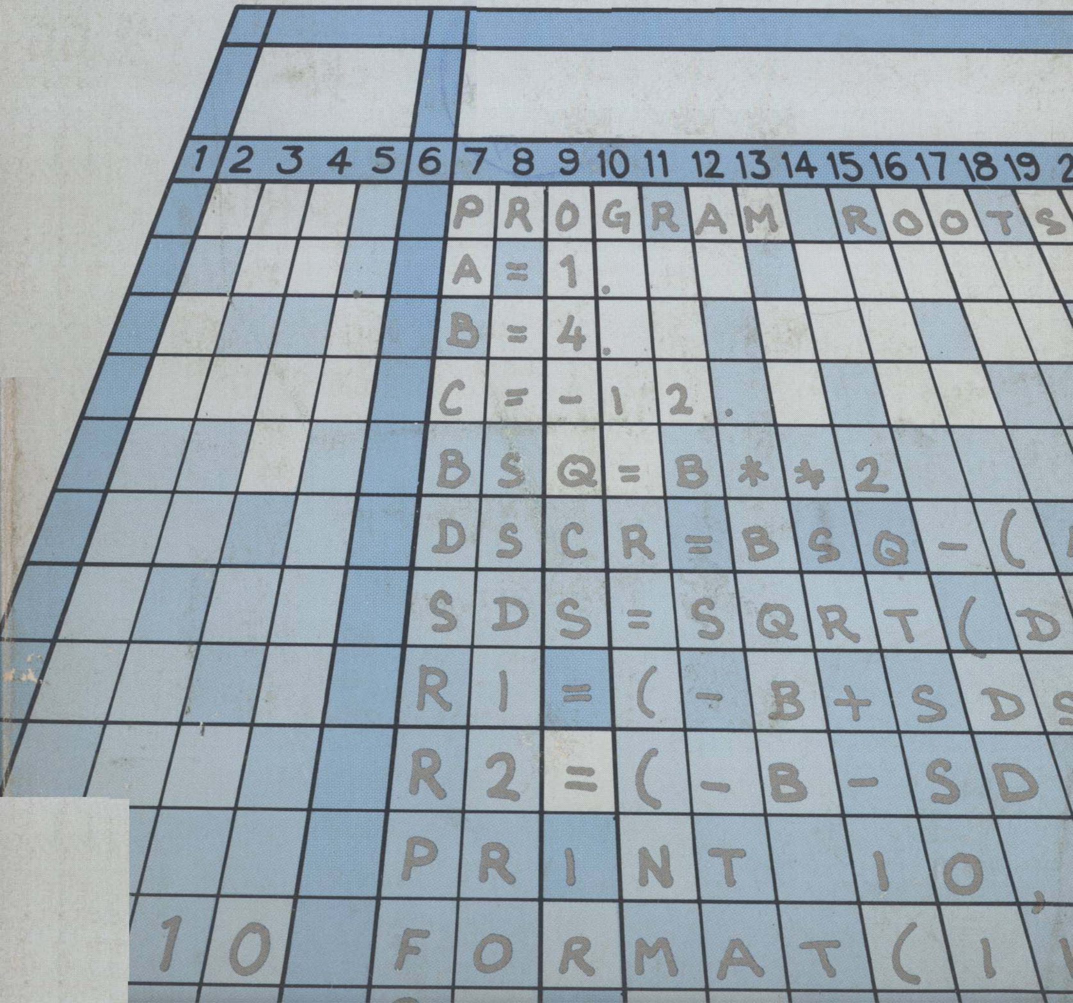
BEGINNING FORTRAN

Joe W. McKinley

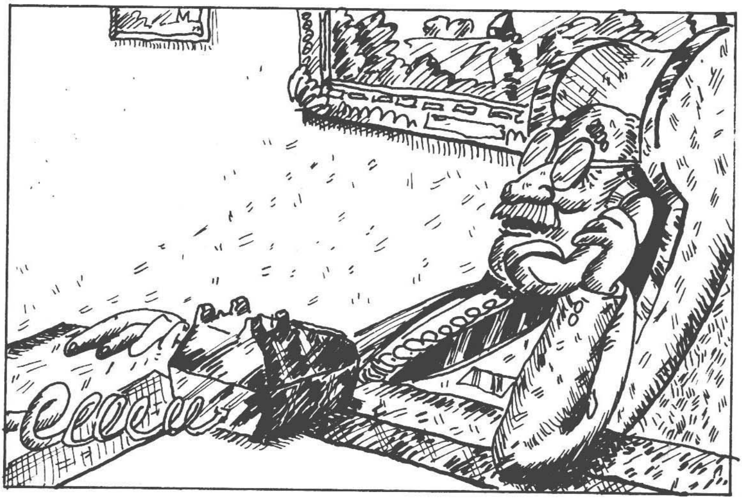


```
      PROGRAM ROOTS
      A=1.
      B=4.
      C=-12.
      BSQ=B**2
      DSCR=BSQ-(
      SDS=SQRT(D
      R1=(-B+SDS
      R2=(-B-SD
      PRINT 10,
 10   FORMAT(1
```

Beginning FORTRAN

Joe W. McKinley
Professor
Department of Aerospace Engineering
California State Polytechnic University, Pomona

Matrix Publishers, Inc.
Beaverton, Oregon • USA



DEDICATION

This book is dedicated to the memory of Barbara Ann Edwards.

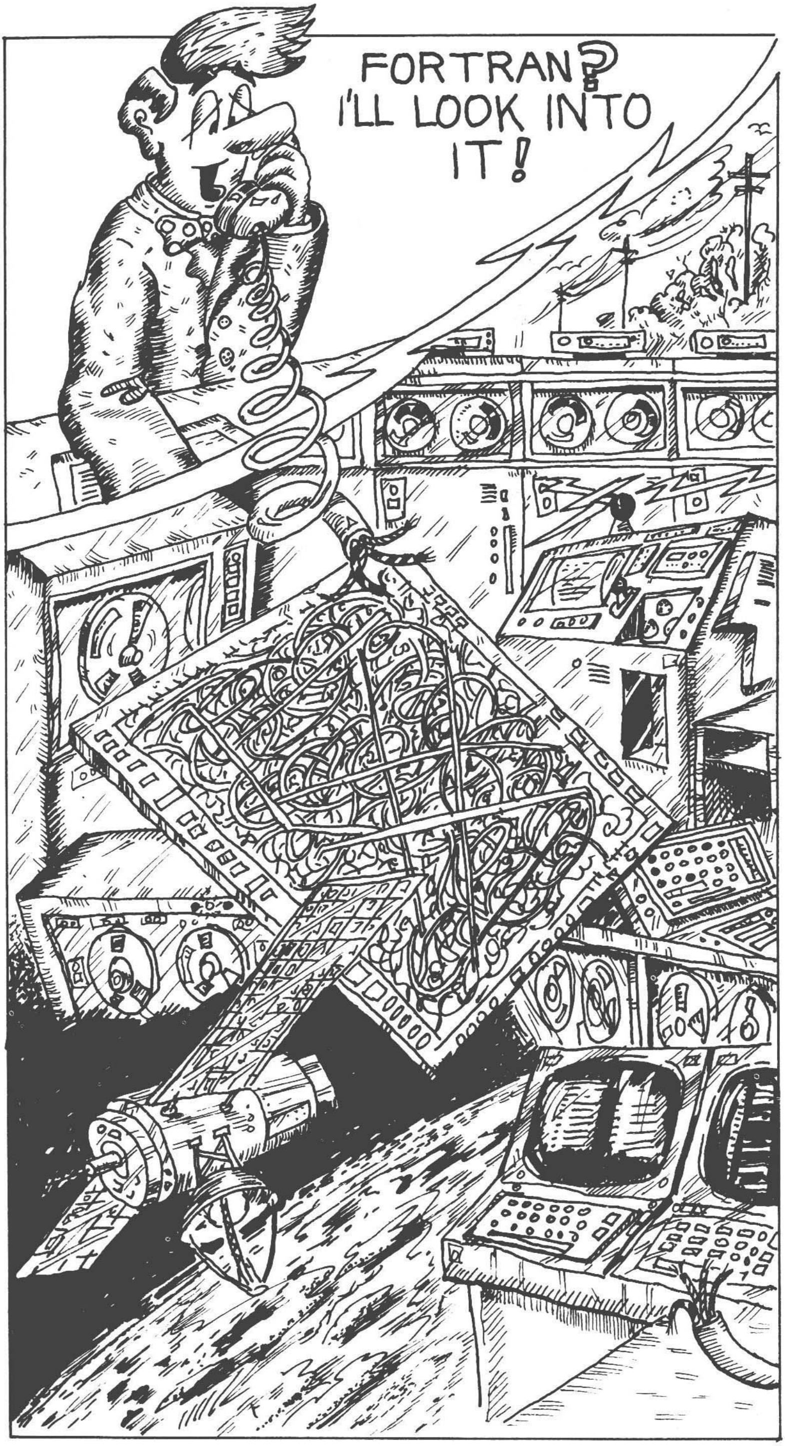
10 9 8 7 6 5 4 3

ISBN: 0-916460-11-8
Library of Congress catalog card number: 80-81871

Production: Patricia Miller
Illustrations: Scientific Illustrators
Printing: Pantagraph Printing
Editor: Merl K. Miller

Matrix Publishers, Inc.
11000 S.W. 11th Street
Beaverton, Oregon 97005

Beginning FORTRAN


FORTRAN? I'LL LOOK INTO IT!

Preface

This book has been generated out of the author's experience of teaching FORTRAN to students at the beginning level in college. It is intended for students in high schools, community colleges, colleges and universities who are beginning to learn the language of FORTRAN.

This book is primarily written about a batch processing mode of operation; however, people who are learning FORTRAN in an interactive or time-share mode will find the book extremely useful in conjunction with the local rules for operating their system.

No book can cover everything or be completely self contained. The book is aimed at the novice. Its goal is to give him, in a streamlined, relevant fashion, the material which is necessary for him to begin programming, and to help him to assimilate the skills needed to grow on his own. Upon the completion of the first two sections of this book, this goal should have been accomplished and the base for advanced study should be established. At this time the student can pursue the projects in the back of the book and the suggested references so as to proceed in his own way to the mastery of FORTRAN which he or she desires.

A key feature of the book is a streamlined, no nonsense approach whereby the student participates in the beginning and intermediate phases of programming directly to the use of the advanced considerations of the computer example projects. There is much emphasis on learning by doing through the execution of simple programs. There are also end of chapter problems and it is recommended that all of these be done by the student. The material of sections I and II form a base whereby the student may go on into specialization in advanced level programming in any chosen field. Simple problems and projects form the basis for learning in Chapters 1 through 6 and it is in these chapters that the base is established. Then examples are given from a wide variety of engineering, scientific, mathematical, business and social science areas, as well as general numerical analysis and computer utilization, so that one may move into the study of his or her particular area of interest. Material in related field such as key punching, flow charting, and matrix algebra is given in the appendices.

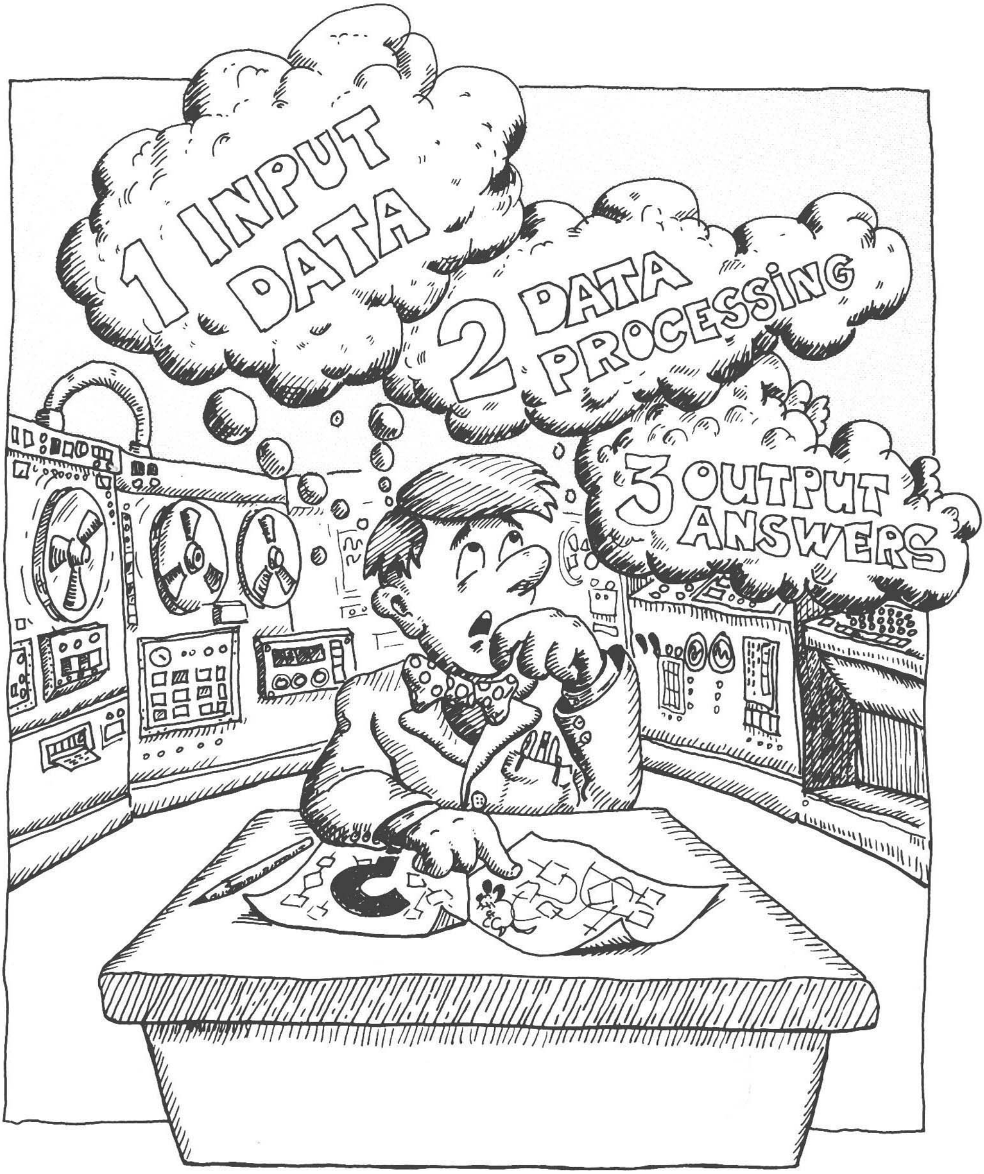
The author would like to express his appreciation to the many students who have inspired this book. The help of Doris Jane McKinley in the course of the author's career is also appreciated as is that of his children, Frank and Celia. In particular, the invaluable assistance of Glenda S. Menges as production assistant and typist is appreciated. Many of the programs were written and run by Mr. Dan Newell. Several other Cal Poly students who contributed to the programming effort are: Richard Miranda, Jim Eshleman, Michael McMaster, and Julie Gardner. The editorial assistance of Prof. Paul Chirlian is gratefully acknowledged. The author would like to express further appreciation to California State Polytechnic University, Pomona, and especially to the Computer Center.

Joe W. McKinley
Pomona, California
January, 1980

Contents

Section I

Beginning Programming



Chapter 1

Introduction: An Overview

This text is written to teach people to utilize the power of a computer in a specific way, namely writing a FORTRAN program to solve a problem in a batch processing environment. The text will also be extremely useful to those people who are using an interactive or time share mode, if it is used in conjunction with the execution instructions of the local facility. FORTRAN is a particular computer language of wide usage. There are, to be sure, other languages – BASIC, COBOL and ALGOL, to name a few – but FORTRAN ranks very high in popularity.

A batch processing environment means that the user will punch cards and leave them at the computer center. After a period of time, his punched cards will be returned to him along with a listing of his program and a listing of his answers. The time which elapses between the programmer's leaving his deck and the answers' being available for pickup is called the *turn-around time* of the computer center operation.

Let us imagine a programmer who wants to solve a simple example problem by using a computer. The programmer wants to solve for the roots of

$$ax^2 + bx + c = 0 \tag{1.1}$$

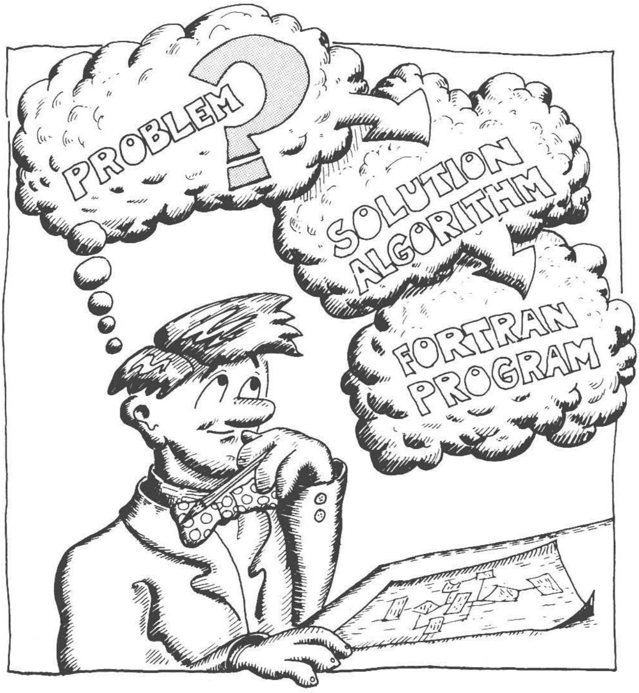
for some particular values of a, b, and c, for example,

$$\begin{aligned} a &= 1 \\ b &= 2 \\ c &= -12 \end{aligned}$$

Our programmer knows that the roots of this polynomial are given by

$$x_{1,2} = -b \pm \sqrt{b^2 - 4ac}/2a \tag{1.2}$$

subject to some restrictions, which we shall ignore for the present time. Equation (1.2) is referred to as an *algorithm*. An algorithm is a computational procedure for getting a numerical answer. The programmer is now in the position of being able to write a program to solve for the answers he wants as depicted in Fig. 1.1.



A handwritten listing of such a program is shown in Fig. 1.2. Do not be concerned about all of the unknown details about this program. They will be explained soon. For now let us say that this is a short, simplified program, which can be improved as we continue to learn to talk to the computer in FORTRAN.

The programmer now punches this program on cards. The punched cards are shown in Fig. 1.3*. Again, we need to defer our complete understanding of these cards until later. A few other cards, called control cards, are now added to the FORTRAN deck. A schematic of

*See Appendix for instructions on keypunching.

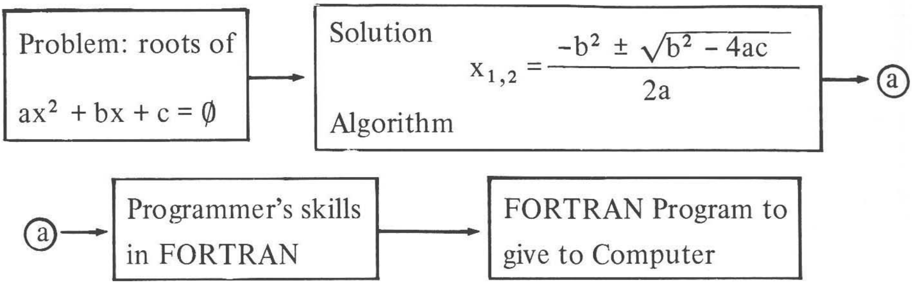
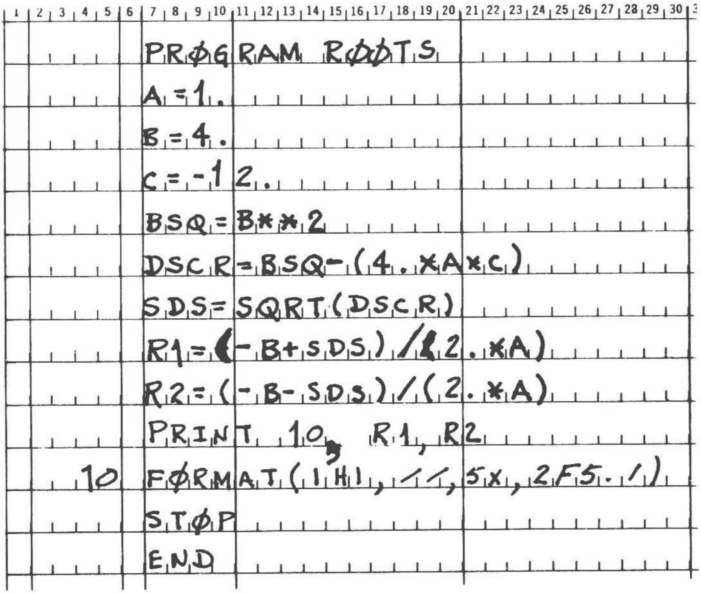

Problem: roots of
$ax^2 + bx + c = \emptyset$
Solution
$x_{1,2} = \frac{-b^2 \pm \sqrt{b^2 - 4ac}}{2a}$
Algorithm
a
a
Programmer's skills in FORTRAN
FORTRAN Program to give to Computer

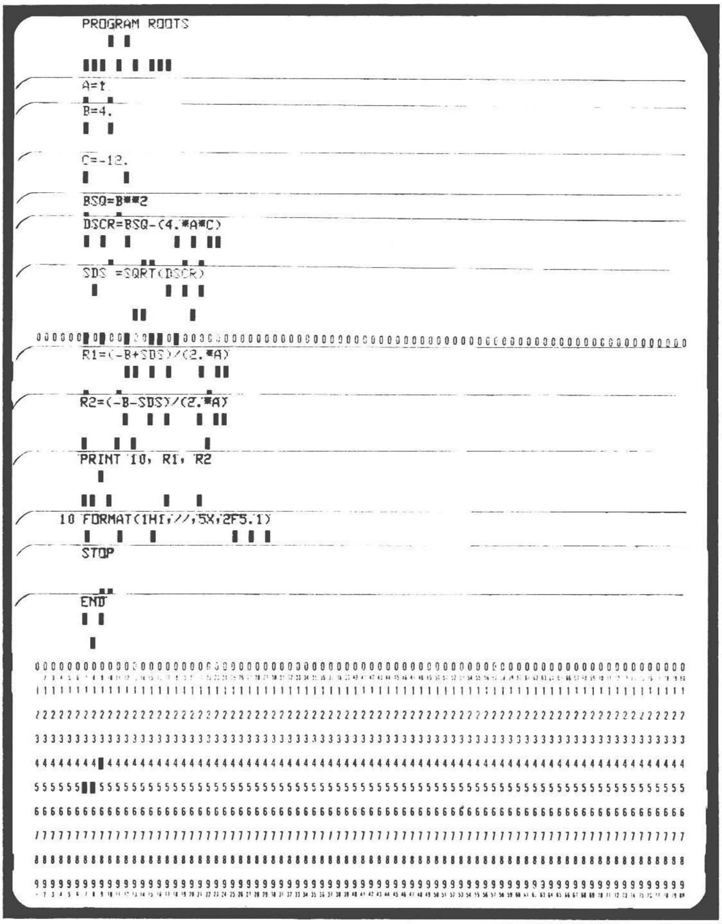
Figure 1.1 The programmer who knows the problem, and *how* to solve the problem, writes a FORTRAN program so that the computer will do the work.



```
      PRØGRAM RØØTS
      A=1.
      B=4.
      C=-12.
      BSQ=B**2
      DSCR=BSQ-(4.*A*C)
      SDS=SQRT(DSCR)
      R1=(-B+SDS)/(2.*A)
      R2=(-B-SDS)/(2.*A)
      PRINT 10, R1, R2
   10 FØRMAT(1H1,//,5X,2F5.1)
      STØP
      END
```

Figure 1.2 The handwritten listing of the programmer's FORTRAN program

his deck setup is shown in Fig. 1.4 and a sample deck is shown in Fig. 1.5. *Note that control cards are different for different computer centers!* The programmer then takes his deck to the computer center, or to some remote site, where it is either picked up or input directly and, in due course, he receives his results. The results for the example program are shown in Fig. 1.6.



```
      PROGRAM ROOTS
      A=1.
      B=4.
      C=-12.
      BSQ=B**2
      DSCR=BSQ-(4.*A*C)
      SDS =SQRT(DSCR)
      R1=(-B+SDS)/(2.*A)
      R2=(-B-SDS)/(2.*A)
      PRINT 10, R1, R2
   10 FORMAT(1H1,//,5X,2F5.1)
      STOP
      END
```

Figure 1.3 Cards for the program of Fig. 1.2

Although we are *not* interested (within the scope of this text) in *computer science*, or in the inner workings of computers, etc., let us look briefly into what the computer system does to the FORTRAN deck. A small amount of knowledge here, from the outset, will help us to understand how to control our computer.

First of all, why do we have a "language" such as FORTRAN? We need it because the computer does not speak our language (English/Scientific or Business), but rather speaks (essentially) in binary arithmetic. Surely, enough texts and enough math courses discuss binary arithmetic so that we need not discuss it here. We immediately realize that for us to communicate with the computer in binary arithmetic