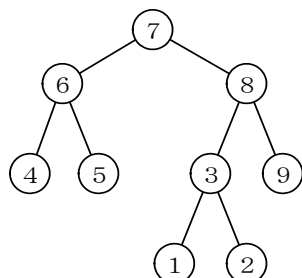
ア



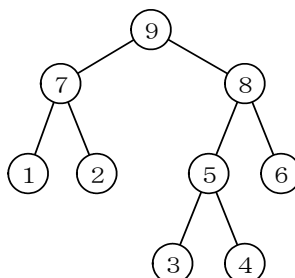
イ



ウ



エ



問5

理解しやすいプログラムにするために、基本構造として、順次、選択、繰返しだけを用いることを原則とするプログラミングの手法はどれか。

ア 関数型プログラミング

イ 構造化プログラミング

ウ 並列プログラミング

エ 論理型プログラミング

問6

アルゴリズムの基本構造に関する説明のうち、適切なものはどれか。

ア “後判定繰返し” は、繰返しの入れ子構造を表現するために使用する。

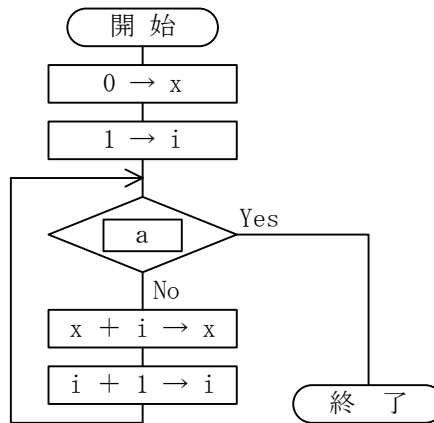
イ “双岐選択” は、二つの処理を並列的に実行する。

ウ “多岐選択” は、直前の処理に戻るときに利用する。

エ “前判定繰返し” は、繰返し処理を一度も実行しないことがある。

問7

流れ図は、1からN ($N \geq 1$) までの整数の総和 ($1+2+\cdots+N$) を求め、結果を変数xに入れるアルゴリズムである。流れ図中のaに当てはまる式はどれか。

ア $i = N$ イ $i < N$ ウ $i > N$ エ $x > N$

問8

2分探索法に関する記述のうち、適切なものはどれか。

- ア データが昇順に並んでいるときだけ、正しく探索できる。
- イ データが昇順又は降順に並んでいるときだけ、正しく探索できる。
- ウ データが昇順又は降順に並んでいるほうが、効率良く探索できる。
- エ データの個数が偶数のときだけ、正しく探索できる。

問9

探索方法とその実行時間のオーダの正しい組合せはどれか。ここで、探索するデータ数を n とし、ハッシュ値が衝突する(同じ値になる)確率は無視できるほど小さいものとする。

	2分探索	線形探索	ハッシュ探索
ア	$\log_2 n$	n	1
イ	$n \log_2 n$	n	$\log_2 n$
ウ	$n \log_2 n$	n^2	1
エ	n^2	1	n

問10

n 個のデータを選択ソートで整列するとき、データ同士の比較回数は何回か。

ア $n \times \log n$

イ $n \times (n+1) \div 4$

ウ $n \times (n-1) \div 2$

エ n^2

問11

ヒープソートの説明として、適切なものはどれか。

ア ある間隔で要素を取り出した部分列を整列し、更に間隔をつめた部分列を取り出して整列する。

イ 中間的な基準値を決めて、それよりも大きな値を集めた区分と小さな値を集めた区分に要素を振り分ける。次に、それぞれの区分の中で同様な処理を繰り返す。

ウ 隣り合う要素を比較して、大小の順が逆であれば、それらの要素を入れ替えるという操作を繰り返して行う。

エ 未整列の部分を順序木に構成し、そこから最大値又は最小値を取り出して既整列の部分に移す。これらの操作を繰り返して、未整列部分を縮めてゆく。

問12

ボイヤ・ムーア法の説明として、適切なものはどれか。

ア キー項目で整列されているレコードを順番に読み込みながら、キー項目の値が同じレコードを同一グループとして合計を求める集計アルゴリズムである。

イ 近似解が判明しているとき、近似解における接線の方程式を求めながら真の解に近付けていく n 次方程式の解法アルゴリズムである。

ウ 経路が判明している節点の中で最低コストとなる節点を確定しながら、すべての節点までの最低コストを決定していく最短経路探索アルゴリズムである。

エ 探索対象から抽出した部分文字列の末尾文字が、探索する文字列中に存在するかどうかで部分文字列の移動量を決定する文字列照合のアルゴリズムである。

索引

和文

— あ —

アーカイバ…………… 157
アーカイブ…………… 149, 157
アーラン…………… 253
アイコン…………… 106, 135
アイソクロナス転送…………… 93
アイテム…………… 194
アイデンティティ連携…………… 345
アウトオブオーダー型実行…………… 67
アウトラインフォント…………… 160
アカウントビリティ…………… 179
アカウント機能…………… 179
アキュムレータ…………… 41
アクセシビリティ…………… 133
アクセス…………… 73
アクセス権…………… 179, 202, 227
アクセス時間…………… 73
アクセス制御…………… 179
アクセスタイム…………… 73
アクセス動作…………… 45
アクセスポイント…………… 252, 272
アクセス方式…………… 178, 197
アクセスモード…………… 42
アクセスログ解析…………… 344
アクセスログ分析…………… 143
アクティブ-アクティブ構成…………… 113
アクティブ-スタンバイ構成…………… 113
アセンブラ…………… 186
アセンブラ言語…………… 180, 181
アセンブル…………… 186
圧縮…………… 149
圧縮率…………… 149
アップデート…………… 335
アップロード…………… 257
アドウェア…………… 307
後判定繰返し…………… 367
アドホックモード…………… 273
アトリビュート…………… 212, 214
アドレス指定方式…………… 50
アドレス修飾…………… 49, 50
アドレススキャン…………… 312, 344
アドレス選択機構…………… 44
アドレス選択動作…………… 45
アドレスバス…………… 42

アドレス部…………… 46
アナログRGB…………… 96
アナログ回線…………… 252
アナログ抵抗膜方式…………… 82
アナログ入出力インタフェース…………… 92
アナログ入出力ボード…………… 92
アプリケーションゲートウェイ…………… 341
アプリケーションサーバ…………… 111
アプリケーションセキュリティ…………… 346
アプリケーション層…………… 267, 268
アプリケーションソフトウェア…………… 156, 160
あふれ…………… 37
あふれ域…………… 198
アプレット…………… 182
アムダールの法則…………… 118
誤り制御…………… 281, 287
アルゴリズム設計…………… 386
アレイ…………… 354
アレイプロセッサ…………… 67
アンケート調査…………… 143
暗号化…………… 343
暗号化技術…………… 313
暗号化制御…………… 179
安全性…………… 126
アンダフロー…………… 37
アンチエイリアシング…………… 150
アンパック…………… 29
アンパック10進数…………… 29

— い —

イーサネット…………… 271
イーサネットヘッダ…………… 259
位相変調…………… 279
一意性制約…………… 216, 225
位置決め…………… 73
位置決め時間…………… 73
一時ファイル…………… 196
一様分布…………… 199, 373
一括処理システム…………… 107
一貫性…………… 107, 223
イニシエータ…………… 167
イニシャルコスト…………… 132
イベントドリブン方式…………… 171
意味解析…………… 186, 189
イメージスキャナ…………… 83
イメージセンサ…………… 83
入口対策…………… 344
入れ子構造…………… 367
色の3原色…………… 88, 147
インオーダー型実行…………… 67
インクジェットプリンタ…………… 88

インサーキットエミュレータ	192
インシデントハンドリングマニュアル	330
インストール	97
陰線消去	150
インターネット	5
インターネットVPN	258
インターネットサービスプロバイダ	251
インターネット層	269
インターネット放送	151
インタビュウ法	143
インタフェースボード	94
インタプリタ	190
インタプリタ型ウイルス	306
インタプリタ方式	183
インタラクティブシステム	134
インタラクティブ性	144
インタレースモード	87
インデックス	217
インデックスレジスタ	41
イントラネット	257
インパクトプリンタ	88
インフォメーションアーキテクチャ	133
インフラストラクチャモード	273
陰面消去	150

— う —

ウイルス対策ソフト	335
ウィンドウ	106, 135
ウェアラブルコンピュータ	14
ウェアラブル端末	14
ウェルノウンポート番号	260
ウォームサイト	116
ウォームスタート方式	223
埋込型SQL	218
運用管理	178
運用管理ツール	158
運用コスト	132

— え —

衛星通信サービス	258
エイリアス	202
エージェント	293
エージング	171
液晶ディスプレイ	87
エキスパートレビュー	143
エクストラネット	257
エグゼキューションサイクル	47
エクスセス方式	34
エスケープ処理	346
枝	360
エッジコンピューティング	291

エディタ	192
エミュレータ	192
エラー	304
エルガマル暗号方式	315
遠隔操作型ウイルス	307
遠隔手続呼出し	111
遠隔バックアップ	332
エンキュー	359
演算	49
演算装置	15, 40
エンティティ	214

— お —

応答時間	120
応用ソフトウェア	156, 160
応用プログラム間連携ソフトウェア	158
オーダ記法	373
オーディット機能	179
オートマトン	187
オーバフロー	37
オーバレイネットワーク	257
オーバレイ方式	175
オープンOS	162
オープンソースコミュニティ	165
オープンソースソフトウェア	163
オープンソースライブラリ	164
オブジェクト指向	182
オブジェクト指向言語	180, 182
オブジェクト指向データベース	244
オブジェクトプログラム	157, 186
オブジェクトライブラリ	159
オブティマイザ	218
オペランド	49
オペランド取出し	49
オペランド部	46
オペレーティングシステム	156, 162
親言語方式	218
音声合成装置	89
音声処理	145
音声入力装置	85
音声認識	134
オンデマンド	151
オンプレミス	112
オンライントランザクション処理システム	107
オンラインリアルタイムシステム	107

— か —

カーソル	240
カーディナリティ	214
カード読取装置	85
カーネル	157

カーネルプログラムテスト	124
カーネルモード	157
ガーベジコレクション	174
改ざん	304
開始タグ	184
回線交換サービス	284
回線交換方式	284
回線交換網	284
回線制御	287
回線速度	253
回線容量	253
回線利用率	253
階層型データベース	211
解像度	86, 88, 147
階層モデル	211
階調	147
回転シフト	39
回転待ち	73
回転待ち時間	73
概念スキーマ	218
概念データモデル	211
開発フレームワーク	159
外部キー	216
外部結合	235
外部スキーマ	218
外部整列	374
外部バス	42
外部モデル	214
外部割込み	52
回路設計	55
会話型SQL	218
書換型	77
可逆圧縮	147, 149
課金管理	292
学習機能	134
確信犯	305
拡張UNIXコード	27
拡張現実感	151
拡張バス	42
拡張メモリ	45
加減乗除	32
重合せ法	199, 355
加算器	40
加重平均	62
仮数	33
仮数部	34
カスケード接続	92, 273, 276
カスタマイズ	160
画素	147
仮想移動体通信事業者	258
仮想化	109

仮想記憶管理	173, 175
画像認識	134
仮想表	226
仮想マシン	109, 192
稼働率	126
可変区画方式	173, 174
可変長レコード	195
画面設計	136
可用性	117, 126, 300
カラム型	228
カレントディレクトリ	201
関係	212
関係演算	212, 213
関係代数	212
関係データベース	211
関係モデル	211, 212
監視カメラ	332
緩衝記憶装置	61
環状リスト	357
関数型言語	180
関数従属性	216
間接アドレス指定方式	51, 199
間接コスト	132
完全2分木	361
完全関数従属	216
完全性	126, 300
ガンブラー	309
管理情報ベース	293
関連	215
関連鍵攻撃	314

— き —

キーバリュ型	228
キーフレーム法	150
キーボード	81
キーロガー	307
記憶階層	63
記憶管理	173
記憶スキーマ	218
記憶装置	15
記憶素子	43
記憶部	44
記憶容量	70
機械学習機能	243
機械語	46, 157, 180
機械語命令	46
機械チェック割込み	52
ギガビットイーサネット	277
木構造	360
擬似言語	386
技術的脅威	302, 304

技術的セキュリティ	334
基数	21, 33
奇数パリティ	282
基数変換	23
基数変換法	199, 355
危殆化	304
基底表	226
機能分散	108
木の巡回法	361
揮発性	17, 43
ギブソンミックス	122
基本域	198
基本交換法	376
基本選択法	374
基本挿入法	378
機密性	300
機密保護機能	221, 346
逆総当たり攻撃	308
逆ポーランド記法	189, 362
キャッシュポイズニング	310
キャッシュメモリ	61
キャパシタ	43
キャパシティプランニング	121, 253
キャラクタ同期方式	280
ギャランティ型	253
キャリアアグリゲーション	252
キュー	358, 359
吸収則	56
脅威	302
強制アクセス制御	346
共通応用ソフトウェア	160
共通鍵	314
共通鍵暗号方式	314
共通言語基盤	183
共通中間言語	189
共有ロック	219
記録領域	71

— < —

クイックソート	380
空間計数量	373
偶数パリティ	282
区画方式	173
クッキー	256
駆動装置	90
区分コード	138
区分編成ファイル	198
組	212
組合せ回路	54
組合せチェック	137
クライアント	109

クライアントサーバシステム	109
クライアント証明書	339
クラウドコンピューティング	112
クラウドサービスのセキュリティ	345
クラス	215
クラス図	215
クラスタ	71, 113, 118, 200
クラスタシステム	113
クラスタリング	113
クラスライブラリ	159
クラッカ	304
クラッキング	304
クラッキング対策	334
グラフ	383
グラフィックスアクセラレータ	87
グラフィックスソフト	161
グラフ型	228
グラフ処理	383
クリアスクリーン	87, 333
クリアデスク	333
繰返し	367
クリックジャッキング	309
グリッドコンピューティング	113
クリッピング	150
グループウェア	161
グループ化	232
グループコミットメント機能	110
クローキング	309
グローバルIPアドレス	262
クローラ	257
クロスアセンブラ	190
クロス結合	234
クロスコンパイラ	190
クロスサイトスクリプティング	309
クロスサイトリクエストフォージェリ	309
クロスチェック	116
クロスブラウザ	141
クロックジェネレータ	41
クロック周波数	41, 122

— け —

計数量	373
形式言語	187
継続条件	367
携帯情報端末	14
携帯端末	14
携帯端末セキュリティ	337
携帯電話	14, 337
経路探索	383
ゲートウェイ	276
ゲスト	179

桁落ち	37
桁別コード	138
結合	212, 234
結合則	56
結線論理制御方式	66
決定表	386
検疫ネットワーク	342
言語プロセッサ	157, 186
検索エンジン	257
検査制約	225
原子性	107, 223
原始プログラム	186
検証局	322
現用系	115

— こ —

故意犯	305
コインマイナー	307
広域Ethernet	258
広域イーサネット	258
公開鍵	315
公開鍵暗号方式	315, 319
公開鍵基盤	323
公開鍵証明書	322
光学式マーク読取装置	83
光学式文字読取装置	83
光学ドライブ	77
交換則	56
交換方式	284
後行順	361
交差結合	235
公衆回線	284
更新	213
高水準言語	180
構成管理	292
合成コード	138
構造エディタ	192
構造型データベース	211
構造化チャート	365
構造化定理	366
構造化データ	240
構造化プログラミング	366
構造体配列	355
高速イーサネット	277
構文解析	186, 188
構文木	188
構文図式	188
コード生成	186, 189
コード設計	138
コールドサイト	116
コールドスタート方式	223
コールドスタンバイ方式	115
コールドバック	342
国際通信サービス	258
国際電気通信連合	278
国際電気通信連合-電気通信標準化部門	278
国際標準化機構	25
誤差	36
故障	302
故障率	131
個人情報保護方針	325
誤操作	303
呼損率	253
五大装置	15
固定区画方式	173
固定小数点数	30
固定長レコード	195
コネクション方式	287
コネクションレス方式	287
コネクタバック通信	311
コピーレフト	165
コプロセッサ	17
個別応用ソフトウェア	160
コマーシャルミックス	122
コマンドインタプリタ	158
コマンド方式	218
コミット	222
コミュニケーションサーバ	109
コモンクライテリア	331
コリジョンドメイン	274
呼量	253
コンティンジェンシープラン	324
コンテナフォーマット	144
コンデンサ	43
コンテンション方式	289
コンテンツフィルタリング	336
コンテンツマネジメントシステム	144
コントロールバス	42
コントロールブレーク処理	385
コンパイラ	157, 186
コンパイル	186
コンパイル方式	181
コンピュータウイルス	306
コンピュータウイルス対策	335
コンピュータグラフィックス	150
コンピュータ犯罪	312
コンプライアンスプログラム	325
コンポーネントウェア	159
コンポーネントベースプログラミング	159

— さ —

差	212

サーチ	73, 368
サーチエンジン	257
サーチ時間	73
サーバ	13, 109
サーバ証明書	339
サービス品質	253
サービスプログラム	157, 191
サービス妨害	304
サーバレット	182
災害	302
再帰	358, 386
再帰的アルゴリズム	380
再帰プログラム	193
サイクル時間	73
サイクルタイム	73, 122
再使用可能プログラム	193
最少権限	346
サイジング	121
最短経路探索	383
最適化	186, 189
最適化コンパイラ	190
サイドチャネル攻撃	312
サイト内検索	141
サイトマップ	141
再入可能プログラム	193
再認証	346
サイバーキルチェーン	311
サイバー空間	308
サイバー攻撃	308
サイバー情報共有イニシアティブ	331
サイバーセキュリティ経営ガイドライン	331
サイバーセキュリティ戦略本部	331
サイバーテロリズム	305
サイバーモール	151
再配置可能プログラム	193
再編成	223, 363, 364
詐欺犯	305
索引域	198
索引順編成ファイル	198
削除	213
座席予約システム	5
撮像装置	90
サニタイジング	346
サブディレクトリ	200
サブネット	262
サブネットアドレス	262
サブネットマスク	262
サブリカント	323
差分解読法	314
差分バックアップ	203
サラミ法	312

算術演算命令	46
算術シフト	38
参照情報	201
参照制約	216, 225
サンドボックス	335
三番地文	189
残留リスク	329

— し —

シーク	73
シーク時間	73
シーケンシャルアクセス	197
シーケンシャル法	199
シーケンスチェック	137
シーケンス番号	259
シェアウェア	163
シェーカソート	377
シェーディング	150
ジェネレータ	190
シェル	158
シェルソート	379
時間計算量	373
磁気インク文字読取装置	83
磁気カード読取装置	85
磁気ディスク	68
磁気テープ	79
磁気テープ装置	79
磁気ヘッド	69
字句解析	186, 187
事故	302
時刻認証	323
資産	302
辞書攻撃	308
指数	33
指数部	34
システムソフトウェア	156
システムバス	42
システムファイル	196
システム利用権	179
事前計算攻撃	308
自然言語インタフェース	134
シソーラス検索	244
持続性	107, 223
実記憶管理	173
実現値	212
実効アクセス時間	62, 75
実効アドレス	49
実効アドレス計算	49
実行可能状態	169
実行時コンパイラ	182, 190
実行状態	169

実表	226
シノニム	199, 372
シノニムレコード	199
指標	354
指標アドレス指定方式	50
指標レジスタ	41
シフトJISコード	26
シフト演算	38
シフト命令	46
時分割システム	107
時分割多元接続	275
シミュレータ	151, 192
ジャーナルファイル	222
射影	212
シャドーIT	305
社内規程	333
シャノンの標本化定理	145
従系	115
集合演算	212, 213
集合関数	232
集線方式	286
集中処理システム	108
周波数変調	279
重複チェック	137
周辺装置	17
集約関数	232
終了条件	367
従量制	253
終了タグ	184
主キー	216
主キー制約	225
主記憶装置	17
縮退運転	125
主系	115
出力装置	15, 86
巡回冗長検査	283
循環小数	24, 36
順次	366
順次アクセス	197
順次探索法	366
順序回路	57
順序木	363
純粹リスク	328
順番コード	138
順編成ファイル	198
ジョイスティック	82
商	212
障害回復機能	222
障害管理	179, 292
状況的犯罪予防	303
照合チェック	137

状態遷移図	169, 187
状態遷移表	187
情報アーキテクチャ	133
情報落ち	36
情報源符号化	281
情報資源辞書システム	244
情報資産	302
情報資産の調査・分類	328
情報セキュリティ委員会	330
情報セキュリティインシデント	303
情報セキュリティ基本方針	324
情報セキュリティ訓練	345
情報セキュリティ継続	324
情報セキュリティ諸規程	325
情報セキュリティ早期警戒パートナーシップ	330
情報セキュリティ対策基準	324
情報セキュリティ対策実施手順	324
情報セキュリティポリシ	324, 333
情報セキュリティマネジメント	324
情報セキュリティマネジメントシステム	325
情報セキュリティリスク	328
情報セキュリティリスクアセスメント	328
情報バリアフリー	142
情報量	19
情報漏えい	302, 303
情報漏えい対策	333
証明書失効リスト	322
商用データベース	244
ショートカット	135, 202
ショートカットキー	136
ショートカット法	314
初期コスト	132
除算法	199, 355
ジョブ	167
ジョブ管理	167
ジョブスケジューラ	168
ジョブスケジューリング	168
ジョブステップ	167
ジョブ制御言語	167
処理権限	227
処理時間順方式	171
処理装置	17
ショルダハッキング	303
シリアルATA	96
シリアルインタフェース	92
シリアル伝送	286
シリアルバス	42
シリアルプリンタ	88
シリンダ	69
人為的脆弱性	305
シンクライアント	111

シンククライアントエージェント	111
シンククライアントシステム	111
シンククライアント端末	111
シングルコアプロセッサ	64
シングルサインオン	342
シングルチップマイコン	14, 17
シングルユーザモード	162
シングルリンク手順	287
信号速度	279
人工知能	243
真正性	300
伸張	149
人的脅威	302, 303
人的セキュリティ	333
侵入検知システム	342
侵入防止システム	342
振幅変調	279
シンプソン法	385
シンプレックスシステム	114
シンボリック言語	180
シンボリックリンク	202
信頼性	126, 300
信頼プロセス	179
真理値表	53

— す —

推移関数従属	216
垂直パリティ	282
垂直分散型	108, 109
スイッチ	90
スイッチングハブ	276
水平垂直パリティ	282
水平パリティ	282
水平分散型	108
数値処理	384
スーバスカラ	67
スーパバイザ	52
スーパバイザモード	157
スーパバイブライン	66
スーパユーザ	179
スキーマ	218
スキャナ	83
スキャベンジング	303
スクリーンセーバ	87
スクリプトウイルス	306
スクリプトキディ	305
スクリプト言語	180, 183
スクロールバー	135
図形エディタ	192
スケールアウト	121
スケールアップ	121

スケジューリング	178
スター型	273
スター接続	92
スタイルシート	141
スタック	358
スタックポインタ	358
スタベーション	171
ステガノグラフィ	337
ステルス機能	343
ストアドプロシージャ機能	110
ストアドロジック制御方式	66
ストライピング	119
ストリーマ	79
ストリーミング	144
ストリーミングデータ	242
ストリーム暗号	316
ストリクトルーティング	276
ストレッチング	334
スニッフィング	304
スパイウェア	307
スパニングツリー	276
スパムメール	310
スパムメール対策	336
スプーリング	168
スプール	168
スプリッタ	252
スマートウォッチ	14
スマートグラス	14
スマートデバイス	14
スマートフォン	14, 337
スマートメディア	83
スミッシング詐欺	312
スラッシング	177
スループット	121
スレッド	172
スロットイン方式	76
スワッピング	175
スワップアウト	175
スワップイン	175

— せ —

正規化	34
生起確率	19
正規表現	187
制御装置	15, 40
制御プログラム	157
整合性制約	216
静止画処理	147
脆弱性	305
脆弱性管理	305
脆弱性検査ツール	305

生体認証	319
生体認証装置	84
静的積算	124
静的配列	355
静的パケットフィルタリング	341
静的優先度順方式	171
静的リンキング	191
静電容量方式	82
性能管理	292
政府認証基盤	323
制約	215
整列	233, 374
整列処理	385
整列プログラム	192
積	212
赤外線遮光方式	82
責任追跡性	300
セキュアOS	346
セキュアブート	336
セキュアプログラミング	346
セキュアプロトコル	338
セキュリティ管理	179, 292
セキュリティ対策ソフト	335, 345
セキュリティトークン	318
セキュリティバイデザイン	305, 346
セキュリティホール	305
セキュリティワイヤ	332
セクタ	68
セクタ方式	71
セグメントページング方式	176
セグメント方式	175, 176
施錠管理	332
窃取	304
セッションアットワンス	77
セッション鍵暗号方式	316
セッション層	267
セッションハイジャック	310
絶対アドレス指定方式	50
絶対誤差	37
絶対バス	201
節点	360
接頭語	20
セマフォ	172
セマフォ方式	220
セマンティック検索	257
セル	285
セルリレー	285
ゼロ知識証明	318
ゼロデイ攻撃	312
全加算器	60
線形解読法	314

線形探索法	368
線形リスト	357
先行順	361
センサ	90
センサネットワーク	291
選択	212, 366
選択ソート	374
選択の知覚	134
センタバッチ処理	107
セントロニクス	96
全二重通信	286
全文検索型	257
専有ロック	219
専用線	284
専用線サービス	258, 284
専用プロセッサ	17

— そ —

総当たり攻撃	308, 314
相關副照会	238
相關名	235
双岐選択	366
総所有コスト	132
増設メモリ	45
相対アドレス指定方式	50
相対誤差	37
相対バス	201
相対編成ファイル	198
挿入	213
挿入ソート	378
増分バックアップ	203
双方向リスト	357
添字	354
ソーシャルエンジニアリング	303
ソースプログラム	157, 186
ソースライブラリ	159
ソート	374
ソートマージ結合法	236
ゾーニング	332
ゾーン10進数	27, 28
即時処理システム	107
属性	212
即値アドレス指定方式	51
疎結合マルチプロセッサシステム	118
組織における内部不正防止ガイドライン	331, 333
ソフトウェア	8
ソフトウェア開発支援ツール	158
ソフトウェア障害	179
ソフトウェアモニタリング	124
ソフトウェアライセンス	163
ソフトリアルタイムシステム	107

ソリッドステートドライブ	78
ソルト	334

— た —

ダークウェブ	305
ターミネータ	92, 167
ターンアラウンドタイム	120
第一正規化	216
待機系	115
大規模並列処理	113
ダイクストラ法	383
台形公式	384
第三者中継	310
第三正規化	216
耐タンパ性	317
ダイナミックアクセス	197
第二正規化	216
タイマ割込み	52
タイムクウォンタム	171
タイムシンクロナス認証	318
タイムスタンプ認証	323
タイムスライス方式	168, 171
ダイヤルアップルータ	276
対話型処理システム	106
ダウンロード	257
ダウンロード型	307
楕円曲線暗号方式	315
多岐選択	366
タグ	184
タグVLAN	277
多次元配列	355
多重化システム	117
多重固定区画方式	173
多重度	215
多重バックアップ	203
多重プログラミング	166, 169
多重プロセッサシステム	118
タスク	169
タスク管理	169
多層ニュートラルネットワーク	243
多層防御	344
多段階認証	317
タッチスクリーン	82
タッチパッド	82
タッチパネル	82
タブル	212
ダブルデータレート	41
タブレット	82
タブレット端末	14, 337
多分木	360
多要素認証	317

タワー型サーバ	13
単一固定区画方式	173
探索	368
探索木	362
単精度	36
単精度浮動小数点数	34
タンデムシステム	114
ダンプスターダイビング	303
単方向通信	286
単方向リスト	357
端末インタフェース	278
端末利用権	179

— ち —

チェーン法	199, 372
チェックキャラクタ	137
チェックサム	282
チェックディジット	139
チェックディジットチェック	137
チェックポイント	222
チェックポイントファイル	222
チェックボックス	135
逐次探索法	368
蓄積交換方式	284
チャタリング	81, 90
チャネル	91
チャネル制御方式	91
チャネルプログラム	91
チャレンジレスポンス認証	318
チャンク	133
中央処理装置	17
中間言語	186
中間者攻撃	310
中間順	361
抽象構文木	188
帳票設計	137
重複チェック	137
調歩同期方式	280
直積	212, 234
直接アクセス	197
直接アドレス指定方式	51, 199
直接記憶アクセス制御方式	91
直接コスト	132
直接制御方式	91
直接編成ファイル	198
直列システム	114, 128
直列伝送	286

— つ —

追記型	76
ツイストペアケーブル	271

通信管理システム	158
通信サービス	258
通信制御装置	90, 278
通信プロトコル	178
通信路符号化	281
ツリー型	273
ツリー構造	360
ツリー接続	92

— て —

ディープラーニング	243	データベースセキュリティ	346
定額従量制	253	データベース操作機能	218
定額制	253	データベースソフト	161
定義域	212	データベース定義機能	218
ディザスタリカバリ	324	データベースの概念設計	214
ディジーチェーン接続	92	データベースの物理設計	217
低水準言語	180	データベースの論理設計	216
ディスクアットワンス	77	データマート	242
ディスク暗号化	313	データマイニング	242
ディスクキャッシュ	63, 75	データマッピング	244
ディスクストライピング	119	データモデル	211
ディスクミラーリング	119	データリンク制御	287
ディスパッチ	171	データリンク層	267, 269
ディスパッチャ	171	テーブル設計	216
ディスプレイ	86	テキストエディタ	192
ディレクトリ	200	テキストボックス	135
ディレクトリ型	257	デキュー	359
ディレクトリサービス	179	テキストチャマッピング	150
ディレクトリトラバーサル	309	出口対策	344
データ圧縮	281	デコーダ	40
データアドミニストレーション	242	テザリング	252
データウェアハウス	242	デシジョンテーブル	386
データ管理	178	デジタイザ	82
データ記憶／記述言語	218	デジタル回線	252
データキャッシュ	62	デジタルカメラ	83
データクリーニング	242	デジタルサイネージ	89
データクレンジング	242	デジタル証明書	322
データ操作言語	218	デジタル署名	321
データ定義言語	218	デジタルトランスフォーメーション	243
データディクショナリ	214, 244	デジタルビデオカメラ	83
データ転送	73	デジタルフォレンジックス	337
データ転送時間	73, 74	デスクトップPC	13
データ転送速度	74	デスクトップディレクトリ	201
データの正規化	216	手続型言語	180, 181
データパス	42	デッドロック	220
データ分析	214	デバイスドライバ	92, 97
データベース	210	デバイスマネージャ	97
データベースアクセス層	110	デファクトスタンダード	27, 268
データベース管理システム	158, 218	デフォルトゲートウェイ	276
データベースサーバ	109	デフラグメンテーション	74
データベース制御機能	219	デュアルシステム	116
		デュアルライセンス	165
		デュプレックスシステム	115
		テレビゲーム	151
		テレマティクス	252
		電気通信事業者	251
		電気電子学会	94, 96
		電子署名	321
		電子透かし	337
		電子部品産業協会	94
		電子ペーパー	89

電子メール	255
伝送制御	287
伝送制御手順	287
転送速度	253
転送命令	46
電力線通信	258

— と —

ド・モルガンの法則	56
動画処理	148
透過性	241
動画認識	134
同期	118, 280
投機実行	65
同期制御	287
投機的リスク	328
等結合	234
統合脅威管理	344
同軸ケーブル	271
導出表	226
到着順方式	171
盗聴	304
動的SQL	240
動的アクセス	197
動的アドレス変換機構	176
動的再配置可能プログラム	193
動的配列	355
動的パケットフィルタリング	341
動的優先度順方式	171
動的リンキング	191
動的リンクライブラリ	159, 191
盗難	302
等幅フォント	160
登録局	322
登録簿	198
トークンバス	275
トークンバスシング	275
トークンリング	275
ドキュメント型	228
特徴抽出	134
独立言語方式	218
特化	215
特権的アクセス権	333
特権モード	52
ドット	86, 88, 147
トポロジ	273
ドメイン	212
ドライブバイダウンロード	309
トラック	68
トラックアットワンス	77
トラックボール	82

トラッシング	303
ドラム型プロッタ	89
トランザクションファイル	196
トランスポート層	267, 269
トランスレータ	190
トレイ方式	76
トロイの木馬	307
ドロー系ソフトウェア	150, 161

— な —

内閣サイバーセキュリティセンター	331
内部結合	235
内部スキーマ	218
内部整列	374
内部バス	42
内部不正	303
内部モデル	214
内部割込み	52
流れ図	365
ナビゲーション	141
なりすまし	303, 304

— に —

二重更新	219
偽セキュリティ対策ソフト	345
偽セキュリティ対策ソフト型ウイルス	306
日本産業規格	26
日本産業標準調査会	26
入出力インタフェース	92
入出力管理	178
入出力制御方式	91, 178
入出力チャンネル	91
入出力バス	42
入出力命令	46
入出力割込み	52, 91, 178
入退室管理	332
ニュートン法	384
ニューメリックチェック	137
入力装置	15, 81
入力チェック	137
人間中心設計	142
認証局	322
認証サーバ	342
認証デバイス	332
認証プロトコル	339

— ぬ —

盗み見	303

— ね —

根	360

ネスト	367
ネストループ法	236
ネットワーク	8
ネットワークOS	293
ネットワークアーキテクチャ	266
ネットワークアドレス	261
ネットワークインタフェース層	269
ネットワーク運用管理	292
ネットワーク型IDS	342
ネットワーク管理	178
ネットワーク管理ツール	293
ネットワークストレージ	111, 119
ネットワーク制御プログラム	178
ネットワークセキュリティ	340
ネットワーク層	267
ネットワークブート	157
ネットワークモデル	211
熱溶解積層方式	89

— の —

ノイマン型アーキテクチャ	42
ノード	360
ノートPC	13
ノンインタレースモード	87
ノンインパクトブリンタ	88
ノンバーバルインタフェース	134
ノンブリエンプション方式	171
ノンブリエンプティブ方式	171
ノンリニア画像編集システム	151

— は —

葉	360
バーコード	4, 84
バーコード読取装置	84
バーコードリーダー	4, 84
バージョン管理	292
バースト誤り	283
パーソナルコンピュータ	13
バーチャルサラウンド	151
バーチャルモール	151
バーチャルリアリティ	151
ハードウェア	7
ハードウェア障害	179
ハードウェアモニタリング	124
ハードディスク装置	69
ハードリアルタイムシステム	107, 162
ハーバードアーキテクチャ	42, 62
バイアス値	34
バイオメトリクス認証	319
廃棄	332
倍精度	36

倍精度浮動小数点数	34
排他制御	219
排他的論理和演算	53
バイト	18, 194
バイナリサーチ法	370
ハイパテキスト	144
ハイパテキストデータベース	244
ハイパフォーマンスコンピューティング	113
ハイパメディア	144
ハイパリンク	256
パイプ機能	162
パイプライン制御	65
ハイブリッド暗号	316
バイポーラ型	43
配列	354
破壊	302, 304
ハクティビズム	305
パケット	259, 285
パケット交換	285
パケット交換サービス	285
パケットフィルタリング	340
パケットライティング	77
ハザード	328
パス	201
バス	42
バス型	273
バスタブ曲線	131
バス幅	42
バスパワー方式	93
パスワード	308, 317
パスワードクラック	308
パスワードクラック対策	334
パスワードリスト攻撃	308
パスワードリマインダ	317
破損	303
パターン認識機能	243
パターンマッチング方式	335
ハッキング	304
バック10進数	29
バックアップ	203
バックアップサイト	116
バックアップファイル	196, 222
バックドア	304
バックドア型	307
バックトラッキング	383
バックトラック法	383
バックワードリカバリ	222
パッケージソフトウェア	163
ハッシュ関数	199, 316, 320, 355, 372
ハッシュ結合法	236
ハッシュ探索法	372

ハッシュ表	355, 372
ハッシュ表探索法	372
ハッシュ編成ファイル	199
ハッシング	199
パッチ処理システム	107
パッチプログラム	335
パッファオーバーフロー	310
パッファリング	168
ハニーポット	337
幅優先探索	361, 383
ハブ	92, 276
ハフマン符号	281
パブリックCA	322
バブルソート	376
ハミング距離	283
ハミング符号	119, 283
ばらまき型	311
パラレルインタフェース	92
パラレル伝送	286
パラレルバス	42
バランス木	364
バランスチェック	137
バリエブル方式	71
パリティチェック	282
パリティ符号	119
パルス符号変調	145, 279
汎化	215
半加算器	58
バンク	64
パンくずリスト	141
半構造化データ	240
半従量制	253
判定	366
半導体メモリ	43, 78
ハンドオーバー	252
半二重通信	286
反復	367
番兵法	369
汎用プロセッサ	17
汎用レジスタ	41

— ひ —

非NULL制約	216, 225
ピアツーピア	108
ヒープ	363
ヒープソート	363, 375
非可逆圧縮	147, 149
比較命令	46
非稼働率	126
光造形方式	89
光ディスク	76

光の3原色	86, 147
光ファイバケーブル	251, 271
ピクセル	147
ピクセルシェーダ	150
非構造化データ	240
ビジネスメール詐欺	312
ヒストリカルファイル	196
非正規形	216
非対話型処理システム	106
ビッグデータ	242
ビット	18, 194
ビット/秒	253
ビット誤り率	253
ビット同期方式	280
ビットマップ	147
ビットマップフォント	160
ヒット率	62, 75
否定演算	53
否定論理積演算	55
否定論理和演算	55
ビデオ・オンデマンド	151
非手続型言語	190
非同期転送モード	285
否認防止	300
非武装セグメント	254, 340
ビヘイビア法	335
秘密鍵	315
秘密鍵暗号方式	314
ビュー	226
ヒューマンインタフェース	133
ヒューリスティック評価	143
ヒューリスティック法	335
表意コード	138
表計算ソフト	160
表制約	225
標的型攻撃	311
標的型攻撃メール	311
標的型攻撃メールに対する訓練	345
標本化	145, 279
標本化周波数	145, 279
表明	225

— ふ —

ファイアウォール	254, 340
ファイル	194, 210
ファイルアクセス権	179
ファイル暗号化	313
ファイル共有	202
ファイル共有ソフト	312
ファイル検索機能	202
ファイル交換ソフト	312

ファイルサーバ	109	不正利用	303
ファイルシステム	200	ブッシュ	358
ファイル処理	385	フットプリンティング	311
ファイル転送サービス	257	物理層	267
ファイル編成方式	178, 198	物理データモデル	211
ファジリング	344	物理的脅威	302
ファストイーサネット	277	物理的セキュリティ	332
ファットクライアント	111	物理レコード	71, 195
ファンクション層	110	不定長レコード	195
フィード	256	浮動小数点数	30, 33
フィールド	212	部分関数従属	216
フィッシング	312	部分集合	212
フィッシング詐欺	312	踏み台攻撃	310
ブート	157	プライオリティスケジューリング	168
ブートストラップ	157	プライバシーバイデザイン	346
ブートセクタ感染型ウイルス	306	プライバシーマーク	325
ブートローダ	157	プライバシーマーク制度	325
フルブルーフ	125, 137	プライバシーセパレータ機能	343
フェールセーフ	125	プライバシーポリシ	325
フェールソフト	125	プライベートCA	322
フェールソフト構造	117	プライベートIPアドレス	262
フェッチサイクル	47	ブラウザ	256
フォーマットチェック	137	ブラウザハイジャッカ	307
フォームオーバーレイ	137	フラグ	370
フォールトアボイダンス	125	プラグアンドプレイ	97
フォールトトレラント	125	プラグインソフトウェア	160
フォールトトレラントシステム	117, 125	フラグ同期方式	280
フォトプリンタ	88	フラグメンテーション	74, 174
フォルダ	202	フラグレジスタ	41
フォワードリカバリ	222	プラズマディスプレイ	87
深さ優先探索	361, 383	ブラックリスト	336
負荷分散	108	フラッシュブートローダ	157
負荷分散クラスタ構成	113	フラッシュメモリ	44
負荷予測	121	フラットベッド型プロッタ	89
不揮発性	17, 44	ブランチ	360
復号器	40	フリーウェア	163
複合キー	216	フリーソフトウェア	163
副照会	237	ブリエンプション方式	171
副スキーマ	218	ブリエンプティブ方式	171
復調	279	ブリコンバイラ	190
副問合せ	237	ブリッジ	276
符号	33	ブリッジ認証局	323
符号化	145, 279	フリップフロップ回路	43, 57
符号付き絶対値表現	30	ブリフィックス値	262
符号部	34	プリプロセッサ	190
符号理論	281	プリンタ	88
不正アクセス	304	プリントサーバ	109
不正アクセス対策	334	ブルートフォース攻撃	308, 314
不正行為	303	ブルーレイディスク	77
不正侵入	302	ブルーレイドライブ	77
不正のトライアングル	303, 333	ブルダウンメニュー	135

フルバックアップ	203
ブレース展開機能	162
ブレースホルダ	346
ブレード型サーバ	13
フレーム	141, 148, 285
フレーム同期方式	280
フレームリレー	285
フレームリレーサービス	285
フレームレート	148
プレゼンテーション層	110, 267
プレゼンテーションソフト	161
ブレンディング	150
フロー制御	287
フローチャート	365
ブロードキャスト	261
ブロードキャストドメイン	276
プロキシサーバ	109, 254, 276
プログラムカウンタ	41
プログラム言語	157
プログラム制御方式	91
プログラムファイル感染型ウイルス	306
プログラムライブラリ	191
プログラムレジスタ	41
プログラム割込み	52
プログレスバー	135
プログレッシブエンハンスメント	141
プロジェクトンマッピング	89
プロジェクト	89, 161
プロセス	169
プロセス管理	169
ブロッキング	71
ブロック	71, 195
ブロック暗号	316
ブロック暗号利用モード	316
ブロック化因数	71
ブロック化係数	71
ブロック間隔	71
ブロックチェーン	323
プロッタ	89
プロトコル	266
プロトコル制御	178
プロトコル番号	259
プロバイダ	251
プロファイル機能	179
プロポーショナルフォント	160
分割統治法	386
分岐命令	46
分散処理システム	108
分散データベース	241
紛失	303
分配則	56

文脈自由文法	187
分離性	107, 223
分類コード	138

— へ —

平均アクセス時間	62, 73, 75
平均位置決め時間	74
平均回転待ち時間	74
平均故障間動作時間	126
平均サーチ時間	74
平均シーク時間	74
平均修理時間	126
併合	381
併合処理	385
併合プログラム	192
米国規格協会	25, 94, 96
米国電子工業会	94
並列システム	115, 129
並列処理システム	113
並列伝送	286
ペイント系ソフトウェア	150, 161
ページアウト	176
ページイン	176
ページ記述言語	88, 161
ベーシック手順	288
ページフォールト	176
ページプリンタ	88
ページリブレースメントアルゴリズム	177
ページング方式	176
ベースアドレス指定方式	50
ベースアドレス方式	176
ベースアドレスレジスタ	41
べき等則	56
ベクトルコンピュータ	67
ベクトルプロセッサ	67
ベストエフォート型	253
ヘッダ	259
ペネトレーションテスト	331, 344
ペリル	328
ベンチマーク	124
ベンチマークテスト	124
ベンチマークプログラム	124
変調	279
変調速度	279

— ほ —

ボイヤ・ムーア法	382
ポインティングスティック	82
ポインティングデバイス	81
ポイントツーポイント方式	273, 286
ボー	279

ボーカロイド	89
ポートVLAN	277
ポートスキャン	312, 344
ポート番号	259, 260
ポートマルチプライヤ	96
ホームディレクトリ	201
ホームレコード	199
ポーランド記法	189, 362
ポーリング／セレクトイング方式	289
ポーリング制御	172
保守性	126
補助記憶装置	17
補数	31
補数器	40, 60
補数表現	31
ホストアドレス	261
ホスト型IDS	342
保全機能	219
保存ファイル	196
ボット	307
ホットサイト	116
ホットスタンバイ方式	115
ボットハーダ	307
ホットプラグ	93, 97
ポップ	358
ポップアップメニュー	135
ボトルネック	121
ポリゴン	150
ポリプロセッサシステム	118
ポリューム	200
ホワイトハッカー	304
ホワイトリスト	336

— ま —

マークアップ言語	184
マージ	381
マージソート	381
マイグレーション	113
マイクロカーネル	157
マイクロコンピュータ	14
マイクロプログラム制御方式	66
マイクロプロセッサ	17
マウス	81
前判定繰返し	367
マクロ	182
マクロウイルス	306
マスクROM	44
マスタスケジューラ	168
マスタファイル	196
待ち行列	359
待ち状態	169

マトリックススイッチ方式	82
マネージャ	293
マルウェア	306
マルウェア対策	335
マルチキャスト	261
マルチコアプロセッサ	64
マルチスキャン方式	86
マルチセッション	77
マルチタスク	169
マルチタスク機能	162
マルチテナント方式	112
マルチブート	157
マルチプラットフォーム	182
マルチプログラミング	169
マルチプロセッサ	67
マルチプロセッサシステム	113, 118
マルチポイント方式	286
マルチメディア	144
マルチメディアオーサリングツール	144
マルチメディアシステム	151
マルチメディアデータベース	244
マルチユーザ機能	162
マルチリンク手順	287
丸め誤差	36

— み —

水飲み場型攻撃	311
密結合マルチプロセッサシステム	118
三つの基本構造	366
ミドルウェア	158
ミラーサイト	116
ミラーリング	119

— む —

無形情報資産	302
無限小数	24
無向グラフ	383
無線LAN	272
無線LANアクセスポイント	272
無線LANインタフェースカード	90
無線LANルータ	252
無手順	288

— め —

命名規約	214
命令アドレスレジスタ	41
命令解説	48
命令解説器	40
命令キャッシュ	62
命令実行段階	47
命令取出し	48

命令取出し段階	47
命令部	46
命令フェッチ	48
命令ミックス	122
命令レジスタ	41
メインメモリ	61
メールアドレス	255, 263
メールクライアント	255
メールサーバ	255
メールヘッダインジェクション	309
メールヘッダインジェクション対策	336
メールボム	310
メタ言語	188
メタデータ	214, 244
メッシュWi-Fi	252
メッシュWi-Fiルータ	252
メッセージ制御	178
メッセージ認証	320
メッセージ認証符号	320
メディアアクセス制御	274
メニューバー	135
メモリ	61
メモリインタリーブ	64
メモリカード	45
メモリコンパクション	174
メモリバス	42
メモリプール	174
メモリリーク	174
メンバ	198

— も —

網型データベース	211
モーションJPEG	148
モーションキャプチャ	151
モーフィング	150
目的プログラム	186
文字コード	25
文字化け	25
モジュール言語方式	218
モジュラス10	139
文字列照合	382
文字列処理	382
モデム	252
モニタリング	124, 178
モノリシックカーネル	157
モバイルWi-Fiルータ	252
モバイル端末	14
モラルハザード	328
問題解決型データ構造	358

— や —

やり取り型	311

— ゆ —

有機ELディスプレイ	87
有形情報資産	302
有限オートマトン	187
有限小数	24
有効アドレス	49
有向グラフ	383
有効桁数	35, 37
ユーザID	317
ユーザアカウント	179
ユーザインタフェース	106
ユーザ管理	178, 179
ユーザビリティ	133
ユーザビリティテスト	143
ユーザビリティ評価	143
ユーザファイル	196
ユーザプログラマブルROM	44
ユーザモード	52, 157
有線LAN	271
有線LANインタフェースカード	90
優先順位方式	168
優先度順方式	171
ユーティリティプログラム	157, 191
愉快犯	305
ユニキャスト	261
ユニバーサルデザイン	142
ユビキタスコンピューティング	291
ユビキタスネットワーク	258

— よ —

四つ組み形式	189
読出し専用型	76
読取り書込み機構	44
より対線	271

— ら —

ライター	167
ライトスルー方式	62
ライトバック方式	62
ライトペン	82
ライブマイグレーション	113
ライブラリ	159
ライブラリモジュール	191
ラインプリンタ	88
ラウンドロビン方式	171
ラジオボタン	135
ラジオボックス	135
ラックマウント型サーバ	13

ラップトップPC	13
ラベル	133
ランサムウェア	307
ランダムアクセス	197
ランニングコスト	132
ランレングス符号	281
ランレングス法	382

— り —

リアルタイムOS	162
リアルタイム処理システム	107
リアルタイム制御処理システム	107
リーダー	167
リーフ	360
リエントラントプログラム	193
リカーシブプログラム	193
リスク	328
リスクアセスメント	328
リスク移転	329
リスク回避	329
リスクコントロール	329
リスク最適化	329
リスク集中	329
リスク受容	328, 329
リスク受容基準	328
リスク対応	328
リスク対応計画	328
リスク低減	329
リスクファイナンス	329
リスク分離	329
リスク保有	329
リスクマネジメント	328
リスク予防	329
リスクレベル	328
リスタート割込み	52
リスト	356
リストボックス	135
リダイレクト機能	158
リッチクライアント	111
リバースブルートフォース攻撃	308
リバースプロキシ	341
リピータ	276
リピータハブ	276
リプレイ攻撃	308
リフレッシュ	43
リポジトリ	244
リミットチェック	137
リムーバブルメディア	76
リモートパッチ処理	107
リユーザブルプログラム	193
利用許諾契約	163

量子化	145, 279
量子化ビット数	145, 279
利用者認証	317
リレーショナルモデル	211, 212
リレーション	212
リレーションシップ	214
リロケーションプログラム	193
リロケータブルプログラム	193
リンカ	191
リング型	273
リンケージエディタ	157, 191
隣接行列	383

— る —

類推攻撃	308
ルータ	252, 276
ルーティング	264
ルーティング制御	287
ルート	360
ルート証明書	322
ルートディレクトリ	200
ループバックアドレス	261
ルーメン	89, 161
ルールベース方式	335

— れ —

レイトレーシング	150
レイヤ2スイッチ	276
レイヤ3スイッチ	276
レインボー攻撃	308
レーザプリンタ	88
レガシーマイグレーション	113
レコード	71, 194
レコード型配列	355
レジスタ	41
レスポンスタイム	120
列制約	225
レッドチーム演習	345
レプリケーション	241
レベル	360
連係ローダ	192
連結リスト	356
レンダリング	150

— ろ —

ローダ	192
ロードシェアリングシステム	118
ロードモジュール	157, 191
ロードライブラリ	159, 191
ローミング	272
ロールアウト	173

ロールイン	173
ロールバック	222
ロールフォワード	222
ロギング機能	179
ログ管理	333
ログデータ分析法	143
ログファイル	222
ロック方式	219
ロボット型	257
論理演算	53
論理演算命令	46
論理回路	53
論理型言語	180
論理関数	55
論理式	55
論理シフト	39
論理積演算	53
論理設計	55
論理チェック	137
論理データモデル	211
論理法則	55
論理レコード	71, 195
論理和演算	53

— わ —

和	212
ワークステーション	13
ワークファイル	196
ワークフロー管理機能	161
ワード	18
ワープロソフト	160
ワーム	306
ワイヤードロジック制御方式	66
ワクチンソフト	335
割込み	52, 157, 169
ワンクリック詐欺	312
ワンタイムパスワード	318
ワンチップマイコン	14, 17

欧文

— A —

A/Dコンバータ	85
A/D変換	145
AACS	77
ACID特性	107, 223
ActiveX	159
Administrator	179
ADPCM	146

ADSL	252
ADSLモデム	252
AES	314, 343
AGP	42
AI	243
Ajax	185, 256
ALU	40, 53
AM	279
AND演算	53
AND回路	54
Android	162
anonymous FTP	339
ANSI	25, 94, 96
ANY接続拒否機能	343
Apacheライセンス	165
Apache Hadoop	165, 243
Apache HTTP Server	164
Apache Kafka	165
Apache Spark	165
Apache Storm	165
Apache Tomcat	165
API	158
APOP	338
APT攻撃	311
AR	151
arp	293
ARP	264, 265, 269
ASCIIコード	25
ATM	285
ATM交換	285
ATMサービス	285
ATMシステム	5
ATM-LAN	277
Availability	126
AVI	148
AVL木	364

— B —

B木	364
BaaS	112
BASE64	255
BCA	323
BCDコード	28
BIND	164
BIOS	97
Bitcoin	307
BLE	291
BLEビーコン	291
Bluetooth	95
BMP	147
BNF記法	188

BOOTP	268
bps	253
BSD系OS	164
BSDライセンス	165
BSDL	165
BSSID	272
BYOD	337

— C —

C	181
C&Cサーバ	307
C++	182
CA	322
CAD	151
CAPTCHA	323
CASL II	181
CATV	258
CCD	83
CCMP	343
CCU	278
CD	76
CD-R	76
CD-R/RWドライブ	77
CD-ROM	76
CD-ROMドライブ	77
CD-RW	77
CDM	286
CDN	256
CG	150
CGI	183, 256
CHAP	339
Chrome	164
CIDR	262
CIL	189
CISC	66
CISO	330, 333
CLI	183
CMOS	43
CMS	144
CMY	88, 147
COBOL	181
Code128	84
cookie	256
Copyleft	165
CORBA	159, 269
CPAN	164
CPI	122
CPRM	79
CPU	17
CPU外部バス	42
CPU内部バス	42

CRC	283
CRL	322
CRTディスプレイ	87
CRYPTREC	331
CRYPTREC暗号リスト	331
CSIRT	330
CSIRTガイド	330
CSIRTマテリアル	330
CSMA/CA	274
CSMA/CD	274
CSRF	309
CSS	141, 184
CSV形式	148
CTI	278
CVSS	305

— D —

D/Aコンバータ	89
D/A変換	146
DaaS	112
DAT	79, 176
DBMS	158, 218
DD/D	214, 244
DDL	218
DDoS攻撃	310
DDR	41
DDR SDRAM	43
DDS	79, 203
DH	314
DHCP	264, 268
DHCPクライアント	264
DHCPサーバ	254, 264
DHTML	184
DIMM	45
DisplayPort	96
DKIM	336
DLL	159, 191
DLP	345
DMA制御方式	91
DML	218
DMZ	254, 340
DNS	263, 268
DNSキャッシュポイズニング	310
DNSサーバ	254, 263
DNSSEC	339
DOM	185
DoS攻撃	310
dpi	86, 88, 147
DRAM	43
DSA	321
DSDL	218

DSU	278
DTD	184, 185
DTPソフト	161
DVD	76
DVD-R	76
DVD-R/RWドライブ	77
DVD-RAM	77
DVD-ROM	76
DVD-ROMドライブ	77
DVD-RW	77
DVI	96

— E —

E-R図	214
E-Rモデル	214
EAL	331
EAP	339
EAP-TLS	339
EAP-TTLS	339
EBCDIC	27
EC	257
ECC	283
ECIA	94
Eclipse	164
ECMAScript	183
EDI	257
EEPROM	44
EIA	94
EOR演算	53
ESC/Page	88, 161
ESSID	272
Ethernet	271
ETL	242
EUC	27
Exif	147

— F —

FaaS	112
FAT32	200
FATファイルシステム	200
FCFS方式	168
FDDI	277
FDM	286
FHD	86
FIFO	359
FIFO法	177
Firefox	164
FLOPS	123
FM	279
fps	148
FQDN	263

FreeBSD	164
FTP	257, 268
FTPサーバ	257
FTPS	338
FTTH	251
FUHD	86

— G —

GCC	165
GIF	147
GIPS	123
GNU	165
GNUプロジェクト	165
GPIB	96
GPKI	323
GPL	165
GPU	17
GUI	135
GZIP	149

— H —

H. 264/MPEG-4 AVC	148
H. 265	149
HAクラスタ構成	113
HD	86
HDB	213
HDLC手順	289
HDMI	96
HEIF	147
HEVC	149
HFS	200
HIDS	342
HPC	113
HTML	111, 184
HTTP	256, 268
HTTPヘッディングエクシジョン	309
HTTPS	338

— I —

IaaS	112
IBG	71
IC	14
ICカード	317
ICカード読取装置	85
ICメモリ	43, 78
ICE	192
ICMP	264, 269
ID連携トラストフレームワーク	345
IDE	96
IDEA	314
IDF	251

IDS	342
IEEE	94, 96
IEEE 802.1X	323
IEEE 802.3	271
IEEE 802.11	272
IEEE 1394	94
ifconfig	293
IMAP	268
IMAP4	255
Integrity	126
iOS	162
IoT	242, 291
IoTエリアネットワーク	291
IP	260, 269
IPアドレス	259, 260
IPスプーフィング	310
IP層	269
IP電話	258
IPヘッダ	259
IPマスカレード	263, 344
IP-VPN	258
IPA	330
IPAセキュリティセンター	331
ipconfig	293
IPL	157
IPoE	269
IPS	342
IPsec	338
IPv4	260
IPv6	260
IrDA	95
IRDS	244
IRIX	164
ISMS	325
ISMS適合性評価制度	327
ISO	25
ISO 9241-210	142
ISOコード	25
ISO/IEC 15408	331
ISO/IEC 17799	325
ISO/IEC 23271	183
ISO/IEC 27001	325
ISO/IEC 27002	325
ISP	251
ITFコード	84
ITU	278
ITU-T	278

— J —

J-CSIP	331
JANコード	84

Java	182
Javaアプリケーション	182
Java仮想マシン	182
Javaバイトコード	189
Java Applet	182
Java Servlet	182
JavaBeans	159, 182
JavaScript	183
JCL	167
JCMVP	331
JDBC	244
JEIDA	45
JIS	26
JIS 7単位符号	26
JIS 8単位符号	26
JIS漢字コード	26
JISコード	26
JIS Q 15001	325
JIS Q 27001	325
JIS Q 27002	325
JIS X 3016	183
JIS X 5080	325
JIS X 8341	133, 142
JIS Z 8520	134
JIS Z 8521	133
JIS Z 8530	134, 142
JISC	26
JISEC	331
JITコンパイラ	182, 190
JPCERT/CC	330
JPEG	147, 149
JPNIC	260
jQuery	164
JVM	182
JVN	330

— K —

KCipher-2	314

— L —

L2スイッチ	276
L3スイッチ	276
LAMP	164
LAN	178, 271
LANアナライザ	293
LANカード	271
LAN間接続装置	276
LANボード	271
LANモジュール	271, 274
LAPP	164
LCD	87

LDAP	179
LFU法	177
LGPL	165
LIFO	358
Linux	162, 164
Linuxカーネル	164
lm	89, 161
LPWA	291
LRU法	177
LSI	14
LTE	252

— M —

MAC	274, 320, 346
MACアドレス	259, 265, 274
MACアドレスフィルタリング機能	343
macOS	162
MariaDB	164
MDF	251
MDM	337
MEMS	89
MHS	255
MHz	41
MIB	293
MICR	83
microUSB	93
MIDI	146
MIL記号	54
MIMD	67
MIME	255, 268
miniUSB	93
MIPS	122
MISD	67
MMR	149
MOS型	43
MOV	148
MP3	146
MP4	148
MPEG	148, 149
MPL	165
MPU	17
MQTT	291
MR	149
MTBF	126
MTTR	126
MVNO	258
MySQL	164

— N —

n進木	360
n進数	21

Nバージョンプログラミング	117
NAND演算	55
NAND回路	55
NAPT	263, 344
NAS	119
NAT	263, 344
NCU	278
NDB	211
need-to-know	333
NetBSD	164
netstat	293
NetWare	293
NFC	95
NFP	62, 75
NIC	260, 271
NIDS	342
NISC	331
NNTP	268
NOR演算	55
NOR回路	55
NoSQL	228, 243
NOT演算	53
NOT回路	54
NSチャート	365
NTFS	200
NTP	268
NULL	216

— O —

OAuth	339
OCR	83
OCSP	322
OECD	301
OECDセキュリティガイドライン	301
OECDプライバシーガイドライン	301
OLAP	242
OLTP	107, 242
OMG	215
OMR	83
ONU	251
OODB	244
OP25B	336
OpenBSD	164
OpenFlow	258
OpenID	345
OR演算	53
OR回路	54
OS	156, 162
OSアップデート	335
OSコマンドインジェクション	309
OSD	163

OSI····· 163, 266
 OSI基本参照モデル····· 178, 266
 OSS····· 163
 OSSライセンス····· 165

— P —

P2P····· 108, 275
 PaaS····· 112
 PAD····· 365
 PBX····· 278
 PC····· 13
 PC用OS····· 162
 PCI····· 42
 PCI DSS····· 331
 PCI Express····· 42
 PCM····· 145, 279
 PCMCIA····· 45
 PDA····· 14
 PDCAサイクル····· 326
 PDF····· 144
 PDL····· 161
 PDP····· 87
 PEAP····· 339
 PEAR····· 164
 Perl····· 164, 183
 PGP····· 338
 PHP····· 164, 183
 PIN····· 317
 ping····· 293
 PKI····· 323
 PLC····· 258
 PM····· 279
 PNG····· 147
 PnP····· 97
 POP····· 268
 POP3····· 255
 POP3S····· 338
 POSシステム····· 4
 PostgreSQL····· 164
 PostScript····· 88, 161, 183
 ppm····· 88
 PPP····· 269
 PPPoE····· 269
 PROM····· 44
 PROXYサーバ····· 109, 254, 276
 PSK····· 343
 PSW····· 41
 Python····· 164, 183

— Q —

QFHD····· 86

QoS····· 253, 257
 QRコード····· 84
 QuickTime····· 148

— R —

RA····· 322
 RADIUSサーバ····· 254, 323, 342
 RAID····· 119
 RAID0····· 119
 RAID1····· 119
 RAID2····· 119
 RAID3····· 119
 RAID4····· 119
 RAID5····· 119
 RAID6····· 119
 RAM····· 43
 RAMディスク····· 78
 RAMファイル····· 78
 RAMファイルシステム····· 78
 RARP····· 269
 RAS····· 126
 RAS技術····· 332
 RASサーバ····· 254
 RASIS····· 126
 RAT····· 307
 RDA方式····· 241
 RDB····· 211
 Reliability····· 126
 RGB····· 86, 147
 RIFF····· 148
 RIP····· 264, 269
 RISC····· 66
 RJE端末····· 107
 ROM····· 44
 root····· 179
 rootkit····· 304
 RPC····· 111
 RS-232C····· 94
 RSA····· 315
 RSS····· 256
 RSSリーダー····· 256
 RTP····· 268
 Ruby····· 164, 183

— S —

S/MIME····· 338
 SaaS····· 112
 SAML····· 345
 SAN····· 119
 SAX····· 185
 SCSI····· 94

UTM	344
UV-EPROM	44
UX	142
UXデザイン	142

— V —

VA	322
VAN	258
VB	182
VBA	182
VBScript	183
VDI	112
VGA	86
VLAN	277
VLIW	67
VLSI	14
VLSM	262
VM	109, 192
VoIP	258
VoLTE	252
VPN	258, 344
VR	151
VRAM	86
VRML	148
VSAM編成ファイル	198
VUI	134

— W —

W3C	140
WAF	346
WAI	142
WAI-ARIA	142
WAN	178
WAV	146
WCAG 1.0	142
WCAG 2.0	140, 142
WDM	286
Web	256
Webアクセシビリティ	140
Webカメラ	83
Webクライアント	256
Webコンテンツ	144
Webサーバ	111, 256
Webシステム	111
Webデザイン	140
Webビーコン	256
Webブラウザ	111
Webユーザビリティ	140
Web API	158
WEP	343
Wi-Fi	252, 272

Wi-Fiルータ	252
Wiki	144
Windows	162
WMA	146
WMV	148
WPA	343
WPA2	343
WPA3	343
WSDL	185
WWW	256
WYSIWYG	161

— X —

XGA	86
XHTML	185
XHTML Basic	185
XML	185
XMLデータベース	244
XMLデジタル署名	321
XMLパーサ	185
XML Schema	185
XOR演算	53
XOR回路	55
XSL	185
XSS	309
XYプロッタ	89

— Y —

YAML	185

— Z —

Zigbee	95
ZIP	149

数 字

0アドレス命令	46
1アドレス命令	46
1語長命令	46
1次キャッシュ	63
1次元バーコード	84
1次元配列	355
1相コミットメント	241
1の補数器	60
2アドレス命令	46
2語長命令	46
2次キャッシュ	63
2次元バーコード	84
2進化10進コード	28

2進数	21
2進セマフォ	172
2層クライアントサーバシステム	110
2相コミットメント	241
2の補数器	60
2の補数方式	34
2分木	360
2分探索木	362, 371
2分探索法	368, 370
2要素認証	317
3アドレス命令	46
3次元映像	151
3層クライアントサーバシステム	110
3層スキーマ	218
3D	151
3Dスキャナ	83
3Dプリンタ	89
3Dプロッタ	89
3DCG	151
4K	86
5G	252
8K	86
8進数	21
10進数	21
16進数	22
100BASE-FX	277
100BASE-T	277
1000BASE-FX	277
1000BASE-T	277

写真提供

アイメックス株式会社

シャープ株式会社

日本マイクロソフト株式会社

- ・ Microsoft®, Microsoft® Windows® は米国 Microsoft Corporation の米国およびその他の国における登録商標です。
- ・ 本書に記載されている各製品名は、それぞれのメーカーの商標または登録商標です。

ITワールド

発行日 2003 年 1 月 10 日 初 版 第 1 刷
2023 年 12 月 1 日 第 24 版 第 1 刷

著 者 インフォテック・サーブ教育研究会

発行者 木田 徳彦

発行所 株式会社 インフォテック・サーブ

〒101-0052

東京都千代田区神田小川町 1-8-5 金石舎ビル 7F

TEL 03(5289)9321

FAX 03(5289)9320

URL <http://www.infotech-s.co.jp>

©インフォテック・サーブ 2023 ISBN978-4-909963-71-0

無断複写・複製、転載は著作権の侵害となります。