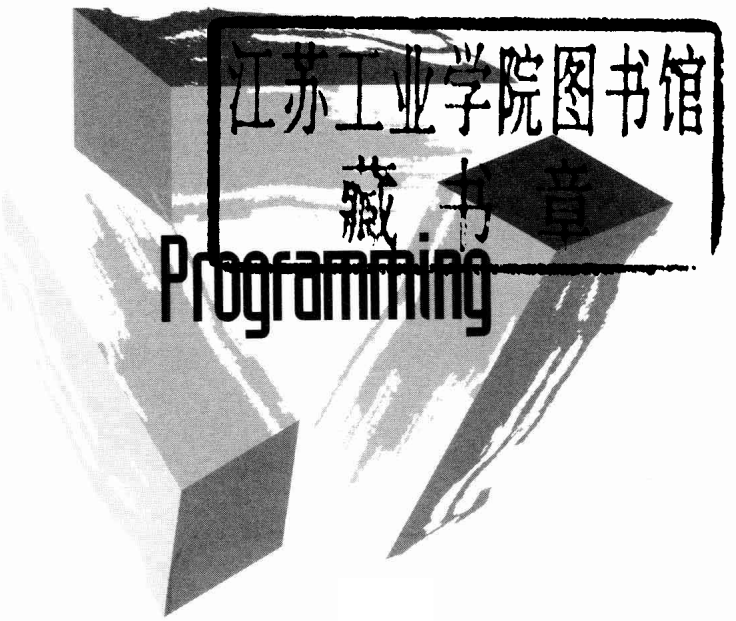
文科系のための
プログラミング論

河村一樹／斐品正照［著］

日刊工業新聞社

文科系のための
プログラミング論

河村一樹／斐品正照［著］



江苏工业学院图书馆
藏书章

日刊工業新聞社

著者紹介

河村　一樹（かわむら　かずき）

1955年東京生まれ．立教大学理学部化学科卒業，日本大学大学院理工学研究科博士前期課程電子工学専攻修了．現在，宮城大学事業構想学部デザイン情報学科助教授．

著書：コンピュータ科学論（ダイゴ），コンピュータ科学入門（実教出版），コンピュータ科学基礎（ITEC），コンピュータ科学基礎（共著，中央情報教育研究所），標準コンピュータ教科書（共著，オーム社），情報システム設計・開発技術（近代科学社），システムの設計と開発（共著，ムイスリ出版），ソフトウェア工学（ソフトバンク），ソフトウェア工学入門（近代科学社），ソフトウェア工学（中央情報教育研究所），マルチメディア社会と情報教育（共著，柴峰図書）など多数

斐品　正照（ひしな　まさてる）

1972年山口県生まれ．大阪府育ち．大阪電気通信大学短期大学部電子情報学科卒業，大阪電気通信大学工学部経営工学科卒業，大阪電気通信大学大学院工学研究科博士前期課程情報工学専攻修了．現在，宮城大学事業構想学部デザイン情報学科助手．

著書：情報社会で役立つ情報教育の知恵（共著，パワー社）など

文化系のための
プログラミング論

発行日　2000年　9月25日　初版第1刷

©著　者　河　村　一　樹
　　斐　品　正　照
発行者　菅　野　泰　平
発行所　日刊工業新聞社
　　〒102-8181　東京都千代田区九段北1丁目8番10号
　　電話　編集部　03-3222-7090～7092
　　販売部　03-3222-7084・7131
　　振替口座　00190-2-186076
印　刷　美研プリンティング㈱　製　本　越後堂

（定価はカバーに表示してあります）
万一乱丁，落丁などの不良品がありましたらお取り替えいたします。
ISBN 4-526-04634-5　C 3034　NDC 007.63　2000 Printed in Japan
㊐〈日本複写権センター委託出版物〉
本書の無断複写は，著作権法上での例外を除き，禁じられています。本書からの複写は，日本複写権センター（03-3401-2382）の許諾を得てください。

はじめに

現在我々が使用しているほとんどのコンピュータは，ノイマン型コンピュータと呼ばれている．これには，プログラム内蔵方式が採用されており，各種のプログラムを記憶装置に記憶させることによって，さまざまな利用が可能になる．

そのプログラムには，多くの既製品（基本ソフトウェアから応用ソフトウェアまで）を使うこともできるし，自分でオリジナルに作成することもできる．ただし，自分で作成する場合には，何らかのプログラミング言語を習得している必要がある．

我々が日常文章を読んだり書いたりできるのは，日本語の文法や書法などについての理解があるからである．そのために，初等中等教育機関において国語教育を継続して学習することになる．同様に，プログラムを読み書きするためには，プログラミング言語の文法や書法を習得していなければならない．そのために，高等教育機関ではプログラミング教育を実施しているところが多い．

ただし，プログラミング教育といっても，情報専門系の学科と非情報系の学科（以降，文科系の学科とする）とでは，その学習すべき対象や習得レベルが異なってくる．情報系の学科では実用言語の習得が前提であり，さらに実用言語を使ってより高度なプログラミングが行えるようになることが目的になる．このため，離散数学を基礎として，コンピュータサイエンスの領域（たとえば，形式言語理論とオートマトン，プログラム理論，計算量理論，グラフ理論，データ構造とアルゴリズムなど）に関して，より専門的な内容を学ぶことになる．

これに対して，文科系の学科では，問題解決手順の理解，コンピュータの動作原理の理解，コンピュータの可能性と限界の理解，などのためにプログラミングを取り上げることが多い．そのため，プログラミング言語の文法や構文に

ついてよりも，プログラミングの考え方やその基盤にある基礎的な概念などを学習する必要がある．ここには，統制感を持った健全なコンピュータの利用者という視点から，プログラミングを学ぶという主張がある．

統制感を持った健全な利用者とは，コンピュータを自分で制御しているという実感をもつとともに，不具合が起こっても的確な対処ができる利用者である．そのためには，コンピュータの内部をある程度ホワイトボックス化して理解できている必要があり，文科系の学生に対してもコンピュータサイエンスの基礎的な部分を取り込む必要がある．

本書では，文科系の学科を対象にしてプログラミングを論じることを目指しており，これによって統制感を持った健全な利用者を育成することを目標にしている．このため，コンピュータサイエンスの領域から，その一部分（形式言語理論の基礎，オートマトンの基礎，データ構造とアルゴリズムの基礎）を取り込むことにする．ここが本書の特色でもある．

第1章は，序論ということで，プログラミングに関する全般的な話題を簡潔にまとめている．

第2章では，プログラミング言語ということで，その歴史的な変遷をはじめ，プログラミング言語のいくつかについて取り上げて説明している．ただし，ここではあくまでも歴史的な経緯に留め，それぞれの言語が持つ構文の詳細などについては省いている．これらについては，他のプログラミング言語に関する専門書を参考にして頂きたい．

第3章では，プログラムの構成を，形式的な側面と構成的な側面から解釈する．形式的な側面からは，形式言語理論の基礎を取り上げる．構成的な側面からは，プログラムがどのような構成要素から成り立つのかについて，要素単位から全体構成といったボトムアップ的な展開のもとに取り上げる．ただし，プログラムについては特定のプログラミング言語を意識せずに，一般的なプログラムとしての枠組みにした．

第4章では，データ構造とアルゴリズムについて取り上げる．ここでも，特定のプログラミング言語に依存することなく，一般的なプログラムを前提にしている．「アルゴリズム＋データ構造＝プログラム」という文があるように，

此为试读，需要完整PDF请访问：www.ertongbook.com

データ構造とアルゴリズムは車の両輪といえる．これらの考え方を学ぶことは，問題解決手順を把握する上で有効な手段であり，コンピュータの可能性と限界を理解することにも効果的である．

第５章では，プログラミング言語で記述したプログラムを実行するための手順について取り上げる．その中のプログラムの実行制御において，オートマトンの基礎を取り上げる．万能チェリ―ング機械こそ，今日のコンピュータの原型でもあるからである．

第６章では，プログラミング言語の一つとしてＣ言語を取り上げる．そして，Ｃ言語を用いたプログラミングについて，文科系の学生向けの演習例を入れながら具体的に説明する．ここでは，Ｃ言語の構文を説明するというよりも，前章までの内容がどのようにＣ言語によるプログラミングに反映されているのか明らかにすることに重点を置く．したがって，Ｃ言語の習得を目標とするのではなく，あくまでも問題解決手段としてＣ言語のプログラミングを行うという視点にもとづいている．

第７章では，Ｃ言語によるプログラム開発手順について取り上げる．プログラムを設計し開発するまでの工程とその中での作業概要について説明する．

なお，本書の執筆にあたっては，第１章から第５章までと第７章を河村が，第６章を斐品が，それぞれ担当した．また，第６章のＣ言語によるプログラミング例については，我々の研究室の４年生である昆野美奈，佐藤敬之，松浦智博諸君の協力を得た．

本書が，高等教育機関の文科系向けのプログラミング教育における一つの教材となれば幸いである．

2000年９月

著者を代表して

河村　一樹

目　次

第3章　プログラムの構成

第4章　データ構造とアルゴリズム

第5章　プログラムの実行環境

第6章　C言語によるプログラミング

第7章　プログラムの設計と開発

此为试读,需要完整PDF请访问: www.ertongbook.com

第1章

プログラミング序論

本章は，本書のタイトルにもあるプログラミング論ということから，プログラミングあるいはプログラム全般に関する基本的な話題について取り上げる．具体的には，プログラムとは何かということについて，JISの情報処理用語での定義を参照しながら論じる．その上で，プログラミング言語について，データ構造とアルゴリズムの関係，などを論じる．

1.1 プログラムとは

一般的に使われているプログラムという言葉は，目録とか番組あるいは計画という意味で使われる．たとえば，音楽会で配布される演奏プログラムや，学校の運動会で配られる競技プログラムなどがあげられる．

これに対して，コンピュータの世界で使われるプログラム（program）という用語は，もう少し限定された意味になる．具体的には，コンピュータに指示命令する段取りをまとめたものという意味で使われている．

現在までに実用化されたコンピュータは，人間があらかじめ与えた命令に対して，忠実にその動作を実行することを前提にしている．その際に，それらの命令の中で，無限に同じ処理を繰り返すようなまちがえがあったとしても，コンピュータはその処理を無限に繰り返すこと（無限ループ）になる．このことからも，コンピュータは，与えられた命令以外に，自らが思考し自律的に動作するといったことが一切できない代物であることがわかる．それだけでなく，人間のように学習する能力もないし感情を表現することもできない．

このように，人間がコンピュータにやらせたい仕事を，逐一指示し命令する

ための手順をまとめたものがプログラムなのである．人間とコンピュータは，プログラムを介して関係し合うとともに，人間にとっては「コンピュータ，プログラムがなければただの箱」と言うことができる．

また，コンピュータが登場した頃は，人間が与えるプログラムというものを，配線の組み換えやスイッチの並びの組合せといった動作によって，逐次コンピュータに与えていた．その後，プログラム内蔵という考え方が提案され，プログラムそのものをコンピュータ内部に記憶させ，必要なときに呼び出して使用するという方式が採用されるようになった．これによって，プログラムは，人間による一連の動作行為といった物理的なものから，記憶装置上に任意のビット列として記憶された目に見えない論理的なものへと変貌することになった．このうちの目に見えないというプログラムの特徴には，記憶装置に記憶されたビット列を実際には見ることができないという意味や，ビット列の構成が理解できないことが見てもわからないことと同様であるという意味を含んでいる．

JIS では，情報処理の用語として，プログラムを次のように定義している[1]．

「アルゴリズムの記述に適した人工言語の規則に従った構文上の単位であって，ある機能若しくは仕事の遂行又は問題の解決のために必要な宣言と文若しくは命令とから構成されるもの．」

このうちのアルゴリズム（algorithm）に関しては，次のように定義している[1]．

「問題を解くためのものであって，明確に定義され，順序付けられた有限個の規則からなる集合．」

これより，アルゴリズムは，問題解決のための手順を表すものであり，人間が考える思考過程そのものであるから，目に見えない抽象的なものといえる．そのアルゴリズムが「明確に定義された」ということは，数理的理論にもとづく厳密な解析が可能であることを示している．また，「順序付けられた有限個の規則」とは，アルゴリズムを構成する個々の規則同士の並びが，逐次的な時間の経過にもとづくとともに，有限の時間の中で一意に決定されることを意味する．したがって，上述したような無限ループを含む規則の集合は，アルゴリズムとは言えないことになる．

人工言語（artificial language）については，

「使用前から規則が明示的に確立されている言語」

と定義している[1]．それに対して，自然言語については，

「明示的に規定された規則ではなく，慣用に基づいた規則をもつ言語」

と定義している[1]．これより，人工言語と自然言語は，お互いに反意語の関係といえる．本来，自然言語は，人間同士におけるコミュニケーションのための道具として用いられる．慣習や例外あるいは流行などに影響され，時代の流れとともに言語の仕様が変化することも多い．このため，「慣用に基づいた規則をもつ言語」と定義されている．自然言語としては，英語や日本語などがあげられる．

これに対して，人工言語は，人間対機械あるいは機械同士におけるコミュニケーションのための道具として用いられる．これより，言語の解釈は機械そのものが行うことから，機械が理解できるように，あいまいさを一切排除した厳密な言語仕様に準拠したものになる．人工言語としては，プログラミング言語があげられる．

なお，本書では，プログラム言語ではなく，プログラミング言語という言葉を用いることにする．これは，プログラミング（programming）は，program という名詞に，ing という接尾語がついて動名詞になっていることから，「プログラムをつくること」という意味になる．これより，プログラムをつくるための言語である人工言語という意味で，プログラミング言語とする．

宣言と文は，いずれもプログラムを構成する要素単位といえる．宣言（declaration）は，コンピュータ資源に対して使用する領域を確保したり，その領域の属性を割り当てる際に用いられる．たとえば，プログラムで用いる変数に関するデータ型の宣言や引数を含めた関数の宣言などがあげられる．

文（statement）は，プログラムの処理の最小単位であり，アルゴリズムを構成する要素単位といえる．これより，プログラムのアルゴリズムは，文による処理のつながりによって構成されるといえる．

1.2　プログラミング言語について

プログラミング言語（programming language）は，プログラムを表現するための人工言語である．つまり，コンピュータに指示命令する一連の動作手順を書き表すために，人間が作り上げた専用の言語である．

プログラミング言語は，コンピュータの技術革新に合わせる形で，その種類を増やしながら機能を拡張しながら発展してきた．新しいコンピュータ技術によるインフラストラクチャが整ったことによって，新しいプログラミング言語が開発され普及するという経緯である．その発展経緯を一言で表すと，「抽象レベルの高度化」ということができる．これは，コンピュータ寄りから人間寄りへの変貌という意味である（図1-1）．

当初のプログラミング言語は，コンピュータが直接解釈できる言語しかなかった．しかし，人間にとっては馴染みにくい言語であり，コンピュータのアーキテクチャをある程度理解していないと使うことができない代物だったといえる．このため，低水準言語とも呼ばれた．

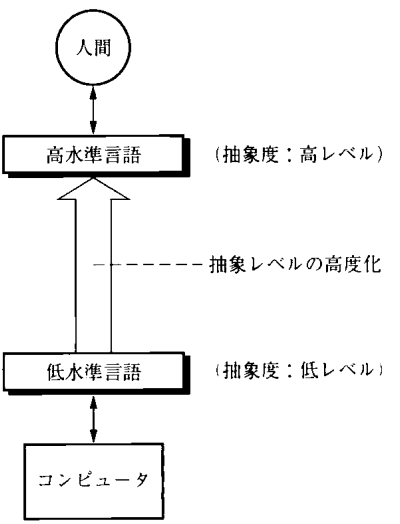


図1-1　抽象レベルの高度化

その後，コンピュータよりも人間にとって理解しやすい言語の開発が進められた．自然言語（ただし，英語が中心）に近く，ある程度の形式的な制約を与えた言語であるとともに，コンピュータのアーキテクチャをそれほど意識しなくても使える言語の開発である．このため，高水準言語と呼ばれた．

ただし，どんな高水準言語でも，最終的にはコンピュータが理解できる低水準言語に変換する必要があり，変換のためのプログラムも各種開発された．それとともに，プログラム開発・管理を支援するためのプログラムも開発され，それらが統合的に機能するプログラム開発環境が整備されるようになった．この結果，プログラム開発は，個人の作業から，プロジェクトによる共同作業へと広がり，その規模も年々大きなものになった．

以上の低水準言語から高水準言語への移行は，コンピュータアーキテクチャからの脱却を実現し，誰でもがプログラミングできるようにするための方策であったといえる．それだけでなく，高水準言語においても，一つの変化を見出すことができる．それは，手続き型から非手続き型への移行である．非手続き型プログラミング言語を開発するためには，より抽象度合いを高める必要がある．その一つに，オブジェクト指向の適用があげられる．これは，状態（データ）と手続き（プロセス）を一体化したオブジェクトを用いて抽象化するというもので，我々の現実世界をそのままモデル化したものといえる．

また，コンピュータ環境におけるインフラストラクチャの拡充によるプログラミング言語への影響としては，GUI 環境とネットワーク環境があげられる．GUI（graphical user interface）環境では，ウィンドウベースのプラットホーム（Ms-Windows，Mac-OS，X-Windows など）があげられる．これにともない，ビジュアルプログラミング言語が，注目をあびつつある．ネットワーク環境では，インターネットがあげられる．これには，インターネットを通じてアプリケーションプログラムを配信できること，配信されたプログラムがマルチプラットホーム対応（Ms-Windows，Mac-OS，UNIX など）であること，といった条件を満たしたインターネット用プログラミング言語があげられる．

以上のプログラミング言語の歴史や種類と特徴については，**第 2 章**で詳細に取り上げることにする．

1.3　アルゴリズムとデータ構造の関係

プログラムは，入力（input）が与えらることによって，一連の処理（process）を実行し，要求通りの出力（output）を生み出すという動作機構を持つ．そこで，プログラムをIPO（Input–Process–Output）によってモデル化すると，図1-2のようになる．これは，プログラムの最も基本的な構成，つまり，どんなプログラムも入力と処理と出力から構成されること，を表している．

一方，入力と出力はそれぞれデータ構造を持つとともに，処理はアルゴリズムそのものでもある．その結果，プログラムは，アルゴリズムとデータ構造の両方から構成されると言うこともできる．

アルゴリズムはプログラムの処理を表したものであり，その処理を実現するために最適なデータ構造を決定する必要がある．このことは，アルゴリズムの構成がデータの構造に強く依存するとともに，データの構造化においてはアルゴリズムの構成をあらかじめ選択しておく必要があることを示している．

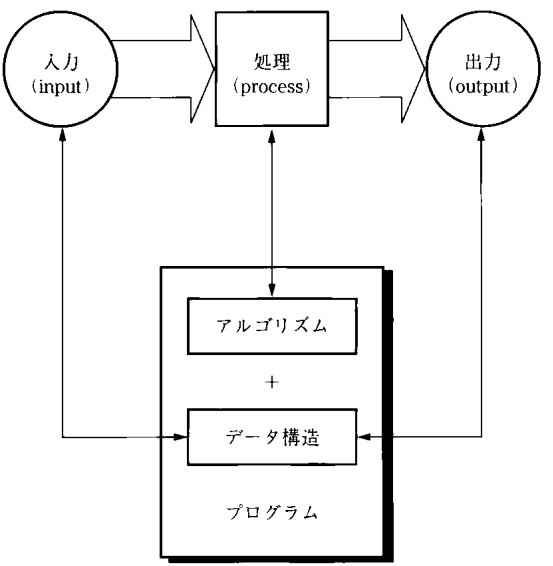


図1-2　アルゴリズムとデータ構造

此为试读，需要完整PDF请访问：www.ertongbook.com