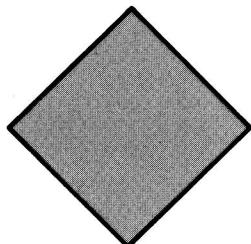


TP3
B2

7960119



Introduction to Computer Science



E7960119

P. M. Banks
J. R. Doupnik

Department of Applied Physics and Information Science
University of California, San Diego



John Wiley & Sons, Inc.
New York • London • Sydney • Toronto



Copyright © 1976, by John Wiley & Sons, Inc.

All rights reserved. Published simultaneously in Canada.

Reproduction or translation of any part of this work beyond that permitted by Sections 107 or 108 of the 1976 United States Copyright Act without the permission of the copyright owner is unlawful. Requests for permission or further information should be addressed to the Permissions Department, John Wiley & Sons, Inc.

Library of Congress Cataloging in Publication Data:

Banks, Peter M

Introduction to computer science.

Includes bibliographies and index.

1. Electronic digital computers—Programming.

I. Doupnik, J. R., 1938- joint author. II. Title.

QA76.6.B353 001.6'42 75-20407

ISBN 0-471-04710-4

Printed in the United States of America

10 9 8 7 6 5 4 3 2

Introduction to Computer Science

Preface

This book is directed towards first and second year college and university students of general background who are entering a first course in computers, computing, and computer science. For most students, and especially those in the Humanities, such a course represents an abrupt change in the pattern of their earlier studies, demanding a rapid integration of new ideas and techniques. Nevertheless, these students are keenly interested in computers as possible tools and as legitimate objects of social concern.

Our experience indicates that considerable care must be taken in choosing the curriculum of the introductory course. Students are strongly motivated by the “hands-on” approach of computer programming, often so much so that they neglect theoretical ideas relating to the development of computer science. One successful approach, adopted in this book, is to initially emphasize the problem-solving character of computers providing, through flowcharts and a flowchart language, a logical framework for reducing complex problems to algorithmic form. Later, as the students gain skill in a particular computing language taught in parallel with this material, more abstract concepts concerning computer organization and operation can be introduced.

Within the text, considerable emphasis has been placed upon the development of problem-solving skills. The problems at the end of each chapter

are appropriate for programming purposes and have been chosen to expand students' appreciation of the use of computers as tools to extend human capabilities of analysis. To a large extent, the problems are nonmathematical in character, relying as much as possible upon simple computations mixed with various levels of decision making.

In the initial chapter the historical development of computers and computing devices is traced in some detail. Such material represents our effort to provide humanities students with an abbreviated view of the way in which concepts of numbers and computation have developed. Although some may regard computer science as a discipline without historical perspective, the development of computers and their related internal organization provides an interesting view of the mutual dependence of developing technology and the philosophy of computation.

In the chapters immediately following the development of flowcharts and problem solving skills, material relating to the internal operation of computers is given, culminating in discussions of programs expressed in simple assembly language. Execution of assembly language problems on a computer is difficult at this level of development, so most problems were solved on paper. In teaching this material, it has been found useful to introduce *CARDIAC*, a cardboard illustrative aid to computation developed by the Bell Telephone Laboratories.¹ *CARDIAC* is simple in operation, yet allows students to practice many of the basic aspects of stored-instruction programs.

In the final two chapters, binary arithmetic and Boolean algebra are introduced and applied to the design of logic networks. In our own courses, these topics have met with considerable enthusiasm from students lacking previous mathematical training.

The order of presentation of topics in this book follows that of an introductory one quarter (10 week) course given at the University of California, San Diego. Lectures covering the first two chapters were given during the first week. During the subsequent 5 to 6 weeks the topics of Chapters Three and Four were presented in parallel with instruction in a specific programming language (ALGOL or FORTRAN in our case). During the final three to four weeks, the students continued to develop their programming skills through a variety of homework assignments, while emphasis in the main lectures shifted to topics taken from Chapters Five through Ten. At the end of this course, students could continue to expand their interests in digital computers through a subsequent 10 week course devoted to system programming with application to computer software and information-handling problems in the Humanities. A final 10 week course, aimed at students with some mathemati-

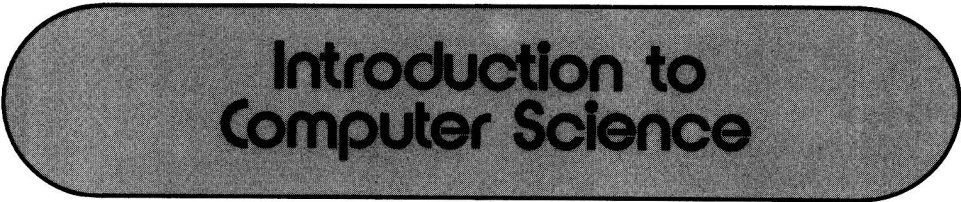
¹D. Hagelbarger and S. Fingerman, *CARDIAC*, A cardboard illustrative aid to computation, Bell System Educational Aid, Bell Telephone Laboratories, 1970.

cal training, gave an introduction to numerical algorithms, again with emphasis upon computer usage.

The level of the material presented here is appropriate for Humanities students in their first 2 years of college or university. More technically inclined students at 2 or 4 year colleges can be expected to make extensive use of the more mathematically oriented topics relating to Boolean algebra and the design of logic circuits. Although it has not been possible to eliminate all mathematical concepts, we have attempted to emphasize the non-numerical aspects of digital computers, keeping the level of instruction within reach of students possessing a normal high school mathematics background. Thus, simple computations and calculations remain as the conceptual mainstay of instruction in the basic computer languages and the presentation of this text.

We would like to acknowledge the assistance given us by the staff of the Department of Applied Physics and Information Science and express our thanks for valuable criticism from our students and various instructors, especially Dr. T. H. Hankins.

P. M. BANKS
J. R. DOUPNIK
La Jolla, California



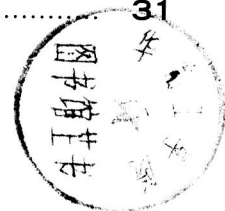
Introduction to Computer Science

Contents

one

Numbers, Calculations, and Modern Computers I

Counting and Calculations	2
Finger Counting	3
Counting Boards	6
The Abacus	11
The Triumph of the Algorithmicists	11
New Tools	14
But What About Computers?	22
References to Further Reading	30
Exercises	31



two

What is a Digital Computer? 33

The Organized Machine	34
The Language of the Computer	39
Algorithms and Programs	43
Programs: A Biographical Sketch	46
Why Digital?	53
Digital Devices	55
A Second Type of Computer	57
References to Further Reading	59
Key Words and Phrases to Know	60
Exercises	60
Problems	61

three

Computers. Algorithms. and Flowcharts 63

The Role of Computers	64
Problem Solving and Computer Applications	64
Problems and Algorithms	65
Algorithm Design and Flowcharts	68
Algorithms, Flowcharts, and Computers	78
Variables	79
Replacement of Values	81
Expressions	84
Arithmetic Expressions	84
More About Arithmetic Variables	89
Expressions with Integer and Real Numbers	90
Alphanumeric Expressions	93
Computer Oriented Flowcharts: The First Steps	97
Temperature Conversion	97
Correcting A String	98
The Uses of Computers	100
References to Further Reading	101
Key Words and Phrases to Know	102
Exercises	102
Problems	104

four

More About Flowcharts and Algorithms 107

More About Flowcharts	108
Process Boxes	108
Decision Boxes	109
Input and Output Boxes	112
Cautionary Comments	114
The Concordance Problem	115
Counters and Loops	120
Finding the Largest Number in a List	124
Roots of Numbers	124
Compound Relational Expressions	130
A Grading Algorithm	134
Subscripted Variables	134
Revised Largest Number Algorithm	139
An Algorithm for Sorting	142
Brute Force Sort	145
The Bubble Sort	148
Multiple Subscripts	152
SubAlgorithms	153
References to Further Reading	157
Key Words and Phrases to Know	157
Exercises	158
Problems	160

five

Numbers for Computers 179

Why Numbers?	180
Numbers from Past to Present	180
Number Systems for Computers	184
Place-Value Notation	185
Binary to Decimal Number Conversion	190
Decimal to Binary Number Conversion	190
Interconversion of Binary, Octal and Hexadecimal Numbers	194
References to Further Reading	195
Key Words and Phrases to Know	196
Exercises	196
Problems	198

six

How a Computer Stores Information 199

Internal Representation of Information	200
Bits, Bytes, and Words	206
Storage of Alphabetic Data	209
Storage of Numerical Data	211
The Fixed Point System	212
The Binary Coded Decimal System	214
Floating Point Numbers	215
Rounding Errors	220
The Storage of Logical Data	224
References to Further Reading	227
Key Words and Phrases to Know	227
Exercises	228

seven

Computer Instructions 231

More About the Internal Organization of Computers ...	232
What is an Instruction?	238
Three Address Instructions	240
Two Address Instructions	242
One Address Instructions	244
Zero Address Instructions	247
What Can be Done with Instructions?	251
Data Modification Instructions	253
Data Transfer Instructions	259
Programs and Instructions	260
References to Further Reading	260
Key Words and Phrases to Know	261
Exercises	261

eight

Putting Instructions to Work 263

Building a Program	264
Indexing and Index Registers	271
Immediate Type Instructions	274
Indirect Addresses	277
Subprograms and the Branch and Link Instruction	279
A Two Argument Subprogram Instruction List	284
Compiling, Loading, and Execution	286
Synopsis of Modified One Address Instructions	287
References to Further Reading	288
Key Words and Phrases to Know	288
Exercises	289
Problems	290

nine

Binary Arithmetic for Computers 293

The Need for Binary Arithmetic	294
Binary Addition	295
Binary Subtraction	296
Binary Integers and Complements	297
On to Binary Subtraction	300
Examples of Binary Subtraction with Complements	301
Binary Multiplication	305
Examples of Multiplication with Positive Numbers	305
Binary Division	306
References to Further Reading	310
Key Words and Phrases to Know	311
Exercises	311
Problems	312

ten

Boolean Algebra 313

Algebras and Reality	314
The Ideas of Boolean Algebra	317
The Formalities of Boolean Algebra	319
The Algebra of Gates and Switches	325
Logical Design	330
Practical Application of Logic Design	336
References to Further Reading	341
Key Words and Phrases to Know	341
Exercises	342
Problems	344

Appendices 347

I One System of Card Punch Codes	348
II Standard Computer Character Codes	349
III More About Number Conversions	350

Index 357

one

Numbers. Calculations. and Modern Computers

Counting and Calculations
Finger Counting
Counting Boards
The Abacus
The Triumph of the Algorithmicists
New Tools
But What About Computers?
References to Further Reading
Exercises

COUNTING AND CALCULATIONS

Numerals, numbers, and arithmetic.

Widespread knowledge of simple calculation makes it easy for us to forget that common use of Arabic numerals and memorized steps of addition, subtraction, multiplication, and division only began in the Middle Ages. Prior to the sixteenth century, numbers in the Western civilizations were written using Roman, Hebrew, Greek, or similar numerals in ways quite different from today's system. (See Figure 1.1.) These older forms of number expression generally lacked the ease of modern written computations and it is not surprising that alternative methods of calculation were developed. These methods, now mostly forgotten in our technology of slide rules, calculators, and digital computers, were based on sophisticated forms of finger counting and mechanical aids such as the ancient counting table and the abacus.

This text is concerned with the ideas surrounding modern digital computers. These machines, combining advanced electronic technology with an internal structure based on the principles of mathematical logic, are a singular invention of our age. Nevertheless, the use of mechanical aids for solving computational problems is not a development unique to the twentieth century. Long before mankind guessed at the laws of electricity or developed its present skill in building machines, repeated attempts were made to ease the tiresome burden of computations arising from the exchange of monies, trading of goods and the complexities of government finance and record keeping. As we shall see, these attempts were at first satisfied by the use of hand and finger computations, followed later by the development of mechanical computing aids in the form of the early Greek and Roman counting boards. From these beginnings a chain of successive computational tools can be traced through the later forms of the abacus; the seventeenth century inventions of calculating rods and the slide rule; the later ingenious mechanical computing machines of Pascal, Leibniz and others; the analytical engines of Babbage; and on to the use of electronic devices in the 1930s.

$$\begin{array}{r} \text{CCCXXV} \cdot \text{XLVII} \\ \hline \text{MMCCLXXV} \\ \text{((I)) MMM} \\ \hline \text{((I)) I) CCLXXV} \end{array}$$

FIGURE 1.1. Multiplication with old Roman numerals.

The modern digital computer has emerged from these extensive foundations as a synthesis of mechanical, mathematical, and electrical inventions. It is now generally recognized that a computer is a machine which not only works with numbers but, even more importantly, acts as a device which can store and process nonnumerical information. Examples of computer uses are found in language translation programs, airline reservation systems, air traffic control networks, newspaper composition, medical diagnosis and control, and a myriad of other applications.

It is in the area of data processing and decision making that computers most directly affect us. The complexities of our technical civilization continually demand choices among alternatives and accountings based upon vast amounts of information which, in turn, requires special storage and rapid processing. Within this context the term *digital computer* is misleading, since the older ideas of computation and number processing have been outgrown and modern general computers are designed to deal equally well with numerical and nonnumerical data.

To gain a better perspective of the roles played by digital computers, it is useful to first emphasize their numerical character while examining the ways counting and simple calculations were done in the past. Such an approach leads in a natural manner through the development of scientific computation into the much broader areas of electronic data processing and the way computers are presently used to manipulate general forms of information.

As we know, our modern concepts of numerals, numbers, and computations were not created at their present level of sophistication, but rather matured slowly during several thousands of years in many different civilizations before emerging as a remarkably outstanding achievement of human intellect. In tracing the development of number symbols and the way numbers were constructed in the past, there occurs an inevitable intertwining of the methods of number expression with the basic procedures of arithmetic calculation. For the present, it is sufficient to focus our attention first upon the way simple calculations were made before the age of the modern digital computer. The parallel discussion of the development of numbers and number expressions will be deferred until Chapter five when a more complex treatment of computer data storage can be given.

FINGER COUNTING

While the number symbols of most ancient civilizations are known from various preserved writings, the different ways these people actually made their calculations remain hidden. Even in the case of Egypt during the age of the Pharaohs, the large number of papyri dealing with agriculture, mathe-