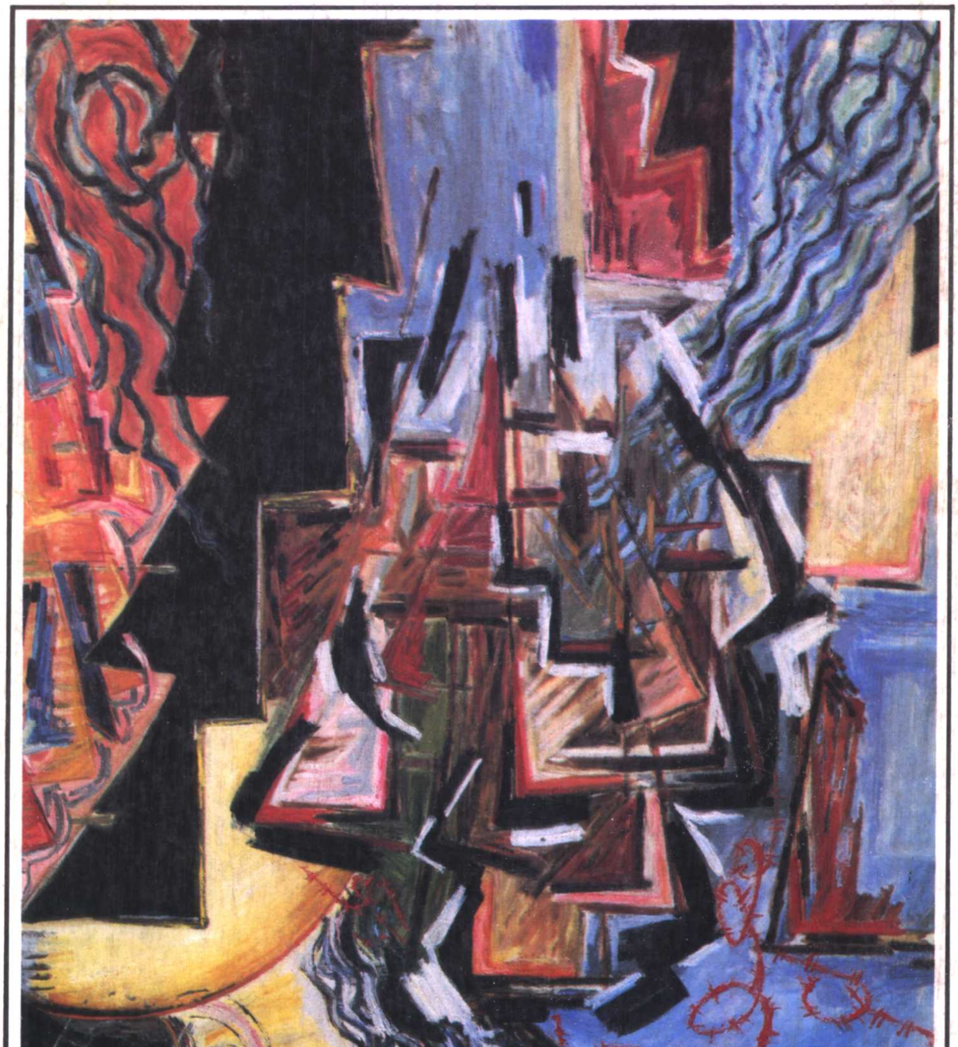
PASCAL

SECOND EDITION

Programming and Problem Solving

SANFORD LEESTMA/LARRY NYHOFF



SANFORD LEESTMA
LARRY NYHOFF
Department of Mathematics and Computer Science
Calvin College

PASCAL Programming and Problem Solving

SECOND EDITION

Macmillan Publishing Company
NEW YORK

Collier Macmillan Publishers
LONDON

Copyright © 1987, Macmillan Publishing Company, a division of Macmillan, Inc.

Printed in the United States of America

All rights reserved No part of this book may be reproduced or transmitted in any form or by any means electronic or mechanical, including photocopying recording or any information storage and retrieval system without permission in writing from the publisher.

Earlier edition copyright © 1984 by Macmillan Publishing Company.

Macmillan Publishing Company
866 Third Avenue New York, New York 10022

Collier Macmillan Canada Inc

Library of Congress Cataloging in Publication Data

Leestma Sanford
  Pascal programming and problem solving

  Includes index.
  1. PASCAL (Computer program language) I. Nyhoff, Larry R. II. Title.
QA76.73.P2L44 1987 005.13'3 86-21692
ISBN 0-02-369690-7

Printing: 6 7 8 Year: 9 0 1 2 3 4 5

ISBN 0-02-369690-7

PREFACE

Pascal was developed in the late 1960s and early 1970s by Niklaus Wirth, a Swiss computer scientist at the Eidgenössische Technishe Hochshule (ETH) in Zurich, Switzerland. His primary goal was to develop a language that makes it possible "to teach programming as a systematic discipline based on certain fundamental concepts clearly and naturally reflected by the language." *The Pascal User Manual and Report,* written by Wirth and K. Jensen and published in 1974, serves as the basic definition of the Pascal language. As the use of Pascal grew, some differences appeared in various implementations. To ensure that Pascal programs written on one system can be executed on another, national and international standards for the language have been formulated. A recent standard is *An American National Standard IEEE Standard Pascal Computer Programming Language,* which was published in 1983 by Institute of Electrical and Electronics Engineers (IEEE). This standard was approved by the IEEE Standards Board and the American National Standards Institute (ANSI) and serves as the basis for this text. Differences between this standard and other popular versions of Pascal are described in brief sections at the ends of appropriate chapters and in Appendix G.

This text is a complete introduction to the Pascal programming language and is designed to meet the objectives of the course CS1: *Introduction to Programming Methodology* as described in the Curriculum '84 recommendations of the Association of Computing Machinery (ACM). The course objectives as described in this curriculum guideline are (cf. "Recommended Curriculum for CS1, 1984," *Communications of the ACM,* October, 1984):

- to introduce a disciplined approach to problem-solving methods and algorithm development;
- to introduce procedural and data abstraction;
- to teach program design, coding, debugging, testing, and documentation using good programming style;
- to teach a block-structured high-level programming language;
- to provide a familiarity with the evolution of computer hardware and software technology;
- to provide a foundation for further studies in computer science.

To meet these objectives, this text emphasizes problem solving using structured algorithm and program design throughout and illustrates these with a large number of complete examples and applications, many of which include algorithms given in pseudocode and/or structure diagrams, complete Pascal programs implementing the algorithms, and sample runs. Each of the examples is intended to demonstrate good algorithm design and programming style. At the end of each chapter a Programming Pointers section summarizes the main points regarding structure and style as well as language features presented and some problems that beginning programmers may experience. In addition to presenting the basic features of Pascal, the text introduces such topics as data representation, machine language, and compilers, and the last two chapters introduce some elementary data structures, such as stacks, queues, linked lists, and trees.

Like the first edition, this text is intended for an introductory computer programming course using Pascal. Thus, the level is the same as that of the first edition, with the intended audience being freshman and/or sophomore college students whose mathematical background includes high school algebra. The first edition was written for the course CS1 as described in the Curriculum '78 guidelines of the ACM. A new course description appeared in 1984, however, and the new text has been carefully revised to implement these recommendations. We have also benefited from constructive comments of instructors and students who used the first edition and we have incorporated many of these suggestions in this second edition. Specifically, the changes in the new edition include the following:

- Procedures are introduced earlier in the text (before functions). The discussion of scope rules is simplified, and more difficult and less frequently used material has been relegated to an appendix. The introduction to recursion is expanded and improved.
- More examples and exercises from nonmathematical applications have been included.
- Material in Chapter 4 has been reorganized so that selection structures are considered before repetition structures; in the case of the latter, **while** and **repeat** statements are considered before the **for** statement.
- Introduction to data structure design and algorithm analysis (Chapters 13 and 14) has been improved.
- Presentation of structured data types has been improved with a new introduction to and better examples of multidimensional arrays in Chapter 9, consideration of records before sets in Chapters 10 and 11, and a simplified treatment of files in Chapter 12.
- The discussion of problem solving and algorithm development in Chapter 2 has been expanded, and additional material on testing and debugging has been included in Chapter 4. More structure diagrams have been used to display the structure of more complicated programs.
- The discussion of data representation and other basic concepts of computer systems (including machine language, assembly language, and compilers) in Chapter 1 has been expanded so that it conforms to the CS1 guidelines.
- A second color has been used to highlight important concepts and to improve readability.
- Brief sections describing common variations of and extensions to standard Pascal have been added at the ends of chapters where appropriate.

此为试读，需要完整PDF请访问：www.ertongbook.com

- A new appendix details these variations and extensions as implemented in Macintosh™ Pascal Turbo Pascal™, and UCSD™ Pascal.

Revised and expanded supplementary materials available from the publisher include the following:

- An instructor's manual containing lecture notes, solutions to exercises, sample test questions, and transparency masters.
- Data disks containing all the sample programs and data files used in the text. Standard Pascal and Turbo Pascal versions are available.
- A test bank and test generation software for microcomputers.

Acknowledgments

We express our appreciation to all those who were involved in the preparation of this text: to our colleagues who have used this material in their classes and whose suggestions have strengthened the presentation; to our students, who have served as test subjects; to David Johnstone, Ron Harris, and all other Macmillan personnel who initiated, supervised, produced, or in some other way contributed to the finished product; to the several reviewers of the manuscript, whose comments were encouraging, helpful, and sincerely appreciated; and to Marge, Michelle, Sandy, Michael, Shar, Jeff, Dawn, Jim, Julie, and Joan, whose patience, love, support, and understanding during the preparation of this text and others has exceeded what we have any right to expect.

S.C.L.
L.R.N.

CONTENTS

此为试读,需要完整PDF请访问: www.ertongbook.com

1 Introduction and History

I wish these calculations had been executed by steam.

Charles Babbage

For, contrary to the unreasoned opinion of the ignorant, the choice of a system of numeration is a mere matter of convention.

Blaise Pascal

The modern electronic computer is one of the most important products of the twentieth century. It is an essential tool in many areas, including business, industry, government, science, and education; indeed, it has touched nearly every aspect of our lives. The impact of this twentieth-century information revolution brought about by the development of high-speed computing systems has been nearly as widespread as the impact of the nineteenth-century industrial revolution. This chapter gives a summary of the history of computer systems and briefly describes their components.

1.1 History of Computing Systems

There are two important concepts in the history of computation: the ***mechanization of arithmetic*** and the concept of a ***stored program*** for the automatic control of computations. We shall focus our attention on some of the devices that have implemented these concepts.

A variety of computational devices were used in ancient civilizations. One of the earliest, which might be considered a forerunner of the modern computer, is the ***abacus*** (Figure 1.1), which has movable beads strung on rods to count and make computations. Although its exact origin is unknown, the abacus was used by the Chinese perhaps three to four thousand years ago and is still used today.

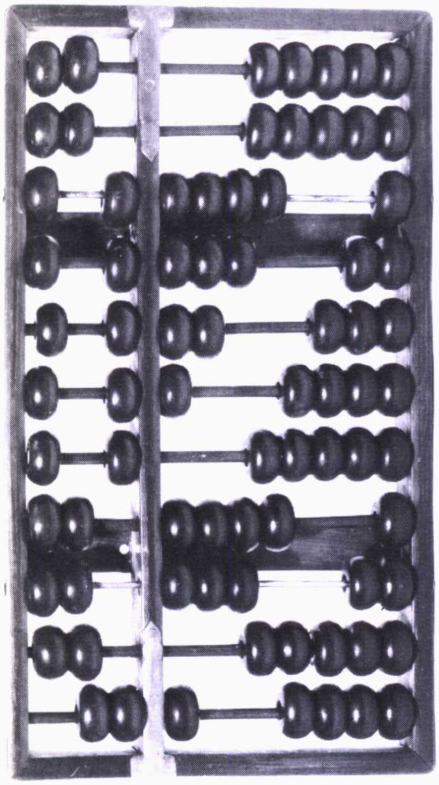


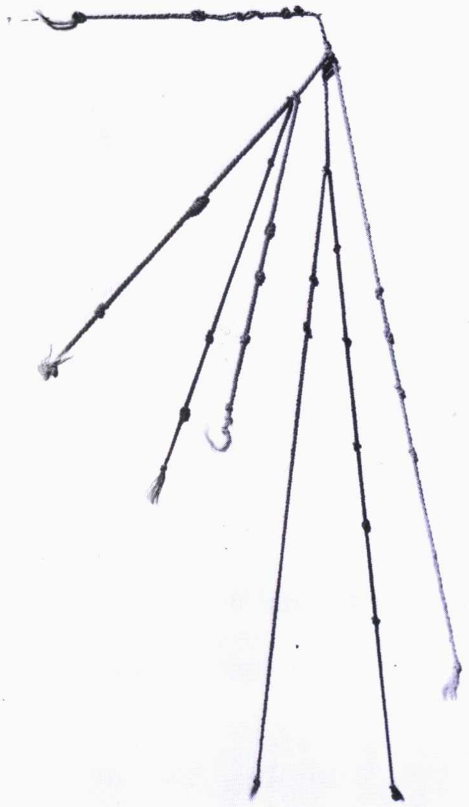
Figure 1.1
Abacus.

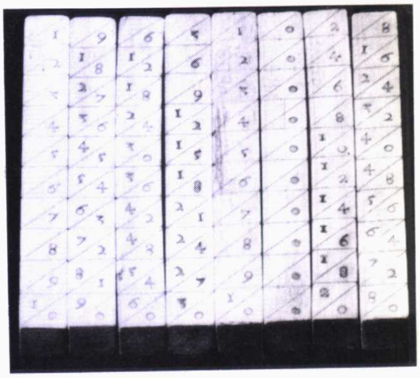
The ancient British stone monument ***Stonehenge*** (Figure 1.2a), located in southern England, was built between 1900 and 1600 B.C. and evidently was an astronomical calculator used to predict the changes of the seasons. Five hundred years ago, the Inca Indians of South America used a system of knotted cords called ***quipus*** (Figure 1.2b) to count and record divisions of land among the various tribal groups. In Western Europe, ***Napier's bones*** (Figure 1.2c) and tables of ***logarithms*** were designed by the Scottish mathematician John Napier (1550–1617) to simplify calculations. These led to the subsequent invention of the ***slide rule*** (Figure 1.2d).

In 1642, the young French mathematician ***Blaise Pascal*** (1623–1662) invented one of the first mechanical adding machines (Figure 1.3). This device used a system of gears and wheels similar to that used in odometers and other modern counting devices. ***Pascal's adder*** could both add and subtract, and was invented to calculate taxes. Pascal's announcement of his invention reveals the labor-saving motivation for its development:

> Dear reader, this notice will serve to inform you that I submit to the public a small machine of my invention, by means of which you alone may, without any effort, perform all the operations of arithmetic, and may be relieved of the work which has often times fatigued your spirit, when you have worked with the counters or with the pen. As for simplicity of movement of the operations, I have so devised it that, although the operations of arithmetic are in a way opposed the one to the other—as addition to subtraction, and multiplication to division—nevertheless they are all performed on this ma-

(a)


(b)


(c)

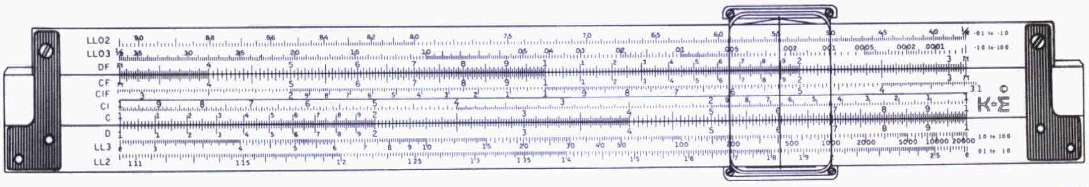

(d)

Figure 1.2. (a) Stonehenge. (b) Quipus (Courtesy of the American Museum of Natural History). (c) Napier's bones (Courtesy of the Smithsonian Institution). (d) Slide rule.

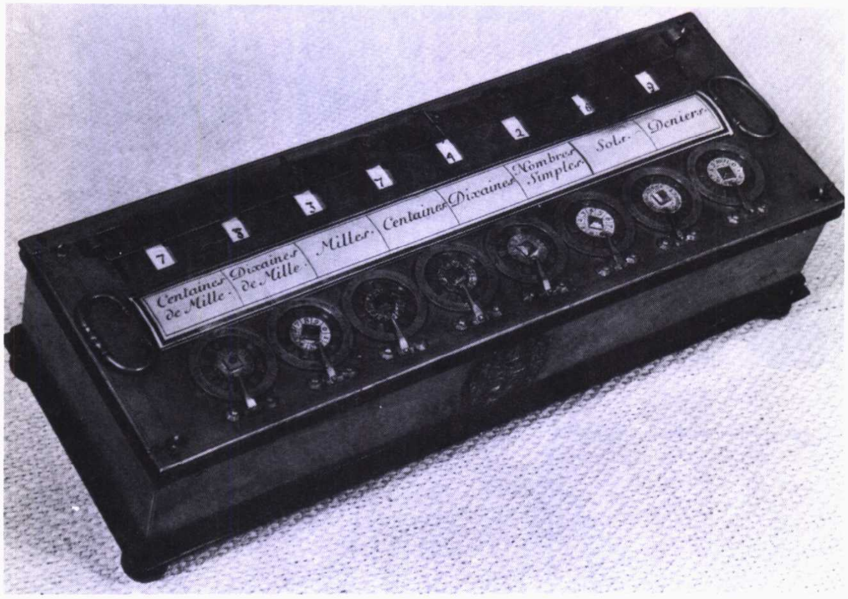


Figure 1.3. Pascal's adder. (Courtesy of IBM.)

> chine by a single movement. The facility of this movement of operation is very evident since it is just as easy to move one thousand or ten thousand dials, all at one time, if one desires to make a single dial move, although all accomplish the movement perfectly. The most ignorant find as many advantages as the most experienced. The instrument makes up for ignorance and for lack of practice, and even without any effort of the operator, it makes possible shortcuts by itself, whenever the numbers are set down.

Although Pascal built more than fifty of his adding machines, his commercial venture failed because the devices could not be built with sufficient precision for practical use.

In the 1670s, the German mathematician ***Gottfried Wilhelm von Leibniz*** (1646–1716) produced a machine that was similar in design to Pascal's, but somewhat more reliable and accurate (Figure 1.4). Leibniz's calculator could

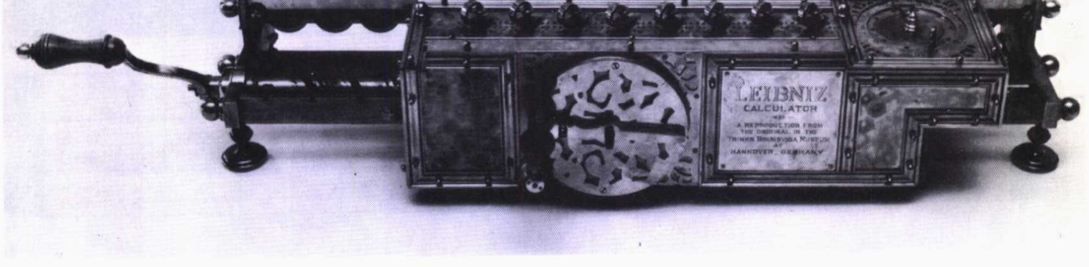


Figure 1.4. Leibniz's calculator. (Courtesy of IBM.)

perform all four of the basic arithmetic operations: addition, subtraction, multiplication, and division.

A number of other mechanical calculators followed that further refined the designs of Pascal and Leibniz. By the end of the nineteenth century, these calculators had become important tools in science, business, and commerce.

As noted earlier, the second idea to emerge in the history of computing was the concept of a stored program to control the calculations. One early example of an automatically controlled device is the weaving loom (Figure 1.5) invented by the Frenchman ***Joseph Marie Jacquard*** (1752–1834). This automatic loom, introduced at a Paris exhibition in 1801, used metal cards punched with holes to position threads for the weaving process. A collection of these cards made up a program that directed the loom. Within a decade, 11,000 of these machines were in use in French textile plants, resulting in what may have been the first incidence of unemployment caused by automation. Unemployed workers rioted and destroyed several of the new looms and cards. Jacquard wrote: ''The iron was sold for iron, the wood for wood, and I its inventor delivered up to public ignominy.'' The ***Jacquard loom*** is still used today, although modern versions are controlled by magnetic tape rather than punched cards.

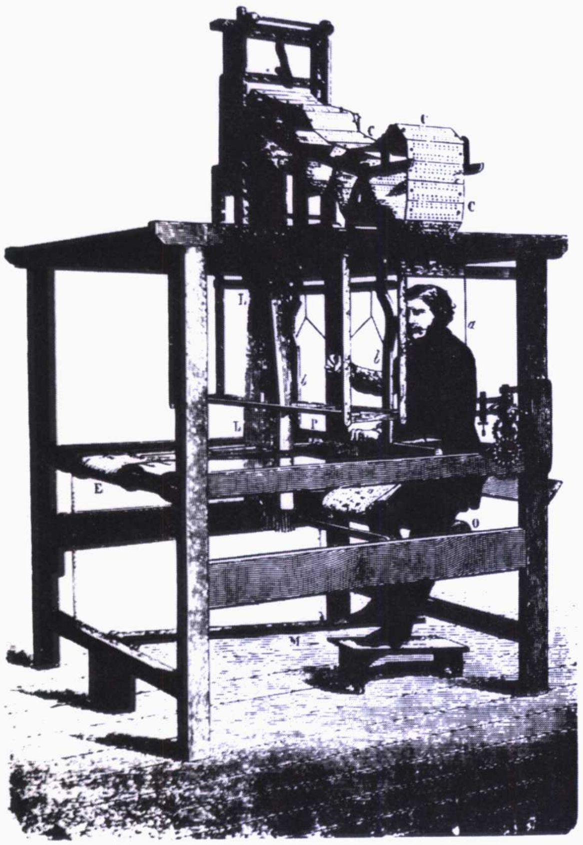
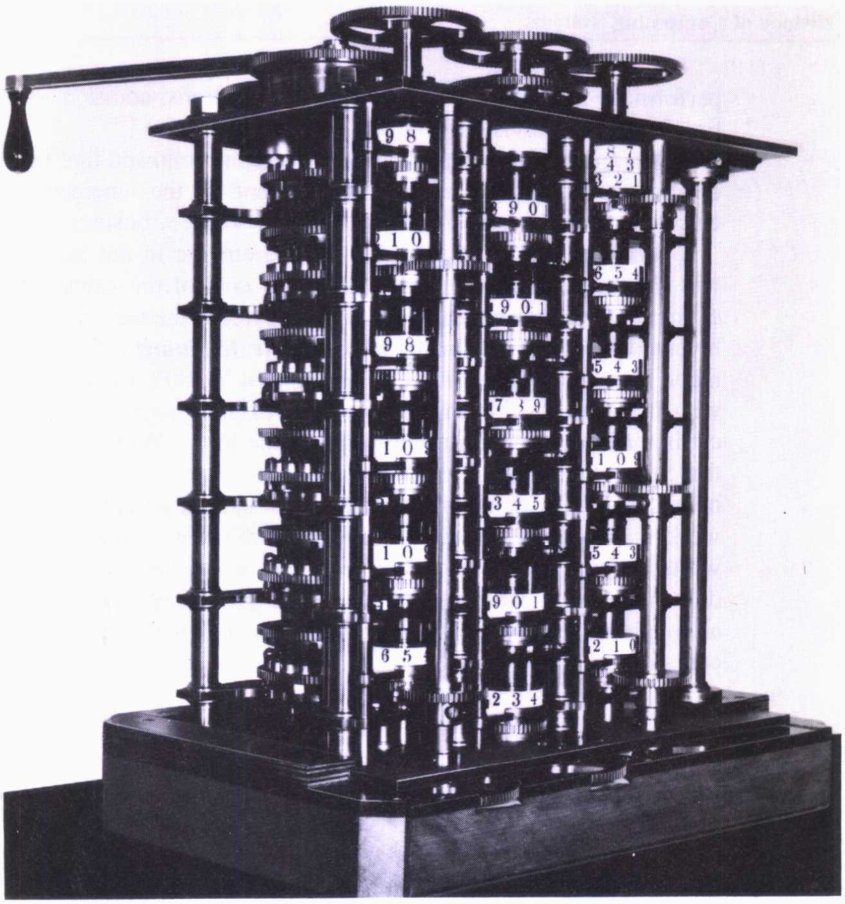


Figure 1.5.
Jacquard loom. (Courtesy of IBM.)

(a)



(b)

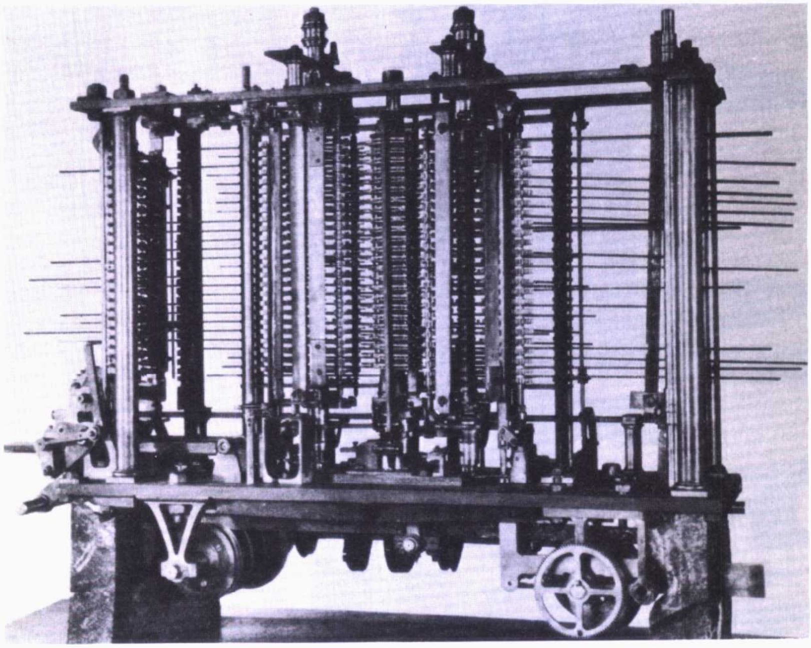


Figure 1.6. (a) Babbage's Difference Engine. (b) Babbage's Analytical Engine. (Courtesy of IBM.)

此为试读，需要完整PDF请访问：www.ertongbook.com