RELACION DE PROFESORES

PROGRAMACION DE COMPUTADORAS

Ing. Joaquín González Marín
Investigador del Inst. de Ingeniería
Sección de Investigación de Operación
Universidad Nacional Autónoma de México
México, D.F.

Ing. Andrés Lasaga Gómez
Director del Centro de Cálculo de
la Universidad Iberoamericana
Av. de las Torres 395
México 21, D.F.

Ing. Leonard Rapoport Yawitz
Sub-Gerente General
Procesos y Sistemas de Información, S.A.
Minería 145 Edif. D. Sótano
México, D.F.

Ing. Fernando Sosapavón Estrada
Jefe de la Oficina de Estudios Eléctricos
Gerencia Gral. de Planeación y Programación
Comisión Federal de Electricidad
México, D.F.

227106A

REFERENCE MANUAL

Introduction to FORTRAN

Revised 3-70

GENERAL ELECTRIC

INFORMATION SERVICE DEPARTMENT

Preface

FORTRAN is a computer programming language designed for the solution of mathematically oriented problems. It can be used, however, to good advantage in developing programs in order to solve business problems or in fact, to solve any problem which is capable of being solved on the computer. FORTRAN also lends itself to the Time-Sharing system because of the brevity with which problem solutions can be written.

This book consists of a logical progression of information and is designed to serve as an easy to follow guide in the use of FORTRAN and in the General Electric Time-Sharing system. It is not within the scope of this book to discuss all elements of the FORTRAN programming language as the FORTRAN Reference Manuals serve this purpose. Instead, topics most useful to the beginner are included here.

For the person who wishes to learn only of the general capabilities of FORTRAN and General Electric Time-Sharing system, the first three chapters are recommended.

Table of Contents

TABLE OF CONTENTS (CONT'D)

Section 1. INTRODUCTION

Before describing MARK I and MARK II FORTRAN, background information will be given under the following headings:

- Computers and Programming
- The FORTRAN Language
- Time-Sharing Concepts

COMPUTERS AND PROGRAMMING

Essentially, a computer is an electronic machine that processes data. By "processing of data," we mean that it converts data from one form to another. For example, computers are used in everyday life to do such things as

- Solve mathematical equations
- Calculate payrolls and generate billings
- Keep control of a business' inventory
- Monitor and control manufacturing of products

In order to do this processing of data, the computer must be supplied with detailed instructions. These instructions allow the computer to accomplish such operations as

- Arithmetic: addition, subtraction, multiplication and division.
- Comparisons of data: to see if one piece of data has a value that is greater, smaller, or equal to another piece of data.
- Tests of data: for such things as whether or not the data contains a plus or a minus sign, alphabetic or numeric characters, errors.
- Modification of data: such as additions to, deletions from, and re-arrangements of data.
- Read in data that is to be processed: from punched cards, tapes, disk, terminals via telephone lines.
- Return data that has been processed: by putting it on punched cards, storing it on tapes and disks, printing it out on printers and remote terminals.

A complete set of instructions which will process data is known as a program. The writing of programs is known as programming, and the person who does this is a programmer. The basic steps in writing a program and having the computer execute it are as follows:

1. The problem to be solved must be stated. For example, suppose we want to calculate the sum of all the numbers from 1 to 90.

2. The steps in solving it must be logically organized.

```
Set SUM to zero
Set X to 1
Add X to SUM
Is X=91 ?
   Yes - Go to END
   No
Add 1 to X
Repeat operation
END
```

3. The actual instructions which are given in order to solve the problem must be written in a computer language, for example, FORTRAN, if the problem is arithmetic.

4. These instructions (a program) are sent to the computer and are stored in the computer's memory.

5. The data to be processed is made available to the computer.

6. The computer executes, sequentially, the instructions of the program, one after the other, generating the required answers to the problem.

7. The computer makes the answers available by printing them out according to instructions given in the program.

Because it is virtually impossible for people to write programs efficiently and to communicate with the computer in its own binary language, various psuedo-computer languages have been developed, each for a particular use. FORTRAN, for example, was designed specifically for the solving of arithmetic problems of virtually any complexity.

FORTRAN

FORTRAN (an acronymn for FORmula TRANslator) is a computer language which is used in order to solve arithmetic problems. These arithmetic problems must have the following characteristics in order to use FORTRAN.

- They must be able to be stated in one or more equations.
- They must have numeric input.
- They must generate numeric answers.

Following are some of the features of FORTRAN programming.

- The basics of FORTRAN, sufficient to write programs, may be learned in a minimal amount of time.
- It can process real numbers, integers, and arrays.
- It can solve virtually any complex equation.
- It can easily repeat the execution of one or more equations, allowing all kinds of variation in the data to be calculated.
- It can accept and can process large amounts of data.
- It can print out large volumes of solutions.
- It can store data and programs in the system and retrieve them for use at a later date.

- It can supply subprograms to the user for inclusion in his program of such arithmetic operations as square root, sine, logarithms.
- It checks the user's programs for various kinds of errors.

TIME-SHARING

Time-Sharing is just that: the sharing of time on a computer by two or more users. More specifically, it means that a number of users at remote locations may have their terminals simultaneously connected to a central computer. Each user enters and executes his programs as if he were the only one using the computer at that moment. Time-Sharing is possible because of the tremendous speeds at which the computer operates.

Connecting a terminal to the computer involves no more than dialing a telephone number and typing in a few lines of information to the computer. Once this has been done, the user can type in his program and then request that it be executed.

The advantages of using time-sharing are:

- Its cost is low.
- It gives you all the features of a big computer.
- It saves time because the terminal is right in your office.
- Your programs are executed immediately.
- You can store your data and your programs for future use.

Section 2. THE ELEMENTS OF A FORTRAN PROGRAM

Before giving full details about the basic aspects of FORTRAN, it would be wise to summarize its main features and list some related information.

An example of a simple FORTRAN program is shown below. We suggest that you look at it carefully.

```
INPUT, GRAVITY, TIME
DISTANCE=GRAVITY/2.*TIME**2
PRINT, DISTANCE
```

Just what the three statements in this FORTRAN program mean will become clear as you read further.

Information about the basic elements of FORTRAN is given under the following headings:

- Structure of a FORTRAN Program
- Basic Statements: Arithmetic, INPUT and PRINT
- Rules About Statements
- Control Statements: IF, GOTO and DO

STRUCTURE OF A FORTRAN PROGRAM

A FORTRAN program consists of a number of statements. Each statement gives instructions or information to the computer on how data is to be supplied to the computer, processed or printed out. Specifically, these statements supply the following instructions and information to the computer:

- The one or more equations of the problem to be solved: (arithmetic statements).
- How data is to be supplied to the program: (INPUT statements).
- What answers (processed data) are to be printed out at the end of the program: (PRINT statements).
- The end of the program: (END statement).

In addition, the program may contain one or more statements which control the execution of other statements in the program as follows:

- Testing of data for a minus, zero, and plus condition, with the ability to shift the sequence of statement execution to any part of the program when such a condition is found: (IF statements).
- A direct shifting of the sequence of statement execution to any part of the program: (GOTO statements).
- The repeated execution of one or more statements in the program for as many times as is required (DO statements).

These are the statements with which a majority of FORTRAN programs can be written. When a computer executes a program containing these statements, the statements are executed, sequentially, in the order in which they were written. The only exception to this is when a control statement interrupts the current sequential execution and specifies that sequential execution begin in another part of the program.

A brief description of the statements plus the rules for writing them will be given in the remainder of this chapter. Details and variations of these statements will be described in Section 3.

BASIC STATEMENTS

There is no one group of FORTRAN statements which can be said to be the basic set which must be present in a program. However, each program must have an END statement and almost undoubtedly will also have arithmetic, INPUT, and PRINT statements.

Arithmetic Statements

Arithmetic statements are used to write down all equations of the problem to be solved. For example, the following equation is a valid FORTRAN statement:

```
A=B+C
```

The variable A is assigned the sum of the values of the variables B and C.

One main difference between the equations in FORTRAN arithmetic statements and algebraic equations is that all arithmetic operations in FORTRAN (addition, subtraction, multiplication, division, exponentiation) are indicated by specific symbols, shown as follows:

+	addition
-	subtraction
*	multiplication
/	division
**	exponentiation

The multiplication of A times B is always shown as A*B, never AB. The division of B by C is always shown as B/C and the 5th power of D is always D**5. The rules for using these symbols are the same as in algebra and are listed later in this chapter under the heading: Priorities of Arithmetic Operations.

Other examples of arithmetic statements are shown below:

```
X=A**2+B+3*C          DISTANCE=GRAVITY/2.*TIME*2
```

Another major difference between the equations in FORTRAN arithmetic statements and algebraic equations is that the following type of statement, K=K+1, is perfectly valid in FORTRAN but it is not in algebra. The statement K=K+1 means that the current value of K is to be increased by 1.

INPUT Statements

The INPUT statement specifies the variables in the program which require input data; it is used to indicate when these values are to be supplied to the program during its execution.

When the INPUT statement is read by the computer, the computer suspends execution of the program and causes a question mark to be typed out on the user's terminal. The user knows that he must then type in the values that are to be used by his program. An example of this is shown below:

```
INPUT, A, B, C
X=A**2+B+3*C
```

When the computer is executing a program of which these statements are a part, it will first read the INPUT statement, halt execution of the program, cause a question mark to be typed out on the user's terminal and then wait for the user to type in the required data. The user must then type in three real numbers which are to be the values of A, B and C. The computer will assign the value of the first number to the variable A, the value of the second number to the variable B and the value of the third number to the variable C. It will then execute the next statement by calculating the values of A^2, B and 3C and assign their sum to the variable X.

At this point, we should mention that it is quite possible for a FORTRAN program to have all of the required data included in the program in arithmetic statements. No INPUT statement would be required in this case. An example is shown below:

```
A=2
B=3
C=4
D=5
X=( (A*B)**C)**D
```

PRINT Statements

The PRINT statement specifies what variables in a program are to be printed out after the values for these variables have been calculated. Following is an example of the PRINT statement:

```
INPUT, A, B, C
X=A**2+B+3*C
PRINT, X
```

When the value of X has been calculated by the computer, the PRINT statement specifies that this value is to be printed out on the user's terminal.

Including the data with the program is only efficient, however, if the program is to be run one time. Otherwise, an INPUT statement should be used so that the statements within the program need not be changed each time the program is executed with new data.

There are other methods for supplying data to a program but they are for the advanced user of FORTRAN and will be mentioned only briefly in Section 7.

END Statements

The END statement indicates the end of a FORTRAN program and must be used as the last statement in all FORTRAN programs. Following is an example of the END statement:

```
INPUT, A, B, C
X=A**2+B+3*C
PRINT, X
END
```

This is a complete, valid, and executable FORTRAN program. When the user executes it on his terminal, the computer will cause a question mark to be typed out. The user will type in three numbers: the values of A, B, and C, respectively. The computer will then calculate the value of X and print it out on the user's terminal.

RULES AND INFORMATION ABOUT FORTRAN STATEMENTS

Now that some of the basic FORTRAN statements have been described, we will list some of the specific rules which must be followed and information about the statements. This is not to imply that the control statements described in the next section are any less important than the statements already described. On the contrary, they are extremely useful and are used in most FORTRAN programs. The information which follows will aid in your understanding the operation of the statements. Rules and information about FORTRAN statements are given under the following headings:

- Statement Rules
- Types of Statements
- Rules for Expressions
- Types of Numbers
- Variable Name Rules
- Priorities of Arithmetic Operations
- Identifying Statements by Statement Labels

Statement Rules

The way in which FORTRAN statements are written is very important. There is a great deal of variation allowed in arithmetic statements, but the rules for numbers and variable names must be carefully followed. The format of all other FORTRAN statements must also be followed carefully, right down to the commas. There are minor variations in some of these statements but you should be sure you know exactly what you are doing before making changes. The penalty for not being careful is a nonexecutable program.

The usual method of writing statements is to put one on each line. It is possible, however, to place more than one statement on a line by putting a semicolon at the end of each statement. For example, the following arithmetic statements can be written on separate lines, or on one line, as shown below:

```
X=1.0
Y=5.0          or          X=1.0; Y=5.0;Z=.0
Z=.0
```

A semicolon is not needed for the last (or only) statement on a line.

Only one variable may be used on the left side of an equation in FORTRAN. Examples of valid and invalid statements are shown below.

A=B+C-D	valid
A+B=3	not valid
LIST=0	valid
LIST+1=0	not valid

Types of Statements

There are two basic types of statements in FORTRAN:

- Arithmetic Statements
- Key Word Statements

Arithmetic Statements: The arithmetic statement is always an equation consisting of a variable name, an equal sign, and an expression:

variable name = expression

where: variable name = one or more letters and digits, the first character of which is always a letter

expression = a variable name or number, or two or more names and/or numbers, separated by arithmetic operators

Examples of arithmetic statements:

```
A=B          X=B**2+D*C          J=M+2
```

More details about variable names and expressions are given later in this section.

As mentioned previously, one difference between FORTRAN and algebra is that the variable name on the left side of an arithmetic statement may be used in the expression on the right side, making the equation unbalanced. Thus, the arithmetic statements below are valid in FORTRAN:

```
K=K+5

M=M-1
```

In the first example, the current value of K is increased by 5. In the second example, the current value of M is decreased by 1.

Key-word Statements: The key-word statement contains a special word that specifies an action to be taken or information to be supplied, plus a list (in most cases) which contains specific data. The list is made up of expressions and/or equations, each separated by commas. Examples of three key-word statements which have already been described are shown below:

```
INPUT, D, E, K

PRINT, A, B, X, Y, Z

END
```

Rules for Expressions

An expression can be any one of the following:

- A variable name
- A number
- Two or more variable names and/or numbers, separated by arithmetic signs

Examples of expressions:

```
A          20.575          X+Y          J+1

   A+B**2-C          -4          -X
```

Parentheses may be used around and within an expression, if needed. The descriptions of numbers and variable names follow.

Types of Numbers Used in FORTRAN

There are several types of numbers used in FORTRAN. They are as follows:

- Real Numbers
- Integers
- Double Precision Real Numbers (MARK II, only)
- Complex Numbers (MARK II, only)

Real Numbers: Real numbers (also known as floating-point numbers) contain a decimal point and therefore can be used in order to express fractional values. Some examples of real numbers are shown as follows:

```
.00095          1.414          -396297.123          25.          -.00000001
```

Since most FORTRAN programs process numbers with fractions, real numbers are used in order to represent the majority of values.

A real number may contain up to nine significant digits. If the value of the real number cannot be expressed with nine digits, an exponent must be used. An example of how different exponents can change the value of a real number is shown below:

$.8 \times 10^{3} = 800$

$.8 \times 10^{2} = 80$

$.8 \times 10^{1} = 8$

$.8 \times 10^{0} = .8$

$.8 \times 10^{-1} = .08$

$.8 \times 10^{-2} = .008$

Note that the decimal point is effectively moved one digit to the right when the exponent is increased by one and is moved one digit to the left when the exponent is decreased by one.

In FORTRAN, when a real number is written with an exponent, the exponent is shown by putting the letter E after the number, followed by the exponent. The following example shows some real numbers and how they are represented in FORTRAN:

```
800.                 can be          .8E3 or 8.E2 or 8E1

.008                 can be          .8E-2 or 8.E-3 or .08E-1

.0000000002567       can be          .2567E-9 or 25.67E-11 or 2567.E-13
```

In MARK I FORTRAN, real numbers may have values from .863616852 X 10^{-77} to .578960444 X 10^{77}. In MARK II FORTRAN, real numbers may have values from .363797880 X 10^{-11} to .274877907 X 10^{12}.

To summarize, real numbers may be written as follows:

- Without an exponent (that is, .5, 55.9, .000035, 2.)
- With an exponent, the decimal point always being to the left of the first significant digit (that is, .3E2, .31416E1, .186E6)
- With or without an exponent, the decimal point being in any part of the number (that is, 8.8E5, .05, .1414E1, 105E10)

Integers: Integers (also known as fixed-point numbers) are whole numbers and are therefore never written with a decimal point. Examples of integers are shown below:

2 -33 1500 -1 186000

Integers in MARK I FORTRAN cannot have a value higher than +524287 or lower than -524287. Integers in MARK II FORTRAN cannot have a value higher than +34359738367 or lower than -34359738367.

Although integers are not generally used to calculate values in FORTRAN, they are used as exponents in arithmetic statements and for control purposes which will be described in forthcoming pages.

Double Precision Real Numbers (MARK II): Double precision real numbers are used for greater precision in calculations in MARK II FORTRAN. They can contain up to 19 significant digits as compared to nine digits in the standard real number. Double precision real numbers always contain the letter "D" following the number. Examples of double precision numbers are shown as follows:

5.3379857390787D2 37D-1 .4742D3 175793807.333D3

Note in these examples that the decimal point can be located in any part of the number and that the same rules for exponents apply in these cases. If no decimal point is used in the number, it is assumed to be to the right of the least significant digit of the number.

Complex Numbers (MARK II): Without going into detail, we will simply mention that MARK II has the ability to handle complex (imaginary) numbers. For details, see the MARK II Reference Manual.

Variable Name Rules

A variable name is used in place of a number when the value of the number is to be varied in an expression or when the value of the number is to be calculated or assigned in an equation. In the arithmetic statement X=Y-3.5, for example, X and Y are variable names. The variable Y can be given any value and this current value determines the current value of the variable X.

A variable name may consist of one or more letters or a combination of two or more letters and numbers. The following rules must be followed when using variable names:

- The first character in a variable name must always be a letter of the alphabet.
- Any combination of letters and/or digits may follow the first letter.

- A variable name may contain up to 30 characters in MARK I FORTRAN and up to 79 characters in MARK II FORTRAN.

- The first letter of a variable name specifies the type of value (real or integer) being used, as follows: an initial letter of I to N indicates integer values, an initial letter of A to H or O to Z indicates real number values. All variable names and numbers in an arithmetic statement should be one type or the other.

Examples of how real numbers and integer values are specified in arithmetic statements are as follows:

PI=3.14159	(real)	I=2	(integer)
X=Y-3.5	(real)	J=K+3	(integer)
RADIUS5=CIRCUM*.159	(real)	LIST=M1+M2-M3	(integer)

If blanks are used in a variable name, they are ignored and the name is considered as identical to same string of characters without blanks. For example, the variable name AIR DENSITY is the same as AIRDENSITY.

- Only one name may be used to the left of the equal sign in an equation, as, for example, BREAKDOWN VOLTAGE=A+B+C.

- The following words must not be used as variable names or as the first part of variable names:

ASSIGN
BACKSPACE
CALL
CLOSEFILE
COMMON
CONTINUE
DATA
DIMENSION
END
ENDFILE
ENTRY
EQUIVALENCE
EXTERNAL
FORMAT
FUNCTION
GOTO
IF
INPUT
INTEGER
OPENFILE
PRINT
READ
REAL
RETURN
REWIND
STOP
SUBROUTINE
TYPE
WRITE

The reason for not using these words as variable names will become apparent as you read further. In addition, the following format should not be used in a variable name because it may be taken for the DO control statement.

DOnn...naaa...a

where n's are digits and a's are letters.

CONTROL STATEMENTS

Control statements do as the word implies: they control the execution of other statements in the program. Normal execution of statements in FORTRAN is sequential, one after the other, just as they are written. All three control statements are able to interrupt the sequential execution of statements and thus control the way in which the program processes the data. The following statements are described briefly, here, and in detail in Section 3.

- GOTO Statements
- IF Statements
- DO Statements

GOTO STATEMENTS

A GOTO statement interrupts the sequential execution of statements and switches execution to another part of the program. The computer then begins sequential execution of statements at the new location. The GOTO statement contains the statement label of the statement where execution is to resume. The GOTO statement is almost always used in conjunction with IF statements. Following is an example of a GOTO statement:

```
GOTO 20
  o
  o
  o
  o
20 (execution continues at this new location)
```

The value of the GOTO statement will be seen immediately when you read the description of the IF statement that follows.

IF Statements

The IF statement is used to shift the sequence of instructions being executed under certain conditions. The IF statement consists of an expression and three statement labels. When the computer executes the IF statement, it tests the expression to see if it is minus, zero, or plus. If the expression in the IF statement is minus, the sequence of execution is shifted to the statement having the first statement label. If zero, the sequence of execution is shifted to the statement having the second statement label. If plus, the sequence of execution is shifted to the statement having the third statement label. An example of an IF statement is shown below:

```
IF (A) 20, 30, 40
```

In this example, if A is minus, the next statement to be executed will have the label 20. If A is zero, the next statement to be executed will have the label 30. If A is plus, the next statement to be executed will have the label 40. An example of how this can be used follows:

```
IF(A) 20, 30, 40
20 A=1.0
GOTO 50
30 A=1.5
GOTO 50
40 A=3.0
50 (program continues)
```

In this example, if A is minus, it is assigned the value of 1.0. If A is zero, it is assigned the value of 1.5. If A is plus, it is assigned the value of 3.0. Notice how the GOTO statements are used to bring all three alternate paths to a common statement (50).

DO Statements

A DO statement allows one or more statements which follow it to be executed a specified number of times. The DO statement contains the word DO, the label of the last statement in the group to be executed, a counter, and the final value of the counter. Following is an example of a DO statement.

```
DO 20 I=1, 12
   .
   .
   .
20 (last statement of group)
```

In this example, 20 is the statement label of the last statement in the group to be executed and I=1, 12 is the counter whose initial value is set to 1 and which will allow the statements following the DO statement to be executed 12 times. This is an extremely brief description of the DO statement. Important details about the DO statement are given in the next section.

Section 3. DETAILS ABOUT FORTRAN STATEMENTS

Each of the basic statements in FORTRAN has been briefly described in Section 2. This was done to give a quick, clear, overall picture of FORTRAN. The details and variations which were left out in the initial description are described in this Section as are the following statements:

- Arithmetic Statements
- IF Statements
- GOTO Statements
- DO Statements
- INPUT Statements
- PRINT Statements

ARITHMETIC STATEMENTS

An arithmetic statement is either an equation which assigns a value to a variable or which requests calculation of an expression whose value is then assigned to a variable. Each statement contains one equation. For example, the equation A=B+C is a valid FORTRAN arithmetic statement. It states that the value of the variable named A is to be calculated from the sum of the values assigned to the variables named B and C.

Following are some common formulas, written as valid FORTRAN arithmetic statements.

P=A+B+C	(Perimeter of a triangle, P=a+b+c)
A=B/2*H	(Area of a triangle, A=1/2bh)
V=D**3	(Volume of a cube, $V=d^3$)

The format of an arithmetic statement, then, is

X = Y

where:

X = the name of a variable

Y = an arithmetic expression whose value is first calculated and then assigned to X

FORTRAN arithmetic statements offer some additional features. For example, the values of variables can be changed any number of times in a FORTRAN program by the use of additional arithmetic statements, shown as follows:

```
      .
      .
      .
      J=5
      .
      .
      .
      J=10
      .
      .
      .
      J=1
```

In this example, the value of J is initially set to 5. After a few statements, it is set to 10, and then later, to a value of 1.

In FORTRAN you can also take the current value of a variable and increment or decrement it, as

```
      .
      .
      .
      K=1
      .
      .
      .
      K=K+1
      .
      .
      .
```

In this example, a value of 1 was originally assigned to the variable K. Some statements later, the value of K is increased by one. The statement K=K+1 is not a balanced equation but is valid in FORTRAN. In effect, what it says is: take the current value of K, increase it by one and make it the new value of K. The ability to increment or decrement a variable can be very useful, for example, when the variable is used in a statement which is to be executed a number of times and the value of the variable must be changed by a fixed amount each time.

IF STATEMENTS

The IF statement is the only FORTRAN statement which can be specifically used for testing an expression's arithmetic condition (minus, zero, or plus). Depending on the condition found, the execution of the program statements will be shifted to one of three specified locations in the program.

The IF statement contains the word "IF", an expression in parentheses and three statement labels. An example is shown as

```
      .
      .
      IF(X+Y) 5, JOB, 12
      .
      .
      .
      5 X=5.
      .
      .
      JOB:X=2.5
      .
      .
      12 Y=0.
      .
      .
```

In this example, if the value of X+Y is negative, the next statement executed will be 5; if zero, the next statement executed will be JOB; if positive, the next statement executed will be 12.

The IF statement, then, allows you to continue the current sequence of statements or to jump to one of several locations in your program, according to the condition found. It is, in effect, a conditional test which can signal that new data is being generated, the next step in calculations has been started, or an error has been made. Basic variations of the IF statement are as follows:

1. If only two statement labels are used in an IF statement, they are used if the expression is negative or zero, respectively. If the expression is positive, the statement following the IF statement is executed next. For example:

```
o
o
IF(A)NEXT, DONE
A=B*C
o
o
```

In this example, if A is negative, program execution shifts to the statement labeled NEXT. If A is zero, program execution shifts to the statement labeled DONE. If A is positive, the next statement (A=B*C) is executed.

2. If only one statement label is used in an IF statement, it is used if the expression is negative. If the expression is zero or positive, the statement following the IF statement is executed. For example:

```
o
o
IF(F+R/P)20
J=3
o
o
```

In this example, if (F+R/P) is negative, program execution shifts to the statement labeled 20. If (F+R/P) is zero or positive, the next statement (J=3) will be executed.

Any expression whose values have been previously defined may be used in an IF statement. Standard statement labels, as defined in the introduction to this section, are used.

GOTO STATEMENTS

The GOTO statement is used to stop the sequential execution of instructions and to begin execution of instructions at another location in the program. The GOTO statement consists of the word "GOTO" and the statement label of the next statement to be executed. An example of a GOTO statement is shown as follows:

```
o
o
GOTO 30
o
o
30 A=B**2+C
o
o
```

In this example, sequential execution of statements is interrupted when the GOTO statement is reached. The computer jumps to the statement labeled 30 and begins executing statements from that point on.

GOTO statements are almost always used in conjunction with IF statements. A typical example of this is shown as follows:

```
o
o
IF(X) 10, 20, 30
10 X=5
GOTO 40
20 X=6
GOTO 40
30 X=7
40 A=X**2
o
o
```

In this example, if X is negative, it will be assigned the value 5. If X is zero, it will be assigned the value 6. If X is positive, it will be assigned the value 7. After X has been assigned its proper value under negative or zero conditions, the GOTO statements return execution to a central sequence of statements starting with A=X**2.

There is a more complex form of a GOTO statement called a "computed GOTO statement." It contains a group of statement labels and a variable name. The value of the variable name determines which statement label is to be used in the GOTO operation. If the value of the variable name is 1, the first statement label is used. If the value is 2, the second label is used, and so on. An example of this is shown below:

```
GOTO(5, 20, LIST, 30)INDEX
```

If the value of INDEX is 1, this is executed as a GOTO 5 statement.

If the value of INDEX is 2, this is executed as a GOTO 20 statement.

If the value of INDEX is 3, this is executed as a GOTO LIST statement.

If the value of INDEX is 4, this is executed as a GOTO 30 statement.

The value of INDEX may be changed at any time by an arithmetic statement, as for example, INDEX=2.

Restrictions in using this type of a GOTO statement are:

- The value of the variable name can be either an integer or a real number. If a real number, only the value to the left of the decimal point is used.
- The value of the variable name must be positive. If it is zero or negative, the program will halt.

DO STATEMENTS

DO statements allow one or more statements of a program to be executed as many times as is required. For example, if a formula has been executed 10 times, one right after the other, the DO statement will allow the formula to be written once but will execute it 10 times.

The DO statement consists of the word "DO", the statement label of the last statement of the group that is to be repetitively executed, a counter and the number of times the group of statements is to be executed. An example of a DO statement is shown below:

```
C
J=1
KSUM=0
DO 30 I=1, 10
KSUM=J**2+KSUM
30 J=J+1
o
o
o
```

where:

30 = the statement label of the last statement to be repetitively executed

I=1 = the counter whose initial value is 1

10 = the maximum counter value which will end the DO operation

In this example, the arithmetic statements KSUM=J**2+KSUM and J=J+1 will be executed 10 times by the DO statement. Each time they are executed, the value of I is automatically increased by one until it reaches the value of 10. When the statements have been executed 10 times, KSUM will be the sum of the squares of the integers 1 to 10. Note that the values of J and KSUM are specifically assigned before the DO statement.

The basic format of the DO statement is shown below:

DO label n_1, n_2, n_3

where:

label = the statement label of the last statement in the DO loop

n_1 = an equation which gives the initial value of the counter

n_2 = specifies the value that the counter must reach before the DO operation is finished

n_3 = the value by which the counter is increased each time and is not required unless the counter is incremented by a value greater or less than 1

Basic variations of a DO statement are as follows:

1. A name may be used instead of a number in the statement label, as shown in the example:

```
J=1
KSUM=0
DO CALC1, I=1, 10
KSUM=J**2+KSUM
CALC1:J=J+1
o
o
```

Note that a comma must be used in the DO statement after a statement name and that a colon must be used after the statement name in the last statement. Commas and colons are not required when using numbers.

2. The values of the counter in the DO statement may be used by the arithmetic statements during the DO operation, as shown in the example below:

```
o
o
KSUM=0
DO 30 I=1, 10
30 KSUM=I**2+KSUM
o
o
o
```

Note that we no longer need to use the variable J because the value of I can be used. Remember that the counter in this example (and as normally used in DO statements) is an integer and can not be used in arithmetic statements where a real number is required. Real numbers can be used in the counter of a DO statement provided certain precautions are taken and details about this are given when describing the restrictions of the DO statement.

3. The initial value of the counter may be any number as long as it is less than the final value, as shown in the following example:

```
DO 150 N=5, 10
```

In this example, the initial value of the counter will be 5 and the statements following this DO statement will be executed six times before the value of the counter reaches 10.

4. Increments greater than one may be used in the counter of a DO statement, as shown in the example:

```
DO MARK L=10, 25, 5
```

In this example, the initial value of the counter will be 10, and the value of the counter will be incremented by five each time, until the value is 25.

5. Increments less than one may be used in the counter of a DO statement provided that the numbers are real, as shown in the example below:

```
DO 5 W=.5, 20., .5
```

In this example, real numbers and a real number variable name are used. The increments are .5 until a value of 20. is reached by the counter. The complications in using real numbers in a DO statement are given under DO statement restrictions.

6. DO statements may be used within DO statements, as shown in the example:

```
o
o
DO 10 I=1, 10
DO 10 J=1, 10
o
```

In this example, the first DO statement sets its counter I to 1. The second DO statement sets its counter J to 1. The statements following the second DO statement are executed down to the last statement of the group whose statement number is 10. The second DO statement then sets its J counter to 2, executes the statements down to 10 again, sets its J counter to 3, and so on. When the second DO statement has cycled 10 times, return is made to the first DO statement whose I counter has had a value of 1 during this time. The I counter in the first DO statement is now set to 2 and the second DO statement again cycles 10 times. At the end of the entire operation, the second DO statement has cycled 100 times and the first DO statement has cycled 10 times. An example of a DO statement used within a DO statement is shown below:

```
o
o
KSUM=0
DO 40 I=1, 10
DO 40 J=1, 10
40 KSUM=(X(I, J))**2+KSUM
o
o
```

In this example, we are assuming that the array X has at least (10, 10) dimensions. Each of the 100 numbers in the array will be squared and the product added to the sum in KSUM.

- The formats we have shown in the examples must be followed.
- No arithmetic signs may be used in a DO statement.
- Although real numbers can be used in a DO statement, it is generally much wiser not to do so. The reason is that the value of fractions in real numbers is never the exact value, just as the exact value of one third can never be written as a number. An example of what can happen when real numbers are used in a DO statement is shown below:

```
o
o
DO 30 A=.1, 2.0, .1
```

In this example .1 can not be expressed exactly by the computer but will instead have the value .09960938. Thus, this DO statement will cycle 21 times instead of 20. If you use the value 1.95 instead of 2.0, it will correctly cycle 20 times as was desired.

- The value of the counter must not be changed by statements within the DO loop.
- The last statement within a DO loop can not be an IF, GOTO, RETURN, CHAIN or STOP statement. In order to obey this rule, put the dummy statement CONTINUE after any one of these statements and use it as the last statement of the DO loop, as shown in the following example.

```
o
o
DO 10 I=1, 10
DO 10 J=1, 10
IF(A(I)-B(J)10, 20, 10
10 CONTINUE
GOTO 30
LOC(1)=I
LOC(2)=J
30 (continue program)
```

In this example, CONTINUE is used as the last statement in the DO loop because the IF statement can not be used.

- A jump in statements must never be made from outside a DO loop into a DO loop. On the other hand, you are always able to transfer from within a DO loop to other statements outside by means of IF and GOTO statements.
- An inside DO loop may never have its last statement beyond the last statement of an outer DO loop.

INPUT STATEMENTS

The INPUT statement is used whenever data is to be supplied to a program during its execution. It contains a list of variables for which the user is to supply values. When the

computer comes to an INPUT statement during execution of a program, it suspends execution of the program, causes a question mark to be printed out on the user's terminal and waits for the user to type in the values of the variables listed in the INPUT statement. When the user does so, the computer resumes executing the program, using the values that were typed in by the user.

The basic INPUT statement consists of the word "INPUT" and a list of variable names for which values are to be supplied during execution of the program. An example of an INPUT statement follows:

```
INPUT, C, F, G, K, L
```

When the computer comes to this statement in the program and causes a question mark to be printed out on the user's terminal, the user must supply the real number values of C, F and G and the integer values of K and L.

More details on how data is typed in are given in Section 7 under "Listing and Executing Your Program."

There are other ways of supplying input data to a program. They are briefly mentioned in Section 7 and described in detail in the Reference Manual.

PRINT STATEMENTS

The PRINT statement is used to print out answers that have been generated by the program, current values of variables, messages, headings, and so forth. The basic PRINT statement consists of the word "PRINT" and a list of variable names. An example of a PRINT statement follows.

```
PRINT, A, B, C, L, M, X
```

This statement causes the current values of the real numbers A, B, C and X and integers L and M to be printed out on the user's terminals. Messages, headings and other alphabetic information may be printed out by means of the PRINT statement also. This is done by enclosing the information in quotation marks as follows:

```
PRINT, "RANDOM NUMBERS GENERATED ARE"
```

When the computer reads this statement, it will cause the message – RANDOM NUMBERS GENERATED ARE – to be printed out in the first 28 print positions of the user's terminal.

Unless a special format is specified, PRINT statements cause numeric data of a program to be printed out according to the following rules:

- Up to five numbers can be printed out on each line.
- Real numbers are printed without an exponent, if possible.
- The decimal points of real numbers are lined up beneath each other on succeeding lines, where possible.
- If a real number without an exponent has only two significant digits to the right of the decimal point, the remaining zeros are left off and the number is treated as a dollar-and-cents value.

Examples of these rules are shown as follows:

```
 373.45          0.20      143721.25      1857.60      42.14
14172.15       -136.16           2.00       117.25
                                  . . .
3.7821E+06         270           -43
```

Section 4. PROCESSING ARRAYS

An array is a list of numbers which is organized in a specific format. An example of an array is shown as follows:

-2	5	39	5
-10	-102	12	-8
4	0	-18	56

This array has two dimensions: three numbers down and four numbers across. We can describe this array with the notation of (3, 4). This simply means that the array has two dimensions: one containing three numbers and the other containing four numbers. If we had an array with four dimensions, each dimension containing five, ten, three and twelve numbers, respectively, it would be written as follows:

(5, 10, 3, 12)

In FORTRAN, arrays can be assigned names just like variables. For example, if we assign the name "STAT1" to the array shown above, we can specify any value in the array as follows:

STAT1 (1, 1) is the value -2
STAT1 (1, 3) is the value 39
STAT1 (3, 2) is the value 0
STAT1 (2, 4) is the value -8

In MARK I, up to 15 numbers may be put in one dimension and the array may have from 1 to 15 dimensions. In MARK II, up to 63 numbers may be put in one dimension and the array may have from 1 to 63 dimensions.

When an array is used in a FORTRAN program, a DIMENSION statement must be used to define the size of the array. The DIMENSION statement must be put at the beginning of the program before any reference to the array is made. An example of a DIMENSION statement follows:

```
DIMENSION ARRAY1(6, 10)
```

In this example, the size of the array named ARRAY1 is defined as two dimensions, the first dimension containing six numbers and the second dimension containing 10 numbers.

When the numbers in an array are to be supplied during execution of the program, the name of the array is specified in the INPUT statement, as in the example:

```
DIMENSION X(8, 8)
INPUT, X
```

In this example, the user will be expected to type in 64 numbers, that is, the size of the array as specified by the DIMENSION statement. When the individual numbers of an array are to be processed in a program, DO loops are normally used sequentially in order to specify

each number. For example, suppose that we wish to use the following formula in a program:

```
SUM=X**2+SUM
```

This arithmetic statement will square the current value of X and add it to the current value of SUM. If we wish to use this formula for the numbers in a 2-dimension array, we would make one change as shown in the following example:

```
SUM=X(I, J)**2+SUM
```

In order to specify all the numbers in the array by means of I and J, we would use two DO statements as shown below:

```
SUM=0
DO 10 I=1, 8
DO 10 J=1, 12
20 SUM=X(I, J)**2+SUM
```

In this example, all the numbers in the array named X would be squared and SUM would contain the sum of the squares.

An IF statement may be used to test the numbers in an array, as shown in the following examples:

```
IF(A(2, 3)) 10, 20, 30
IF(X(1, 1)-Z(1, 1)) 10, 20, 30
```

When array numbers are to be printed out, either part or all of the array may be printed. An example of how an entire array is printed out is shown below:

```
PRINT, ZRAY
```

In this example, all the numbers in the array named ZRAY will be printed out. If only a few numbers of an array are to be printed out they can be listed in the PRINT statement just as regular variables are listed, such as

```
PRINT, V(1, 2), V(2, 2), V(3, 4)
```

In this example, 3 numbers in the array named V are printed out. If we wish to print out a consecutive group of numbers in a single-dimension array, we can set up a counter as is used in a DO statement:

```
PRINT, (TEMP(I), I=6, 10)
```

In this example, let us assume that we have defined the array named TEMP as having 10 numbers in one dimension. This statement will then type out the last five numbers in the array. To print out a series of numbers in a 2-dimension array, we can use a statement similar to the one which follows:

```
PRINT, ((TEMP(I, J), I=1, 5), J=1, 5)
```

Assuming that the array named TEMP is larger than (5, 5), the first five numbers in the first five lines will be printed out.

Section 5. HOW TO WRITE A FORTRAN PROGRAM

To write a FORTRAN program, we suggest you do the following:

1. Summarize the problem you wish to solve in a statement. For example, you might make the following statement: "I wish to compare the weights of 100 men against a norm, keep a count of how many are over, under, and equal to the norm and calculate the average weight from all the weights used in this program."

2. Write down the basic equations to be used.

3. Convert the equations to FORTRAN equivalents.

4. Add DO statements if multiple operations are to be used.

5. Use IF statements for testing.

6. Set up the INPUT statement for supplying data to the program.

7. Decide what answers are to be printed out and write a PRINT statement accordingly.

8. Put an END statement at the end of the program.

9. Desk-check your program thoroughly to make sure it does what you want it to do. Try various values in your program to see if correct answers have been calculated.

10. Enter your program on the time-sharing terminal and see if you can execute it.

Section 6. AN EXAMPLE OF A FORTRAN PROGRAM

An example of an executable FORTRAN program is shown below. This program can be summed up by the following statement: Compare the weights of 100 people with a norm, keeping a count of how many are over, under, and equal to the norm and, at the end of the program, calculate the average weight of the 100 men.

```
INPUT, AVERAGE                          average weight to be typed in
KUNDER=0                                initializes counters to zero
KEQUAL=0
KOVER=0
DO 70 N=1, 100                          DO loop, 100 times
INPUT, WEIGHT                           weight typed in
IF(WEIGHT-AVERAGE) 20, 30, 40           compared weight to average
20 KUNDER=KUNDER+1                      underweight
GOTO 60
30 KEQUAL=KEQUAL+1                      equal weight
GOTO 60
40 KOVER=KOVER+1                        over weight
60 TOTAL=WEIGHT+TOTAL                   calculate total weight
70 CONTINUE
AVERAGE=TOTAL/100                       calculate average weight
PRINT, KUNDER, KEQUAL, KOVER, AVERAGE   print out answers
END
```

As soon as this program begins execution, a question mark is printed out, requesting that the average weight be typed in. The rest of the program is fairly self-explanatory. Each time the user types in a weight, this weight is compared with the average and one of the counters is increased by one, according to whether or not the current weight is below, above, or equal to the average weight. The weight is also added to a weight total and at the end of the program the average is calculated. The totals of the counters and the new average weight is then printed out and the program ends.

There are a number of improvements which can be made to this example. Some of the improvements which could be made would be to:

- Read in the weights as an array (which means that they will all be entered at one time).
- Allow the size of the array to be varied (which means that the number of weights which the program processes can be varied each time).
- Add comments to the program which include an overall description of the program as well as what function major parts of the program are performing.
- Print out messages specifying which data is to be typed in at that particular time.
- Print out headings for the answers.

The comments and the additional printouts will take a bit more time to add to the program; however, this will be more than compensated for because the user knows which data to enter at which time and the program is more quickly understood because of these comments, which make it easier to modify.

In the example that follows, we will show where comments are put in the program. The indication that they are comments is given when the program is entered on the terminal and these details are listed in Section 7 under "Entering Your FORTRAN Program."

```
**PROGRAM1
**THIS PROGRAM COMPARES THE WEIGHT OF A GROUP OF PEOPLE WITH A
**SUPPLIED AVERAGE.  A COUNT IS KEPT OF HOW MANY WEIGHTS ARE
**OVER, EQUAL TO AND UNDER THE AVERAGE.  AT THE END OF THE
**PROGRAM, THE AVERAGE WEIGHT OF THE WEIGHTS ENTERED IS CALCULATED
**THIS AVERAGE PLUS THE TOTAL COUNTS OF UNDER, EQUAL AND OVER
**WEIGHTS IS PRINTED OUT.
DIMENSION WEIGHT(K, L)
PRINT, "TYPE IN THE AVERAGE WEIGHT USING A REAL NUMBER"
INPUT, AVERAGE
PRINT, "TYPE IN THE 2 DIMENSIONS OF THE WEIGHT ARRAY USING INTEGERS"
INPUT, K, L
PRINT, "TYPE IN THE WEIGHTS TO BE PROCESSED USING REAL NUMBERS"
INPUT, WEIGHT
**THE COUNTERS ARE SET TO ZERO
KUNDER=0
KEQUAL=0
KOVER=0
**READ IN EACH WEIGHT FROM THE ARRAY, TEST IT AND ADD 1 TO THE
**APPROPRIATE COUNTER.
DO 70 I=1, K
DO 70 J=1, L
IF(WEIGHT(I, J)-AVERAGE) 20, 30, 40
20 KUNDER=KUNDER+1
GOTO 60
30 KEQUAL=KEQUAL+1
GOTO 60
40 KOVER=KOVER+1
60 TOTAL=WEIGHT(I, J)+TOTAL
70 CONTINUE
**CALCULATE THE AVERAGE WEIGHT OF THE WEIGHTS ENTERED
AVERAGE=TOTAL/(K*L)
PRINT, "UNDERWEIGHT", "EQUAL WEIGHT", "OVERWEIGHT", "AVERAGE"
PRINT, KUNDER, KEQUAL, KOVER, AVERAGE
END
```

Section 7. ENTERING AND EXECUTING FORTRAN PROGRAM ON A TERMINAL

Now that the basic statements of FORTRAN have been described we will discuss the operations of the Time-Sharing System and terminal related to entering and executing your program. The processes to be discussed include the following:

- Connecting your terminal to the System
- Entering and Listing your FORTRAN Program
- Executing and Debugging your Program
- Saving Your Program

CONNECTING YOUR TERMINAL TO THE SYSTEM

After a brief introduction to the keyboard and controls of the terminal this section presents a step-by-step method of dialing and connecting your terminal to the computer and identifying yourself and your FORTRAN program with the system. The terminal key-board used for input in the Time-Sharing System is similar to an ordinary typewriter keyboard without lower case letters.

Both the Model 33 and the Model 35 terminals are in current use with the Time-Sharing Service. The principle parts of these terminals include the following:

- Control Units
- Keyboard
- Paper Tape Punch (optional)
- Paper Tape Reader (optional)

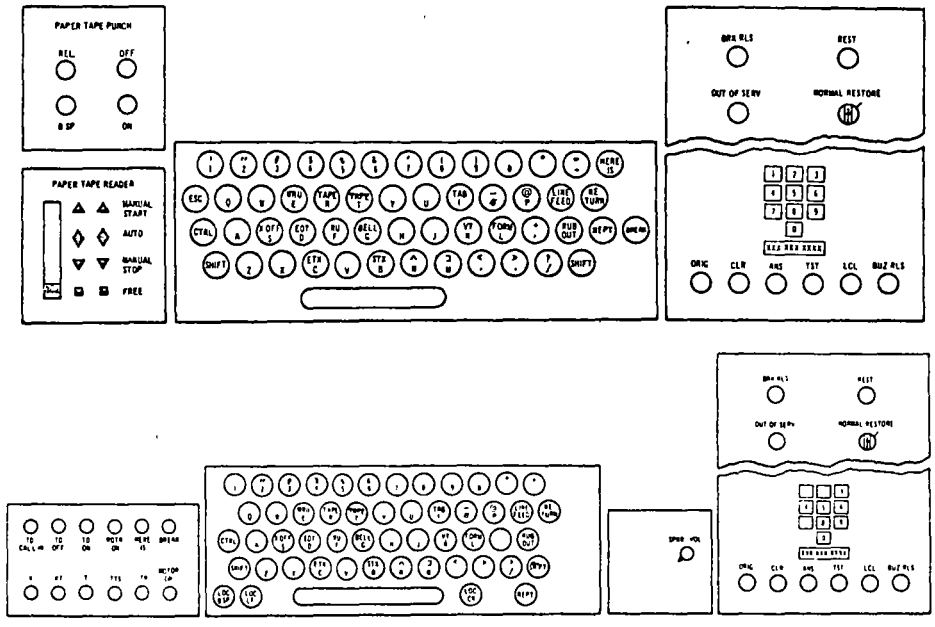


Figure 1. Model 33 Keyboard and controls (above) and Model 35 keyboard and controls (below)[1]

[1] On some Model 33 and Model 35 terminals the upward arrow (↑) is replaced by the caret (^) and the backward arrow(←) with the underline (_).

This Section refers to and explains only the use of those controls of the terminal which are needed to enter and to execute your program. For further information about all other controls and features of the terminal consult the Command Reference Manual.

ACTION	REASONS
1. Press ORIG key.	Turns on the terminal and obtains a dial tone.
2. Dial your time-sharing telephone number.	Connects the terminal to the computer. Under normal operation the phone will ring twice followed by a beep. Then the computer will send the following message: GE TIME-SHARING SERVICE TIME DATE USER NUMBER ---- If a busy signal occurs, this indicates that the computer is momentarily unavailable. Press the CLR button and repeat the process. If a connection cannot be made after several tries, dial the service number assigned to you for assistance.
3. Press the K button on control panel.	Allows input to be typed on the keyboard.
4. Type in your USER number.	Identifies you with the system. The user number is a unique six-character number which you have been assigned.
5. Press RETURN key.	Moves the print head to the left margin. The RETURN key must be pressed after every line of input is entered. It transfers the information which you have just typed on the terminal into the computer system. The system will then respond to your input. The computer sends the message: SYSTEM----
6. Type in FORTRAN.	This identifies the language in which your program is written. The computer sends the message· NEW OR OLD----
7. Type in NEW.	This indicates that a new program is to be entered and to be executed. If a program or data has been previously saved in the system and you wish to recall it, type in OLD.

ACTION	REASONS
	The computer sends the message: NEW FILE NAME ---- or (MARK II) ENTER FILE NAME----
8. Type in the name you have assigned to your program.	Identifies your program with the system. Qualifications for creating this name: 1. The first character must be a letter. 2. A name may consist of up to six letters and/or digits in MARK I and up to eight letters and/or digits in MARK II. 3. The remaining characters may be any combination of letters and digits. 4. Files cannot be named HEK, HEL, STN, STO, HELLO, STOP, or S because these words are interpreted as system commands. Now that the preliminaries have been completed, the computer sends the message: READY

A typical introductory conversation between you and the computer is illustrated below.

```
GE TIME-SHARING SERVICE
ON AT 10:03 W1 8/12/69

USER NUMBER -- B41822
SYSTEM -- FØRTRAN
NEW ØR ØLD -- NEW
NEW FILE NAME -- CHARLIE

READY
```

In the illustration, the underscores indicate which material is typed in by the user.

ENTERING AND LISTING YOUR PROGRAM

After you have connected your terminal to the system, the computer will send the message READY. This indicates that you can now begin with the typing in of the statements of your program. The information about entering and listing your program is described under the following headings:

- Entering Your Program
- Correcting, Deleting, and Adding Statements
- Listing Your Program

ENTERING YOUR PROGRAM

Each statement of your program is normally typed on its own line. Each line consists of a unique line number and a statement of the program. Line numbers allow you to specify each statement in your program so that after you have entered part or all of the program, you can:

- Change any statements you wish to change.
- Insert one or more new statements between two existing statements.
- Remove any statement which you wish removed.

The line numbers which you assign to the statements in your program:

- Must be in ascending order.
- Are usually incremented by 10; for example, 10, 20, 30, 40, etc.
- Must be separated from the statement by at least one space.
- May contain up to five digits.

An example of line numbers is shown in the example below:

```
10 INPUT, A, B, C
20 X=A**2B+3*C
30 PRINT, X
40 END
```

A line in a program may not contain more than 80 characters. In MARK I, a program cannot contain more than 256 lines or 6144 characters. MARK II programs have no defined limits on size.

The actual steps of entering the statements of your program are as follows:

ACTION	REASON
1. Type in the first line number.	Identifies the first statement.
2. Type in a space.	Separates the line number from the statement.
3. Enter the first statement on the program.	
4. Press RETURN (From this point on we will assume that you know you must press RETURN after you have typed all the characters that are to go on a line).	Indicates that no more characters are to be typed on this line.
5. Type in the next line number, space and statement, etc.	Repeat this process until all the statements in your program have been entered.
6. Check the program you have for errors:	Check for: • correct ascending line number sequence • typing errors • missing statements that were not entered.

7. Correct errors — By changing, adding, or removing statements.

An example of the method used in order to enter the program in Section 6 is shown as follows:

```
10 INPUT, AVERAGE
20 KUNDER=0
30 KEQUAL=0
40 KOVER=0
50 DO 70 N=1, 100
60 INPUT, WEIGHT
70 IF(WEIGHT-AVERAGE) 20, 30, 40
80 20 KUNDER=KUNDER+1
    •       •
    •       •
    •       •
```

Entering Comments with your Statements

To enter comments along with your statements type a comment on the line in the following format:

Type the line number, c or *, and then the comment.

An example of how you would enter comments and statements is shown below:

```
10*THIS PROGRAM COMPARES THE WEIGHT OF A GROUP OF PEOPLE WITH A
20*SUPPLIED AVERAGE
30 DIMENSION WEIGHT(K, L )
40 PRINT, "TYPE IN THE AVERAGE WEIGHT USING A REAL NUMBER"
50 INPUT, AVERAGE
 •
 •
```

Correcting, Deleting, and Adding Statements

Corrections can be made to a program either before or after all the statements are entered. It depends on when you see the error or when you decide to make a change. You have the option of correcting a statement, of deleting a statement, or of inserting a new statement between two previously entered statements

To change a statement that contains an error use the following format:

Type the line number of the statement, a blank and the correct statement.

To delete a statement that has been entered already, do the following:

Type the line number of the statement and press RETURN.

To insert a statement between two previously entered statements, you must select a line number whose value is between the line numbers of the two statements and do the following:

Type a line number whose value is between the line numbers of the previously entered statements, then type a blank and the new statement.	

If you make any changes, deletions, or insertions you will not actually see that they have been executed until you request that your program, as it now stands, be printed out (listed).

Listing

After you have changed, deleted and/or added statements in your program, you may wish to have the program printed out (listed) to make sure that now it is correct. To list your program use the following format in order to print out the current version of your program:

1. Type in the word LIST.
2. Check to see if the program is now correct.
3. Execute the program if it is, make additional changes if it is not.

EXECUTING AND DEBUGGING YOUR PROGRAM

After you have entered your program and checked it for errors, you can request that the computer execute it. This information is listed under the following headings:

- Executing Your Program
- Typing in Data During Program Execution
- Debugging Your Program
- After your Program is Executed
- Stopping Your Program During Execution

Executing Your Program

In order to begin execution of your program after it has been entered, do the following:

Type in the word RUN.	Requests that the computer execute your program. The computer will print out the name of your program, and the time and the date before it begins executing your program.

If your program has no format errors, the computer will execute it and print out what is specified by the PRINT statements in your program.

Typing In Data During Program Execution

If your program contains INPUT statements, the computer will suspend execution and type out a question mark each time it encounters an INPUT statement. The question mark indicates that you should type in the required data in the following format.

Type in the values, separating them by blanks or commas.	Supplies the data to the program
	If you did not type in all of the data required by the INPUT statement, the computer will type out another question mark.
	If you typed in more data than was required by the INPUT statement, the data is saved and is used by the next INPUT statement the computer encounters.
	The computer resumes execution of the program.

Debugging Your Program

There are two types of errors that you may find in your program:

- Format Errors
- Logic Errors

A format error means that you have not written a statement correctly. When the computer begins executing your program, it will check the format of each statement. If it finds an error, it will halt execution of the program and type out a message. A list of the error messages and their meanings is given in the Appendix of the Reference Manual. To correct the format error, do the following:

1. Type in the line number of the statement, retype the statement correctly.	This changes the statement.
2. To delete a statement, type in the line number.	
3. To change or to delete any other statements which require it, type in the word RUN.	Begins execution of the program again. If any other errors are found during the re-execution of the program, make the necessary corrections.

A logical error is an error in the logical procedure you are using to solve your problem and generate answers. Just because your program is executed does not automatically mean that the answers it generates are correct. You should know what the correct answers are when you initially execute your program. If the answers are not correct, check your program statement by statement to find where the logic is incorrect. When you find the error, change the necessary statements and try execution of the program again.

Following the Execution of Your Program

After the computer has finished executing your program, it will type out the following message:

USED XXXXX.XX UNITS

The xxxxx.xx specifies the number of units of computer time used to execute your program. If you have no other programs to enter or execute and you wish to disconnect your terminal from the computer, do the following:

1. Type in the word GOODBYE	Disconnents your terminal from the system. The system will then type the time, turn off the terminal, and break the telephone connection.

If you wish to enter or execute another program:

1. Type in the word NEW or OLD	Begins another operation. The computer will then type in the message: (MARK I) NEW FILE NAME OR OLD FILE NAME or
2. Type in the program name.	(MARK II) ENTER FILE NAME

Stopping Your Program During Execution

There are times during which you may wish to stop your program during its execution. For example, answers are being printed out which are obviously wrong or are not needed and you wish to stop the operation. To stop execution of the program, do the following:

1. Press the BREAK button and then the BRK-RLS button.	Ends execution of the program. Pressing the BREAK button stops execution and locks the terminal keyboard. Pressing the BRK-RLS button unlocks the keyboard.
2. Type in line number, etc.	Indicates that statements are to be changed or deleted.
3. Type in the word NEW or OLD	Indicates that another program is to be entered and/or executed.
4. Type in the word GOODBYE	Indicates that you wish to sign off the system.

Another way of halting execution of the program if there is no printout at the moment is to:

1. Simultaneously press the CONTROL, SHIFT and P keys.	Stops execution of the program when no printout is being done.
2. Take any one of the options as listed, above.	

SAVING YOUR PROGRAM

Whenever you enter a program in the GE Time-Sharing System, you have the option of saving it, that is, of storing it in the system so that you may use it at another time. If you do not save your program, it is destroyed when you sign off and must be re-entered if you wish to run it again. Information about saving your program is given under the following headings:

- How to Save Your Program
- Referring to The Catalog
- Setting Up a Password for Your Program
- Recalling a Saved Program
- Removing Saved Programs
- Replacing and Renaming Saved Programs

How to Save Your Program

1. Enter your program.	Answers as previously stated.
2. Type in the word SAVE.	This saves it. It will then be saved by the system and available for use at a later time.

When you save your first program, the system sets up a catalog for you, and each time that you save a program, the name of the program is put in this catalog.

Referring to the Catalog

In order to find out which programs you have saved, you must ask for a printout from the catalog in the following manner:

1. Type in the word CATALOG.	To request that the system print out the names of the programs you have saved.

Setting Up a Password for Your Program

If you wish to protect your program, either from other people, or to use it or to change it, you can assign a special password to it. Only those people who know the password will then be able to access the program. The password consists of from one to eight characters.

The first character must be a letter and the remaining seven characters may be any combination of letters and/or digits.

1. Enter the program.
2. Type in the words SAVE, password — Saves it and protects it by means of the password you have selected.

Anyone wishing to look at, or to use the program at a later date must then supply the password as well as the program name.

Recalling a Saved Program

In order to see, to use, or to change a previously saved program, take the following steps:

1. Type in the word OLD. — The system will then type out the message:

   OLD FILE NAME

2. Type in the name of the program you wish to use, to see, or to change.
3. Type in the password, if a password was assigned to this program — As soon as the program has been retrieved from storage, the system will type out the message.
4. Type in the word LIST or RUN — In order to list the statements of the program or to begin executing it.

Removing Saved Programs

When you wish to remove a saved program because it is no longer of use to you, take the following steps:

1. Type in the word OLD. — NEW OR OLD

   The system will then type out the message:

   OLD FILE NAME

2. Type in the name of the program you wish to remove.
3. Type in the password, if a password was assigned to this program. — As soon as the program has been retrieved from storage, the system will type out the message:

   READY

4. Type in the word UNSAVED — To remove the program from being stored by the system.

As soon as saved programs are no longer used, they should be removed from the system.

Replacing and Renaming Saved Program

If you try to save a program whose name is the same as a program you have previously saved, the system will send you the following message:

DUPLICATE FILE NAME, REPLACE OR RENAME

If you wish to delete the old program and store the new program in its place under the same name, type in the word:

REPLACE

If you wish to leave the older program in storage and also save the new program, give the new program another name. To do this, type in the word RENAME. The system will then request the name of the new program and you will respond by typing it in.

Section 8. ADVANCED FORTRAN CONCEPTS

Once the user has learned the basic operations of MARK I and MARK II FORTRAN, there are a number of advanced operations that can be used. These advanced operations will be listed and described, briefly, in this chapter. For details, see the MARK I or MARK II FORTRAN Reference Manual. These advanced capabilities are described under the following headings:

- Logical and Relational Operations
- Data Descriptions
- Other Method of Supplying Data to a Program
- Storing Processed and Raw Data
- Use of Subroutines and Functions
- Saving Storage Space
- Building a Program
- Linking Separate Programs
- Paper Tape Operations

LOGICAL AND RELATIONAL OPERATIONS (MARK II, ONLY)

In MARK II FORTRAN, the programmer is able to write logical expressions by using the logical operators AND, OR, and NOT which connect expressions that are either TRUE or FALSE. In MARK II FORTRAN, the following relational operators may be used in the expressions of his program:

.GT. greater than
.GE. greater than, or equal to
.LT. less than
.LE. less than, or equal to
.EQ. equal to
.NE. not equal to

DATA DESCRIPTIONS

The programmer has the option, in special situations, of assigning real number values to integer variable names, and integer values to real number variable names. This can be done by means of REAL, INTEGER and FORMAT statements.

The FORMAT statement also allows the programmer to specify the size and format of data being read into the computer and to specify the size and spacing of data being printed out or stored.

OTHER METHODS OF SUPPLYING DATA TO A PROGRAM

Data can be supplied to a program in several other ways than that of the INPUT statement.

- Data that has been previously saved (either generated by another program or typed in by a user) can be used by a program via a READ statement.
- In MARK I, a temporary data file ($DATA) can be set up for a program and used via a READ statement.
- In MARK II, a DATA statement which contains initial values of variables can be included in the program.

STORING PROCESSED AND RAW DATA

The programmer not only has the option of saving programs but also of saving the data output of his programs. He can then use the data as input to other programs or can simply store it until he is ready to print it out. He can also type in raw data and save it until he is ready to process it.

USE OF SUBROUTINES AND FUNCTIONS

Fortran supplies a number of standard subroutines that can be used in programs in order to calculate such things as square roots, sine, cosine, tangent, and logarithms. A complete list of these subroutines is given in the Appendix of the MARK I and MARK II Reference Manuals.

In addition, the programmer can write his own subroutines for his programs. Each time he wishes to use the subroutine, he can transfer to it from the main program, and when the subroutine has been executed, he can return to the same place in the main program. Although the subroutine can be used any number of times, it is necessary to enter it one time, only.

SAVING STORAGE SPACE

When a large program is processing a great deal of data, there may be a storage problem. FORTRAN allows the programmer to conserve storage space by allowing a number of programs to share the same data instead of each program needing its own data space. The COMMON and EQUIVALENCE statements are used for this purpose.

BUILDING A PROGRAM (MARK I, ONLY)

Mark I allows the programmer to build a program by means of assembling segments of programs he has previously saved and by putting them in any order he desires in order to form a new program. He is able to do this by means of $USE.

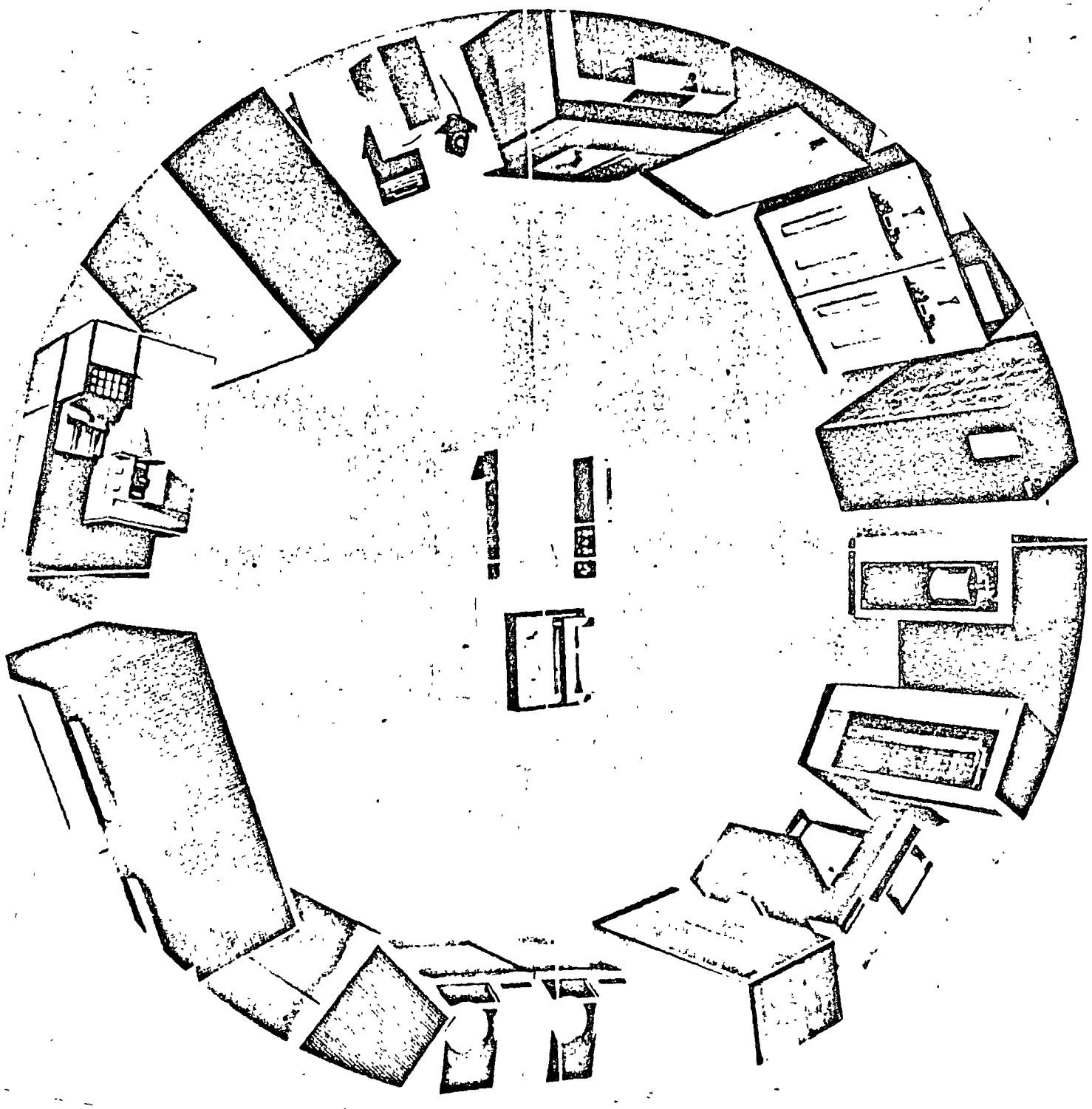
LINKING SEPARATE PROGRAMS

FORTRAN allows the programmer to link any number of saved programs together so that they will be consecutively executed without stopping. In MARK I, the $CHAIN Command is used. In Mark II, the CHAIN statement is used.

PAPER TAPE OPERATIONS

The terminal is able to read punched paper tape while connected to the computer and when operated independently. This means that data and programs may be put on punched tape before the terminal is connected to the computer and then may be sent to the computer after it is connected. It also means that messages, data, statements, programs, and answers which are typed in or printed out may also be punched on paper tape and reused at a later date. This can be a very practical alternate to saving programs, for example, when the program is relatively small and not used very often.

QUE HACEN LAS COMPUTADORAS Y COMO LO HACEN



El objeto de esta breve reseña sobre las computadoras electrónicas y sus múltiples aplicaciones al servicio del hombre, es transmitir al lector una completa visión de conjunto, mediante un lenguaje sencillo que permita comprender conceptualmente los temas tratados, sin necesidad de conocimientos previos en la materia.

Esperamos que estas páginas, muy simples en apariencia pero con profundo contenido, permitan, a quienes las lean, ingresar al maravilloso mundo de las máquinas automáticas.

PROLOGO

La necesidad de dominar la naturaleza y el medio ambiente ha nacido con el hombre, y lo ha impulsado a crear, ya en la era primitiva, herramientas capaces de amplificar su fuerza muscular. Más tarde, con idéntico fin pero en un nivel más evolucionado, el hombre inventó mecanismos y luego máquinas para las más diversas aplicaciones.

Pero esta sed de dominio es insaciable, y en el siglo XVII surgió una nueva meta: un amplificador de la inteligencia del hombre.

La aparición de estos amplificadores ha originado verdaderas revoluciones tecnológicas, sociales y económicas. A continuación enunciaremos brevemente los más importantes éventos que dieron lugar a dichos procesos:

PRIMERA REVOLUCION INDUSTRIAL

Siglo XVIII: La industria está en manos de los artesanos, cuya limitada producción no alcanza a satisfacer la creciente demanda. Los trabajos muy rudos, tales como el bombeo de agua en las minas y el arrastrar barcos río arriba, están a cargo de los esclavos. Las comunicaciones y los transportes se basan en lentos y primitivos métodos.

Año 1769: Jacobo Watt (Inglaterra, 1736-1819) inventa la Máquina de Vapor. Nace así el primer "Amplificador Automático de Potencia", y se lo aplica rápidamente a la navegación, la industria —especialmente la textil—, el bombeo de agua y la locomoción.

El artesano se vuelca en las fábricas, y la industria se centraliza alrededor de enormes máquinas de vapor.

Fines del siglo XIX: Hace su aparición el motor eléctrico, brindando la posibilidad de producir energía en pequeña escala y lejos de la planta generadora de fuerza motriz.

Esto da lugar a un nuevo tipo de artesanía, y la industria se descentraliza.

SEGUNDA REVOLUCION INDUSTRIAL

Año 1642: Blas Pascal (Francia, 1623-1662) inventa la primera máquina de sumar.

Año 1694: Godofredo Leibnitz (Alemania, 1646-1716) crea la primera máquina de multiplicar.

Siglos XVII a XIX: Isaac Newton (Inglaterra, 1642-1727) formula teorías que describen un Universo que se rige según leyes físicas exactas y simples.

Siglo XIX: Los físicos Boltzmann (Alemania) y Gibbs (Estados Unidos) introducen el concepto de "comportamiento más probable" de los fenómenos físicos. Estas ideas, que luego se extienden a otros campos, inician una era de estudios estadísticos.

La ciencia, la técnica y los negocios avanzan rápidamente. Hacia fines de siglo, se agregan las comunicaciones telegráficas. En todos los órdenes, las crecientes masas de datos que se manejan superan las posibilidades de los precarios medios existentes para procesar la información.

Año 1834: Charles Babbage (Inglaterra, 1792-1871) comienza la construcción de la primera computadora, capaz de "leer" datos perforados en código en tarjetas de cartulina, procesarlos e imprimir los resultados. Babbage adopta la idea de las tarjetas inspirado en un telar, que las utiliza, creado poco tiempo atrás por José María Jacquard (Francia, 1752-1834). La tecnología de la época está muy por debajo de las ambiciones del inventor: Babbage trabaja durante 37 años en la construcción de la computadora, y muere sin haber llegado a completarla. El proyecto es luego abandonado.

Año 1890: Hermann Hollerith (Estados Unidos, 1860-1929) crea el Equipo de Tabulación y Estadística, a base de tarjetas perforadas, para realizar un censo de población. Estas unidades electromecánicas se perfeccionarán luego y serán utilizadas como "equipo periférico" de las computadoras.

Año 1940: Norbert Wiener (Estados Unidos, 1894-1964) enuncia la Cibernética. Esta nueva ciencia, basada en la Teoría de los Mensajes, tiende a un lenguaje común a todas las ramas del saber humano: un "esperanto de las ciencias" que permita una comunicación más directa entre los científicos de distintas especialidades, para solucionar problemas comunes a ellos mediante máquinas automáticas.

Año 1944: Howard Aiken (Estados Unidos) crea la primera Computadora Electrónica: la "Mark I". Este primer "Amplificador Automático de Inteligencia" puede "aprender", y procesa la información a increíbles velocidades.

Tres siglos han pasado desde la sumadora de Pascal. Pero la ciencia y la tecnología —especialmente la Electrónica— avanzan a pasos agigantados, y la evolución de la computadora es acelerada desde el principio: en muy pocos años más, esta revolucionaria máquina automática se transformará en uno de los más fieles y útiles aliados del hombre.

Año 1954: Aparecen los primeros sistemas de "Teleprocesamiento de Datos", que posibilitan la descentralización de los procesos mediante unidades remotas que se comunican con la computadora a través de líneas telefónicas, telegráficas o de televisión, o bien por ondas electromagnéticas.

IBM

Existe una notable analogía entre ambas revoluciones industriales. La experiencia obtenida de la primera resultará de inapreciable valor para regular las situaciones que sobrevengan.

Por ser actores de la segunda, nos toca una gran responsabilidad: canalizar la enorme potencia de las computadoras en beneficio de la humanidad, para que las próximas generaciones dispongan de útiles autómatas que las liberen de pesados y rutinarios trabajos.

El hombre del futuro tendrá de esta manera una vida mejor, y un mayor tiempo libre para enriquecer su vida espiritual.

CONCEPTO DE COMPUTADORA

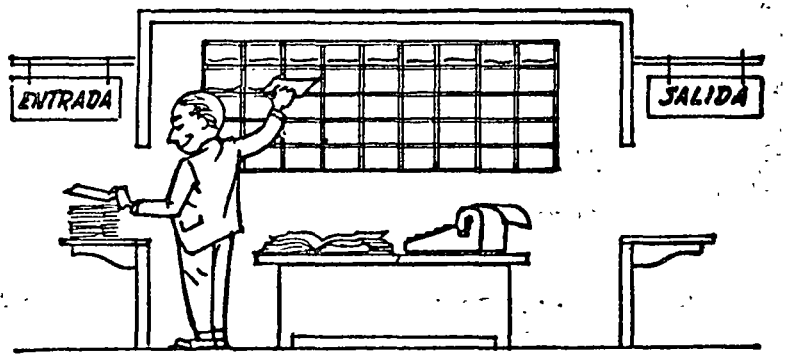
1

1 Este señor se llama **Control.** Trabaja en una pequeña habitación. Tiene a su disposición una máquina de calcular que suma, resta, multiplica y divide. Tiene también el señor Control un archivo parecido al casillero que existe en los trenes para clasificación postal.

Hay, además, en la habitación, dos ventanillas identificadas con sendos carteles: "Entrada" y "Salida".

El señor Control tiene un manual que le indica cómo debe desenvolverse con estos elementos, si alguien le pide que haga un trabajo.

2

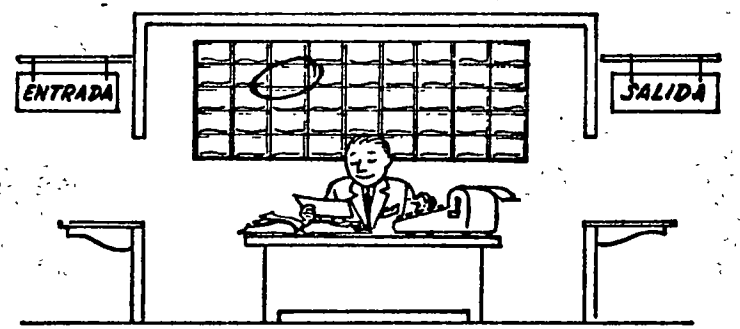


2 Una persona quiere saber el resultado de un complicado cálculo. Para ello, escribe ordenada, precisa y detalladamente, cada una de las operaciones que, en conjunto, integran ese cálculo, anota cada instrucción elemental en una hoja de papel y coloca todas las hojas en orden en la ventanilla "Entrada".

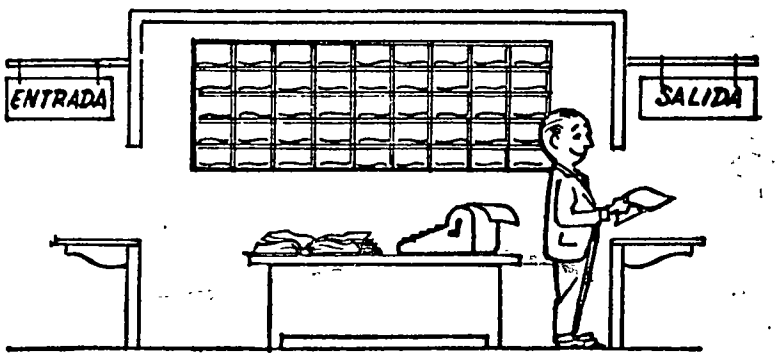
El señor Control, al ver las hojas, lee en su manual que debe tomar esas hojas con instrucciones, una por una, y colocarlas correlativamente en su archivo.

Y así lo hace.

3



4



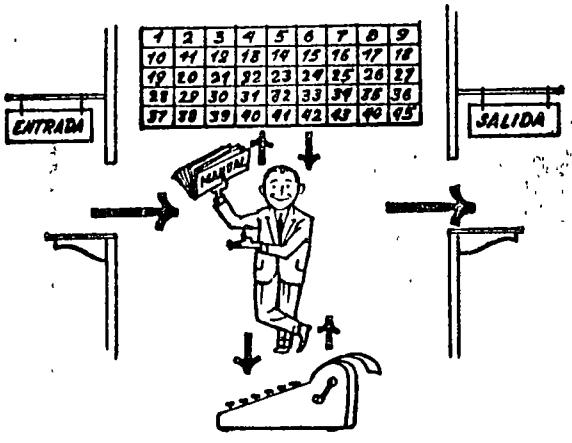
3 Una vez ubicadas todas las instrucciones en el archivo, el señor Control consulta nuevamente el manual. Allí se le indica que, a continuación, debe tomar la instrucción de la casilla 1 y ejecutarla; luego, la de la casilla 2 y ejecutarla, y así sucesivamente hasta ejecutar la última instrucción. Algunas instrucciones indicarán que hay que sumar una cantidad a otra (instrucciones aritméticas); otras, que el señor Control debe ir a la ventanilla "Entrada" para buscar algún dato que intervenga en el cálculo (instrucciones de "entrada/salida"), dato que la persona que le formuló el problema habrá colocado ya en dicha ventanilla, en otra hoja de papel.

Finalmente, otras instrucciones indicarán que debe elegirse una de entre dos alternativas (instrucciones lógicas): por ejemplo, supongamos que una parte del cálculo —desde la instrucción que está en la casilla 5 del archivo hasta la que está en la casilla 9— debe ejecutarse 15 veces porque el cálculo así lo exige.

En tal caso, la instrucción que está en la casilla 10 indicará que, si los pasos 5 a 9 se han ejecutado menos de 15 veces, se debe volver al paso 5. Cuando se hayan realizado las 15 repeticiones, y no antes, el señor Control seguirá con la instrucción de la casilla 11.

4 Después de ejecutar todas las instrucciones del archivo, haciendo con la máquina de calcular las operaciones en ellas indicadas, el señor Control entrega, a través de la ventanilla "Salida", los resultados obtenidos... y se sienta a esperar un nuevo trabajo.

5



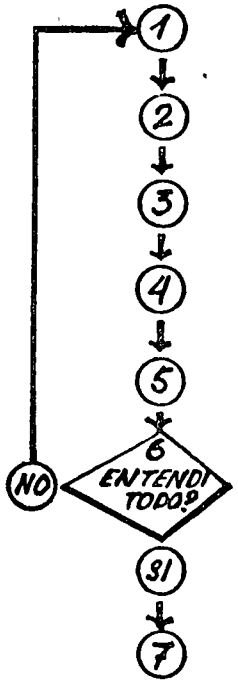
5 Obsérvese que la actuación del señor Control es puramente mecánica: sólo sigue las indicaciones de su manual y cumple de acuerdo con ellas las instrucciones que recibe a través de la ventanilla "Entrada".

Toma decisiones, pero solamente cuando se le señalan las alternativas que existen y con qué criterio debe elegir una de ellas.

El señor Control puede resolvernos cualquier problema, por complicado que éste sea. Pero para ello debemos indicarle paso a paso, en la forma más elemental y detallada, todo lo que debe hacer para resolverlo, sin olvidarnos absolutamente nada porque, en ese caso, el señor Control no sabría continuar por sí mismo.

Haga el lector la prueba de formular un problema cualquiera de modo tal que una persona que no conozca nada acerca de ese problema pueda resolverlo sin necesidad de hacer consultas. Verá que es una experiencia interesantísima.

6



6 Ahora pediremos al lector que reflexione sobre todo lo dicho. Si algún detalle no está suficientemente claro, le rogamos que lea nuevamente, a partir de la figura 1. Si ha comprendido todo, siga adelante.

(Observe, de paso, que el párrafo anterior es una instrucción lógica.)

7 El esquema que acabamos de representar mediante el señor Control y sus elementos de trabajo, corresponde exactamente al esquema de funcionamiento de una computadora electrónica: las **unidades de entrada** (representadas por la ventanilla "Entrada") son, en la computadora, unidades que leen tarjetas o cintas de papel perforadas en código por otras máquinas, o bien documentos impresos, teclados y una gran variedad de elementos de entrada.

La unidad de almacenamiento o memoria (representada por el archivo del señor Control) permite registrar las instrucciones y los datos para resolver un problema. Su funcionamiento se basa, normalmente, en pequeños anillos que pueden magnetizarse en un sentido o en otro, "recordando" así un 1 o un 0 respectivamente. Con 8 de esos anillos se forma una "posición de memoria", en la cual puede registrarse una letra, un dígito o un carácter especial, según las distintas combinaciones de anillos "en 1" y anillos "en 0", de acuerdo con un código predeterminado.

En varias posiciones consecutivas pueden "memorizarse" así palabras o cifras completas.

En esta forma se graban los datos y las instrucciones en el Almacenamiento. La información se registra, se procesa y se "borra" a enorme velocidad.

El dispositivo aritmético (representado por la máquina de calcular) realiza las cuatro operaciones aritméticas.

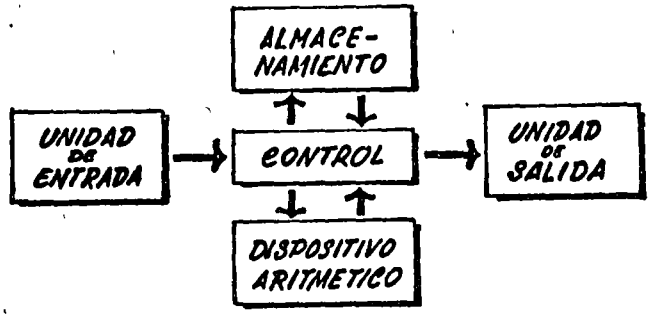
Se trata de un dispositivo electrónico automático.

Las unidades de salida (representadas por la ventanilla "Salida") pueden ser impresoras, máquinas de escribir, grabadores de cintas magnéticas e, inclusive, unidades de respuesta oral o pantallas de televisión que muestran imágenes generadas por la computadora, además de aparatos que dibujan automáticamente y una gran variedad de otros dispositivos de salida.

Finalmente, un dispositivo electrónico de control (representado en nuestro esquema por el señor Control)

IBM

7



gobierna todas las operaciones de todas las unidades que componen la computadora. Consiste también en un dispositivo electrónico automático.

8

8 Vamos a simplificar aún más el esquema: como todas las computadoras tienen Almacenamiento, Dispositivo Aritmético y, por supuesto, Unidad de Control, llamaremos "Unidad Central" al conjunto de estos tres elementos.

Veremos, más adelante, cómo puede un esquema tan simple, mediante una adecuada cantidad de instrucciones también simples, resolver desde los más elementales problemas hasta las más asombrosas, refinadas y complejas realizaciones del hombre.

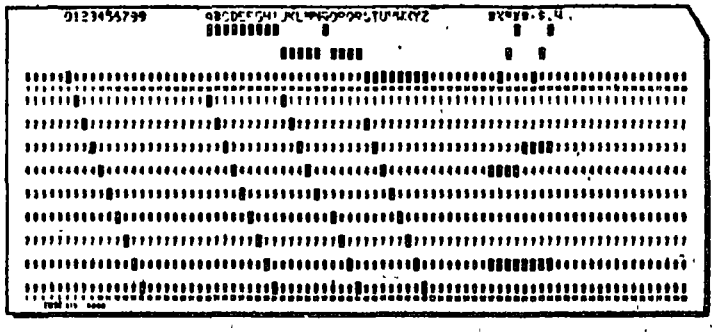
TARJETAS PERFORADAS

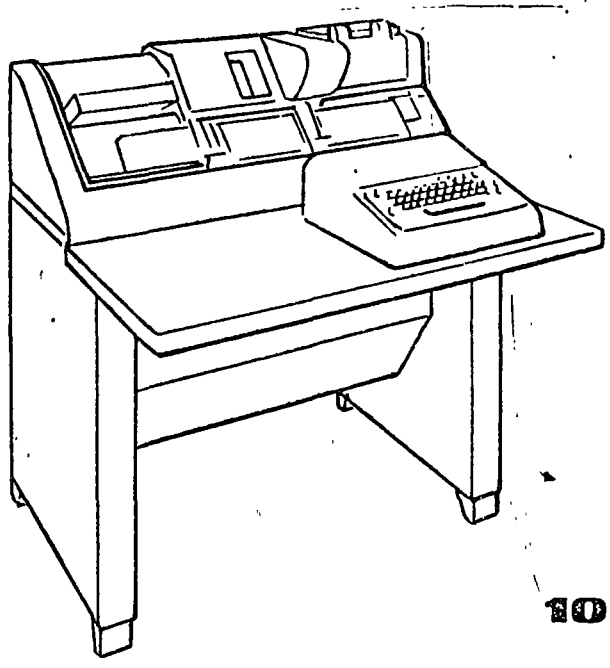
9 Ahora que el lector conoce conceptualmente el funcionamiento elemental de una computadora, veremos, a través de algunas unidades reales, cómo se han llevado a la práctica estos conceptos.

Para que la computadora pueda "leer" automáticamente las instrucciones y los datos que le entregaremos, se ha inventado un sistema basado en tarjetas de cartulina que se perforan de manera que cada perforación significa un número, una letra o un símbolo especial, de acuerdo con un código predeterminado.

En una tarjeta caben 80 letras, números o símbolos formando palabras y cifras.

9





10

10 Esta máquina se llama "Perforadora de Tarjetas". Tiene un teclado similar al de una máquina de escribir. A medida que oprimimos sus teclas, hace perforaciones en código en tarjetas sucesivas, y, además, imprime las letras, los números y los símbolos respectivos en el borde superior de las tarjetas.

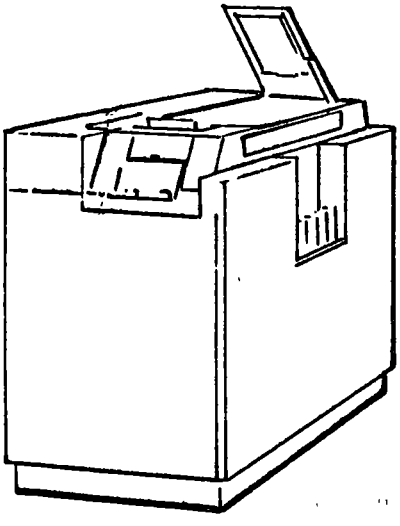
Mediante la Perforadora de Tarjetas hemos "perforado" en tarjetas las instrucciones y los datos correspondientes a un problema que queremos plantear a la computadora. Tomamos esas tarjetas y, ahora sí, nos dirigimos hacia la computadora electrónica.

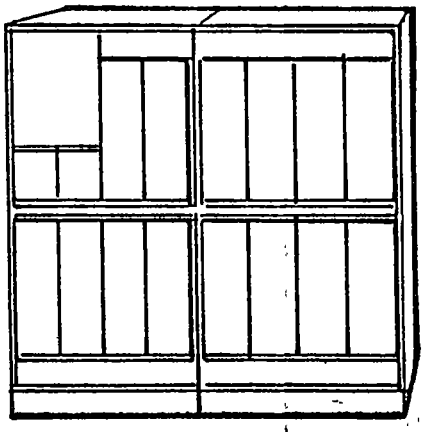
COMPUTADORA ELECTRONICA

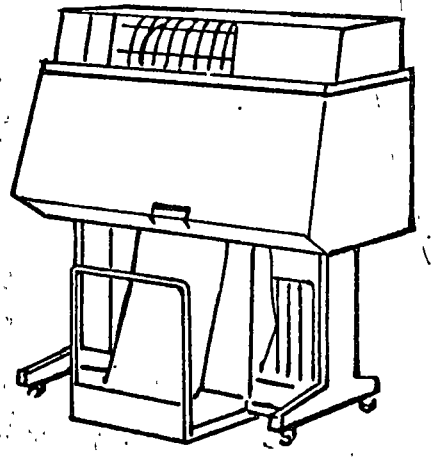
IBM

11 Esta es una computadora. A la izquierda, puede verse la "unidad de entrada". En este caso, se trata de una Lectora de Tarjetas Perforadas. A enorme velocidad, esta unidad "lee" automáticamente las tarjetas y transmite la información a la "unidad central" (en el centro de la figura). Allí están la "unidad de control", el "almacenamiento" o "memoria magnética" y el "dispositivo aritmético". Todos estos elementos son electrónicos y transistorizados y operan a tan altas velocidades que los tiempos de operación se miden en "microsegundos" (millonésimas de segundo) y aun en "nanosegundos" (milmillonésimas de segundo). En tercer lugar, se observa una Impresora, que actúa como "unidad de salida". Se pueden imprimir los resultados en esta unidad a razón de más de mil renglones por minuto.

11



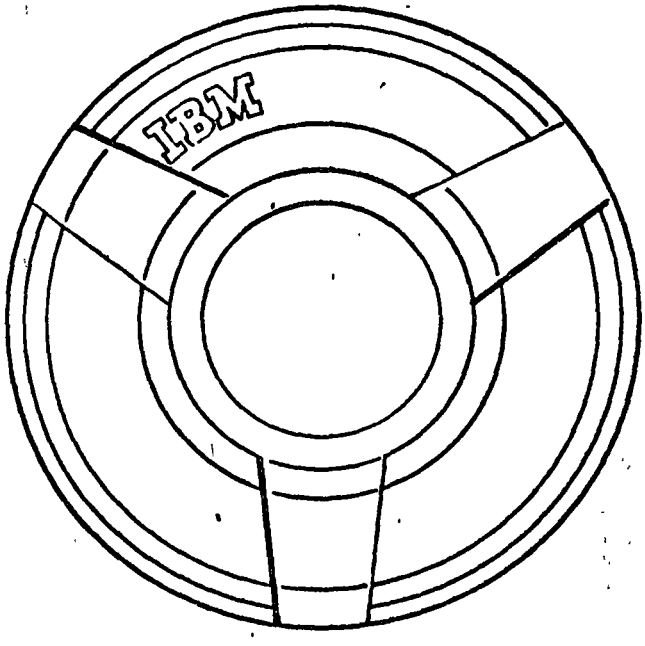




CINTA MAGNETICA

La velocidad máxima a que pueden leerse las tarjetas está limitada por la resistencia del papel especial con que éstas se fabrican. Para procesos que implican muy grandes cantidades de datos (por ejemplo: el archivo de cuentas corrientes de un banco), se desarrollaron otros tipos de unidades de registro o "memorias externas", más veloces y que permiten almacenar la información en forma "más concentrada". Una de ellas es la cinta magnética.

12



12 Consiste en una cinta de plástico de 730 mts. de longitud, aproximadamente, con una de sus caras recubierta por un material magnetizable, enrollada en un carrete. Como vemos, es prácticamente igual a u cinta de grabador. Sólo se diferencia de ésta por dimensiones, resistencia, etc.

En la cara magnetizable, puede grabarse información en forma de puntos magnetizados, que, en un código apropiado, permiten registrar todos los caracteres necesarios (letras, números, etc.).

La misma cabeza grabadora puede usarse para "leer" la información de la cinta (pero no simultáneamente con la grabación).

Este tipo de "memoria" externa puede recibir o enviar información a la unidad central a velocidades que van desde 10.000 hasta 680.000 caracteres por segundo.

La información está "concentrada" a razón de 80 a 2.400 caracteres por centímetro de longitud.

Como la cinta se va desenrollando de su carrete y enrollando en otro, para que su uso sea práctico la información debe estar grabada en orden, para procesarla a medida que se desenrolla la cinta, sin tener que ir y volver a lo largo de ella. Por ello se dice que la cinta magnética constituye un "Archivo Secuencial".

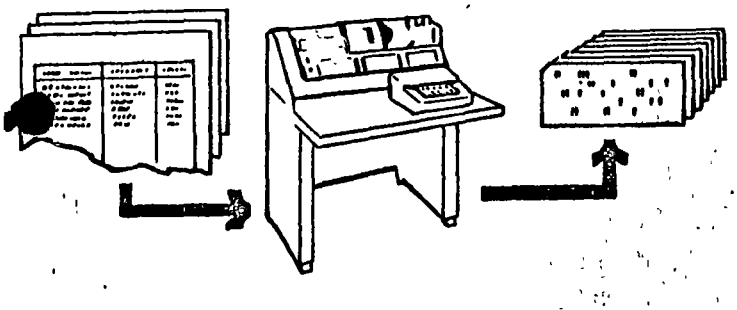
APLICACION CON CINTAS MAGNETICAS

IBM

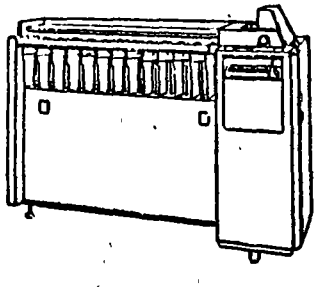
13

APELLIDO Y NOMBRE	HORAS TRABAJADAS	VALOR HORA M$N
PEREZ, JUAN	8,35	150

14



15



Veremos ahora un ejemplo de aplicación de la computadora electrónica. Plantearemos un problema sencillo para mostrar en qué forma se puede encarar su solución.

Supongamos que una empresa tiene una fábrica y que se quiere mecanizar el cálculo de jornales de los obreros.

13 A tal efecto, el capataz de cada sección confecciona diariamente una planilla en la cual indica el nombre de cada operario, la cantidad de horas trabajadas por el mismo ese día y el valor hora del trabajo realizado.

14 Las planillas provenientes de todas las secciones son enviadas, al fin de cada día, a la oficina de perforación. Allí se perfora una tarjeta por cada renglón de cada planilla.

15 Finalizada la quincena, se reúnen todas las tarjetas y se las clasifica de manera que todas las tarjetas correspondientes a Juan Pérez, por ejemplo, queden agrupadas. Además, esos grupos de tarjetas —correspondientes cada uno de ellos a un operario— se clasifican por orden alfabético. Todas estas operaciones de clasificación son llevadas a cabo por medio de una máquina clasificadora de tarjetas, fuera de la computadora.

16

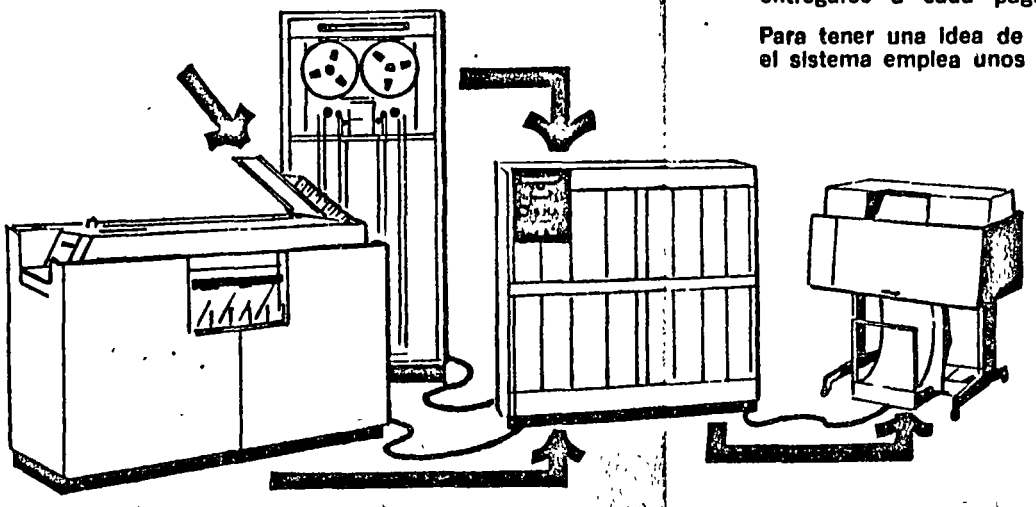
16 El "lote" de tarjetas de la quincena, ya clasificado, se coloca en la lectora de tarjetas de la computadora. Al leerse las tarjetas, el contenido de las mismas es transmitido a la unidad central. Por otra parte, está grabado de antemano en una cinta magnética el archivo de empleados, que contiene datos tales como bonificaciones, salario familiar, descuentos, etcétera.

Así, cada vez que se leen las tarjetas de un operario, se "leen" de la cinta magnética los datos constantes correspondientes.

Entonces la unidad central multiplica las horas por el "valor hora", suma los jornales, opera las bonificaciones y los descuentos y transmite a la impresora la liquidación completa.

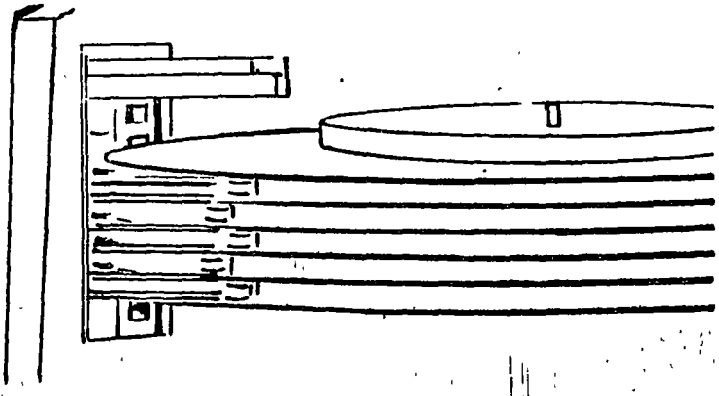
Normalmente, pueden obtenerse en este proceso: 1) Los sobres de pago. 2) Las planillas de haberes por sector. 3) Los listados de retenciones. 4) Las planillas de distribución de billetes (donde se indica la cantidad de billetes y monedas que debe entregarse a cada pagador). 5) Planillas estadísticas, et

Para tener una idea de la velocidad del proceso, diremos qu el sistema emplea unos 30 minutos por cada 1.000 operarios.

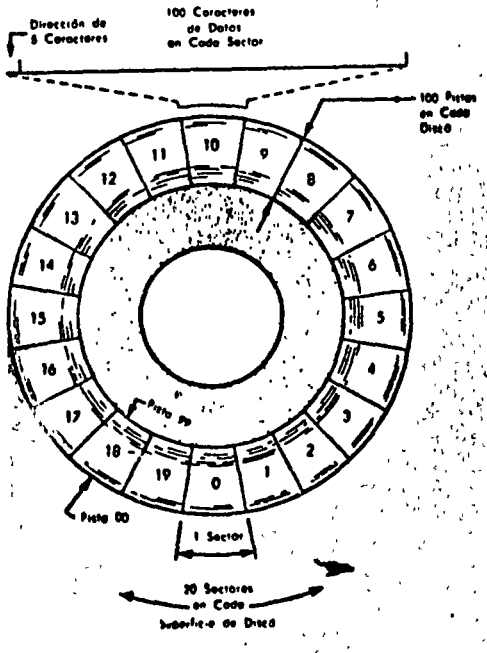


DISCOS MAGNETICOS

17



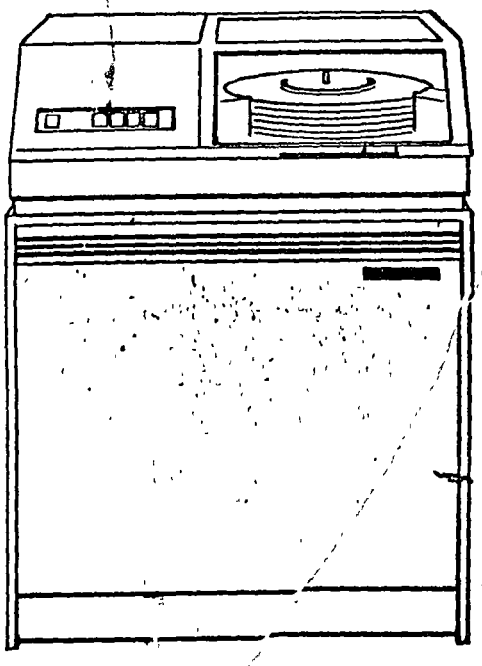
18



Existe otra unidad de entrada/salida que, al igual que la cinta magnética, se utiliza para grabar archivos. Por tal razón, se la denomina también "unidad de memoria externa" de la computadora. Se trata de la "Unidad de Discos Magnéticos".

17 Varios discos metálicos, recubiertos en sus caras por un material magnetizable, giran sobre un mismo eje. Unos brazos dispuestos en forma de peine se desplazan simultáneamente sobre las caras de los discos, y en cada brazo hay cabezas lectoras/grabadoras que graban o leen información sobre las superficies de los discos, en forma de puntos magnetizados y de acuerdo con un código predeterminado.

18 Los datos se graban en sentido circular sobre los discos. Esta figura muestra la disposición adoptada para ordenar el archivo. En ambas caras de un solo disco, cuyo diámetro es de 30 cms., se pueden grabar 400.000 letras, números y caracteres especiales, formando palabras, cifras, registros completos.



19

19 Aquí vemos la unidad de discos. El "juego" de discos está colocado y listo para grabar en él un archivo, para leerlo o para actualizarlo. Puesto que los juegos de discos son intercambiables, la extensión del archivo puede ser ilimitada.

¿Qué diferencia existe entre un archivo a base de cintas magnéticas y otro a base de discos magnéticos? En las cintas, los registros de información se graban o leen secuencialmente.

En los discos, en cambio, se tiene "libre acceso" a un registro cualquiera, en forma inmediata, pues cada registro se localiza por su posición física dentro del juego de discos. Por eso, se da también el nombre de "Memoria de Acceso Directo" a la unidad de discos.

Las unidades de Discos Magnéticos Intercambiables pueden grabar o leer a razón de 77.000 a 312.000 caracteres por segundo. El tiempo de "acceso" a un registro cualquiera alcanza un promedio de 60 milisegundos (milésimas de segundo).

APLICACION CON DISCOS MAGNETICOS

Ahora mostraremos un ejemplo de aplicación en el cual intervienen todas las unidades vistas hasta el momento.

20 Supongamos una computadora con esta configuración, instalada en un Banco para realizar el proceso de Cuentas Corrientes. En la Unidad de Discos se han grabado los saldos de todas las cuentas y, al lado de cada saldo, se ha grabado el número de cuenta correspondiente.

En la ventanilla del cajero hay una máquina eléctrica de escribir, conectada a la computadora, que funciona como unidad de entrada y salida de datos.

Cuando un cliente del Banco se acerca a la ventanilla para hacer un depósito, el cajero escribe, mediante el teclado, el número de la cuenta y el importe depositado. Los datos son transmitidos automáticamente a la Unidad Central la cual, comandada a su vez por el programa almacenado en su memoria, actualiza el saldo en los discos.

Si el cliente quiere hacer una extracción de fondos, el procedimiento a seguir es análogo, salvo que, en este caso, antes de efectuar el proceso, el cajero "pregunta" a la computadora el saldo de la cuenta.

La Unidad Central "lee" el saldo en los discos y lo transmite a la máquina de escribir. Comprobada así la existencia de fondos suficientes, el cajero procede a concretar la extracción.

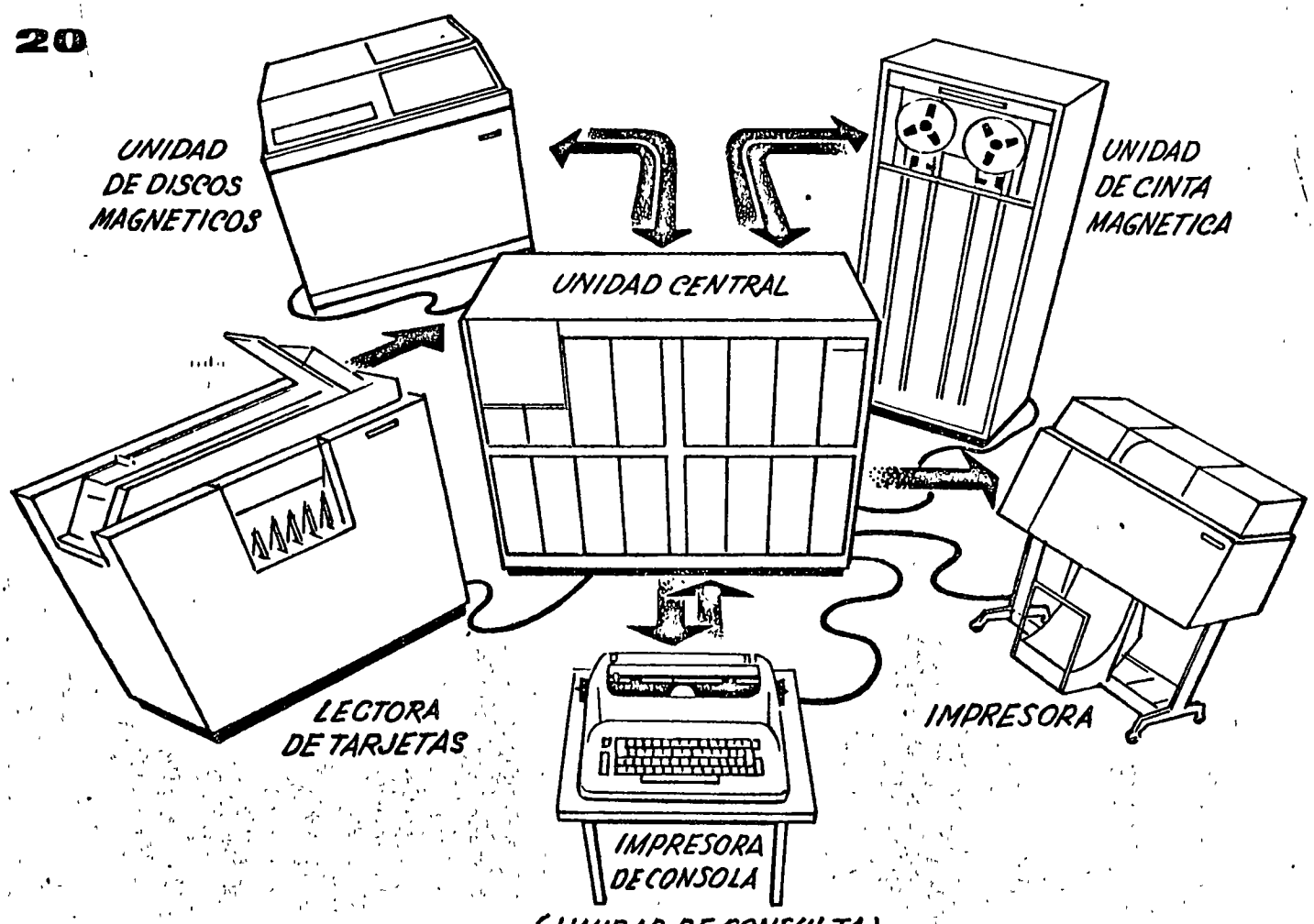
Simultáneamente, se graban en la Unidad de Cinta todos los movimientos que se producen, y se obtiene así un registro histórico de todas las operaciones. La atención del cliente demanda escasos segundos.

A través de la Impresora, se emiten periódicamente resúmenes de cuentas, balances, listados, estadísticas y muchos otros estados impresos correspondientes a ésta y a otras aplicaciones que se realicen con el sistema.

Pueden conectarse en esta forma muchas Unidades de Consulta a la Unidad Central.

Observe el lector que, en el ejemplo presente, la computadora hace el proceso simultáneamente con la realización de las transacciones del Banco. A las tareas de este tipo se las denomina "Aplicaciones en Tiempo Real".

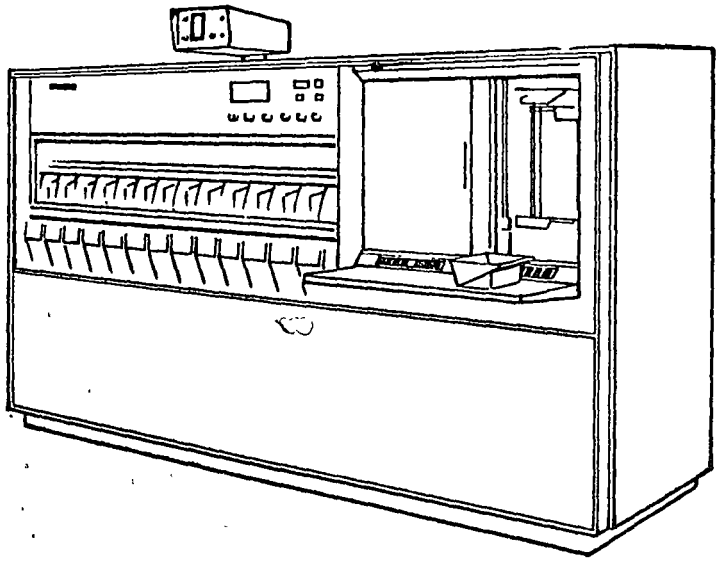
20


UNIDAD DE DISCOS MAGNETICOS
UNIDAD CENTRAL
UNIDAD DE CINTA MAGNETICA
LECTORA DE TARJETAS
IMPRESORA
IMPRESORA DE CONSOLA
(UNIDAD DE CONSULTA)

LECTORA OPTICA DE CARACTERES IMPRESOS

Una de las conquistas más espectaculares en el campo del procesamiento de datos, es la lectura óptica de caracteres impresos. El aspecto más importante de esta realización es el hecho de que un documento impreso por una máquina de escribir, o por una máquina de contabilidad, o por la impresora de una computadora, puede ser leído indistintamente por el ojo humano o por una máquina automática.

21

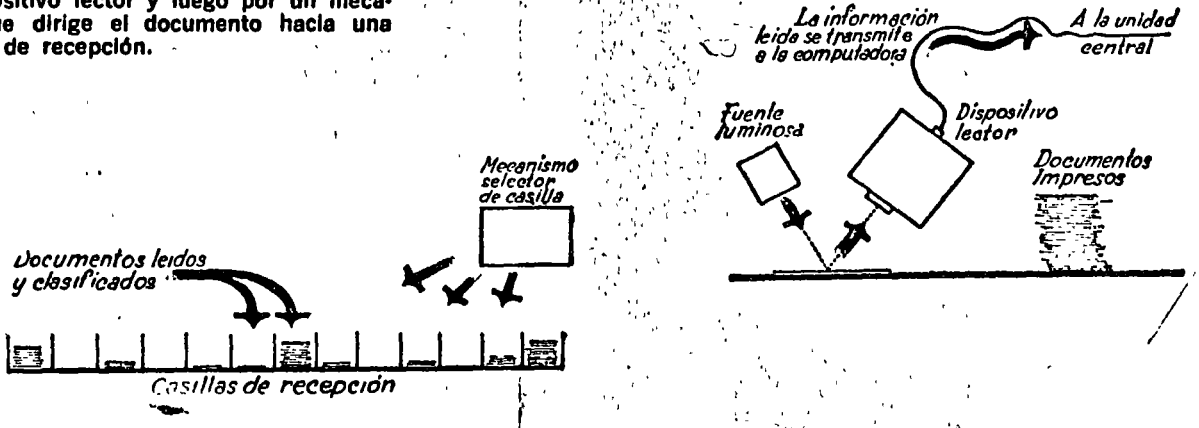


21 Esta "Lectora Optica de Caracteres" actúa como unidad de entrada de datos de una computadora y se acopla directamente a la Unidad Central de Procesamiento de la misma. Puede "leer" hasta 420 documentos impresos por minuto, y esto significa la lectura automática de 30.000 caracteres por minuto. Simultáneamente con la lectura, se pueden clasificar los documentos.

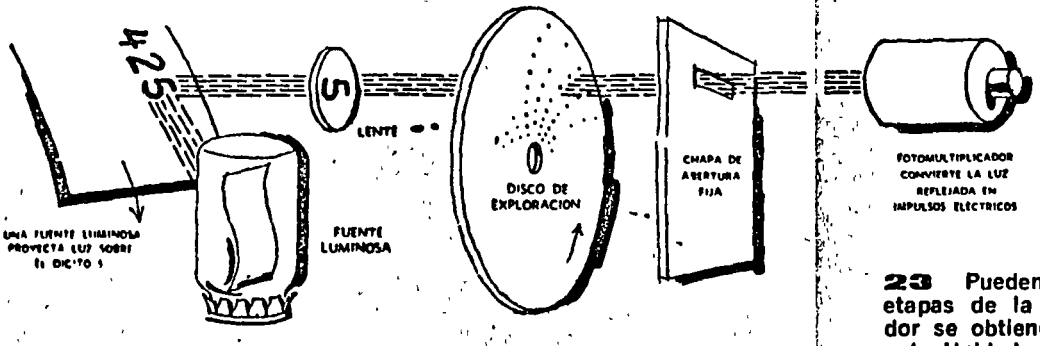
IBM

22 El esquema muestra la marcha de los documentos dentro de la máquina. Cada uno de ellos pasa bajo el dispositivo lector y luego por un mecanismo selector que dirige el documento hacia una de las 13 casillas de recepción.

22

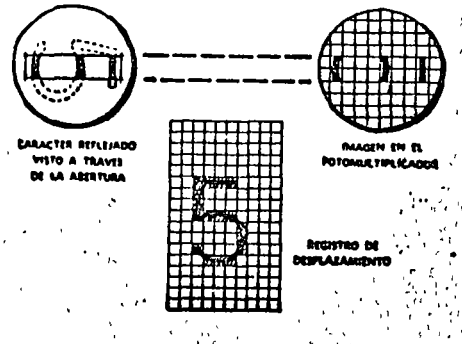


23



23 Pueden observarse en esta figura las distintas etapas de la lectura automática. Del fotomultiplicador se obtienen los impulsos eléctricos que llegarán a la Unidad de Control.

24



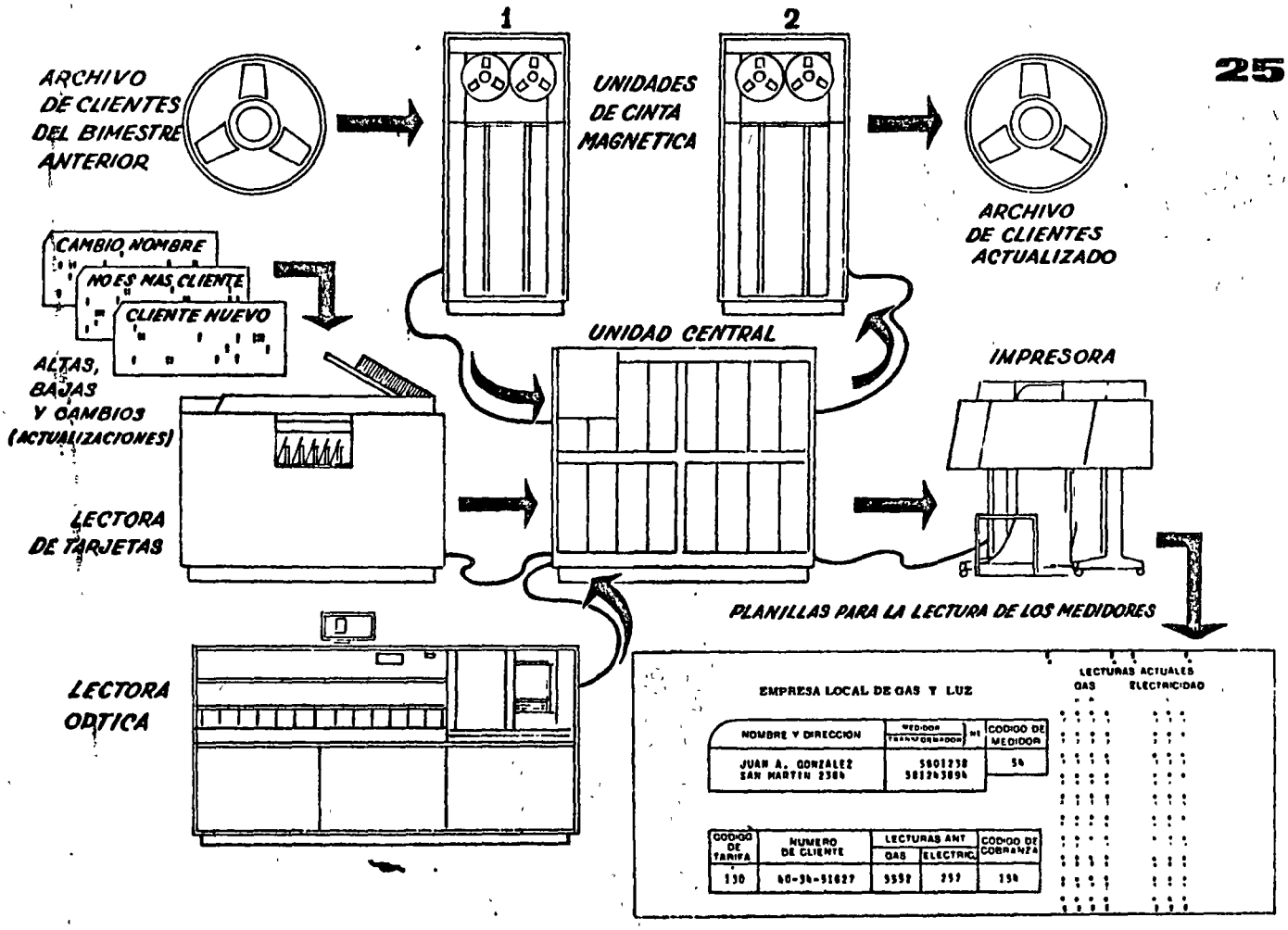
24 A medida que cada carácter impreso se desplaza bajo el dispositivo lector, el fotomultiplicador reconstruye su imagen completa.

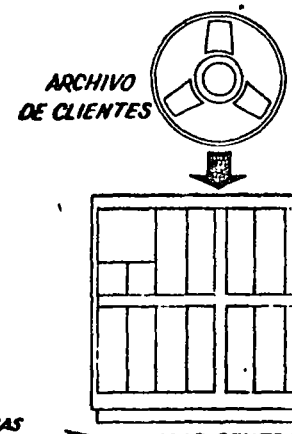
— Existen también "Lectoras Magnéticas" que reconocen la forma de los caracteres magnéticamente. Para ello, los caracteres son impresos con una tinta especial magnetizable.

APLICACION CON LECTORA OPTICA

Describiremos seguidamente una aplicación de la Lectora Optica de Caracteres. Supongamos a tal efecto que una empresa de servicios públicos quiere automatizar la facturación y el control de cobranzas, y cuenta para ello con una computadora cuya configuración se muestra en la figura 25. Los procesos básicos de esta mecanización serían los siguientes:

25 Al finalizar un bimestre se "lee" el Archivo de Clientes que está grabado en una cinta magnética, y simultáneamente se "leen" los cambios que se han producido durante dicho bimestre y que ingresan en tarjetas perforadas. Se obtiene así en la unidad 2 de cinta magnética otro carrete que contiene el archivo actualizado. Al mismo tiempo, se imprime una planilla por cada cliente, y esas planillas se entregan a los empleados que irán con ellas al domicilio de cada cliente, para registrar las cifras que marcan los medidores de gas y electricidad.


1
2
ARCHIVO DE CLIENTES DEL BIMESTRE ANTERIOR
UNIDADES DE CINTA MAGNETICA
ARCHIVO DE CLIENTES ACTUALIZADO
CAMBIO NOMBRE
NO ES MAS CLIENTE
CLIENTE NUEVO
ALTAS, BAJAS Y CAMBIOS (ACTUALIZACIONES)
UNIDAD CENTRAL
IMPRESORA
LECTORA DE TARJETAS
PLANILLAS PARA LA LECTURA DE LOS MEDIDORES
LECTORA OPTICA
EMPRESA LOCAL DE GAS Y LUZ
LECTURAS ACTUALES
GAS
ELECTRICIDAD
NOMBRE Y DIRECCION
MEDIDOR TRANSFORMADOR
Nº
CODIGO DE MEDIDOR
JUAN A. GONZALEZ
SAN MARTIN 2386
5801238
5812638954
56
CODIGO DE TARIFA
NUMERO DE CLIENTE
LECTURAS ANT
GAS
ELECTRIC.
CODIGO DE COBRANZA
130
40-34-51627
3552
252
156



LECTURA OPTICA DE LAS MARCAS Y DEL Nº DE CLIENTE

EMPRESA LOCAL DE GAS Y LUZ

LECTURAS ACTUALES GAS ELECTRICIDAD

NOMBRE Y DIRECCION	NUMERO TROQUELADO	NS	CODIGO DE MEDIDOR
JUAN A. GONZALEZ SAN MARTIN 2384	5801738 5012453896		56

CODIGO DE TARIFA	NUMERO DE CLIENTE	LECTURAS ANT GAS	LECTURAS ANT ELECTRIC.	CODIGO DE COBRANZA
136	40-34-51627	5552	257	194

MARCAS RECTAS A LAPIZ POR EL TOMADOR

FACTURA PARA LA COBRANZA

EMPRESA LOCAL DE GAS Y LUZ

DEVUELVA ESTA SECCION JUNTO CON EL PAGO

	TIPO DE SERVICIO	LECTURA DE MEDIDOR ACT	LECTURA DE MEDIDOR ANT	MTS CUB O KWH	TOTAL
JUAN A. GONZALEZ SAN MARTIN 2384 BUENOS AIRES	GAS	6030	5552	478	2190
	EL	348	257	910	2275

SERVICIO DESDE	SERVICIO HASTA	NUMERO DE CLIENTE	NETO A PAGAR	
30-8	30-8-66	40-34-51627	$4465	

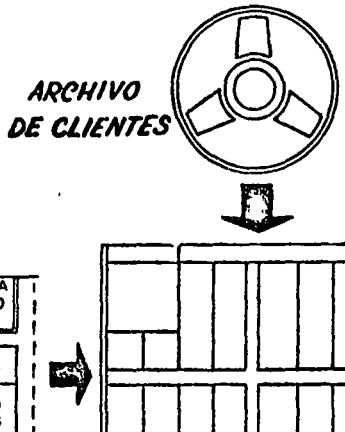
EMPRESA LOCAL DE GAS Y LUZ

RECIBO PARA EL CLIENTE

	TIPO DE SERVICIO	LECTURA DE MEDIDOR ACT	LECTURA DE MEDIDOR ANT	TOTAL
	GAS	6030	5552	2190
	EL	348	257	2275
				$4465.-

SERVICIO AL	NUMERO DE CLIENTE
30-8-66	40-34-51627

26



27

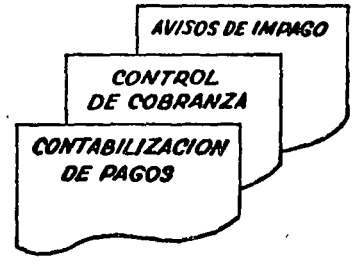
COMPROBANTE DE PAGO
LECTURA OPTICA

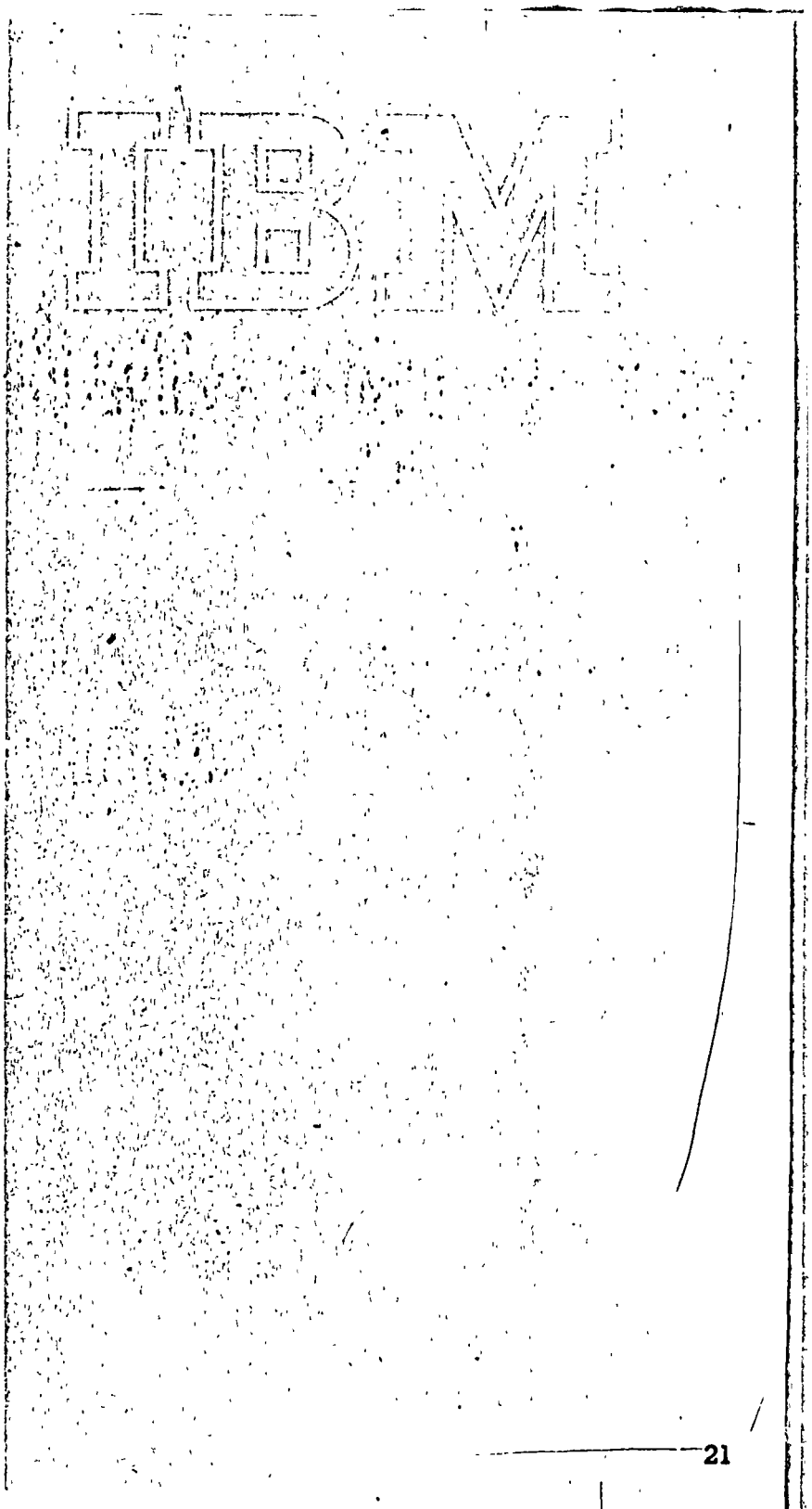
EMPRESA LOCAL DE GAS Y LUZ

DEVUELVA ESTA SECCION JUNTO CON EL PAGO

	TIPO DE SERVICIO	LECTURA DE MEDIDOR ACT	LECTURA DE MEDIDOR ANT	MTS CUB O KWH	TOTAL
JUAN A. GONZALEZ SAN MARTIN 2384 BUENOS AIRES	GAS	6030	5552	478	2190
	EL	348	257	910	2275

SERVICIO DESDE	SERVICIO HASTA	NUMERO DE CLIENTE	NETO A PAGAR	
30-8	30-8-66	40-34-51627	$4465	





26 Los empleados regresan. Con un lápiz han asentado en cada planilla, mediante marcas verticales, el estado de los medidores. La Lectora Optica "lee" las planillas. Simultáneamente se "lee" la cinta magnética actualizada. Se determinan los consumos y los importes. Para cada cliente se elabora una Factura con un talón que servirá de Recibo. Las Facturas-Recibo se entregan a los cobradores.

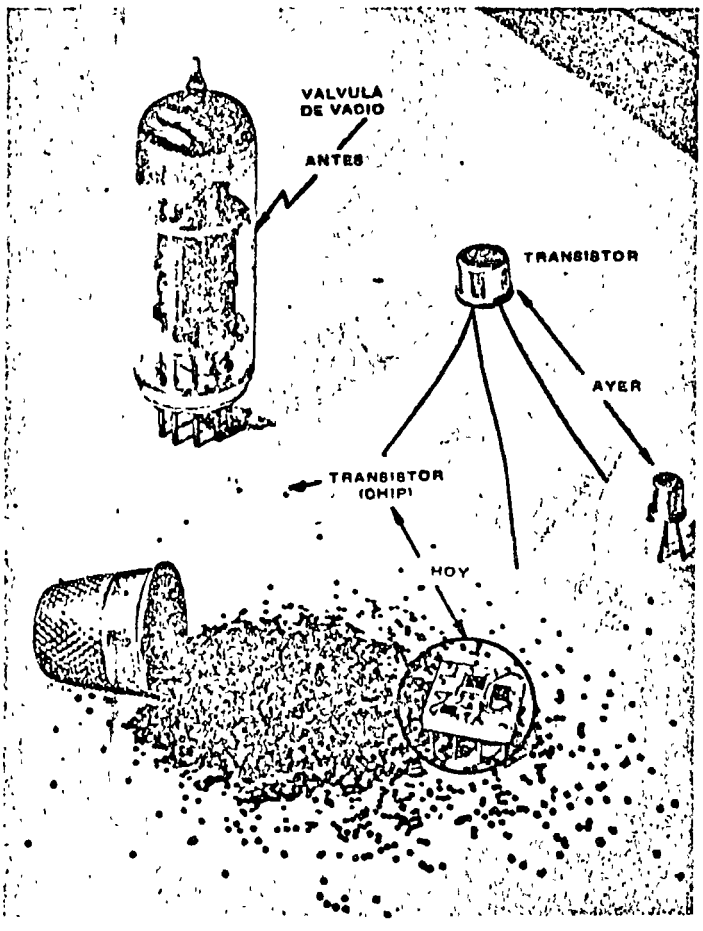
27 Los cobradores regresan y devuelven la sección izquierda de cada Factura-Recibo que ha sido abonada por el cliente. Esa sección, que es un comprobante de pago, es "leído" por la Lectora Optica.

Simultáneamente se "lee" el archivo de clientes.

En esa forma, la computadora realiza la contabilización de los pagos, controla la cobranza, confecciona avisos para reclamar el pago a los clientes que no lo han realizado, y muchos otros informes impresos para control y estadísticas. Estos últimos permitirán incluso a la empresa determinar tendencias en el consumo, y sobre esa base podrá elaborar planes futuros de expansión.

TERCERA GENERACION DE COMPUTADORAS

28



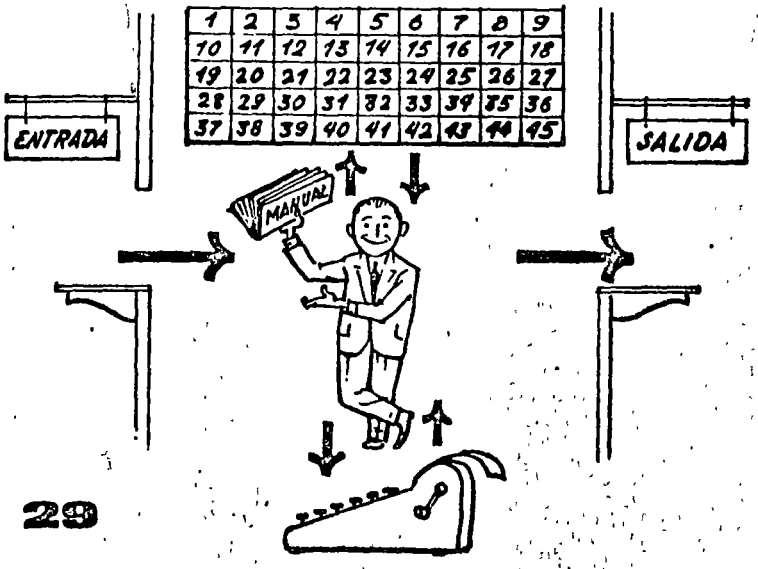
Hemos hablado hasta este momento de la computadora electrónica desde el punto de vista conceptual.

Durante las dos últimas décadas se han producido avances tecnológicos tan extraordinarios en materia de electrónica que la computadora ha sufrido enormes transformaciones. Veremos ahora cómo se ha ido modificando la idea original hasta llegar a los más modernos sistemas de procesamiento de datos.

28 Las primeras computadoras tenían circuitos con válvulas de vacío. Los tiempos de operación se medían en ellas en **milisegundos** (milésimas de segundo). Cuando aparecieron los transistores, el diseño de los circuitos se mejoró notablemente y la duración de las operaciones en las computadoras que utilizaban esta "Tecnología de Estado Sólido" se midió en **microsegundos** (millonésimas de segundo).

El hecho de que las nuevas máquinas fueran miles de veces más rápidas que las anteriores trajo aparejada la creación de unidades de entrada, salida y memoria externa mucho más veloces.

La invención de un nuevo tipo de transistor ("chip") provocó una verdadera revolución en los circuitos electrónicos y sus procesos de fabricación; el nuevo elemento es tan pequeño que en un dedal de costura caben más de 50.000 chips. Puede observarse en la figura, marcado con un círculo, un circuito completo basado en esta nueva "Tecnología de Lógica Sólida". Debido a su tamaño, se los denomina circuitos microminiaturizados o microcircuitos. Los tiempos de operación se miden ahora en nanosegundos (milmillonésimas de segundo). Ha nacido en esta forma la **tercera generación de computadoras**, y las altas velocidades alcanzadas posibilitaron un nuevo enfoque en el diseño de los sistemas de procesamiento de datos.

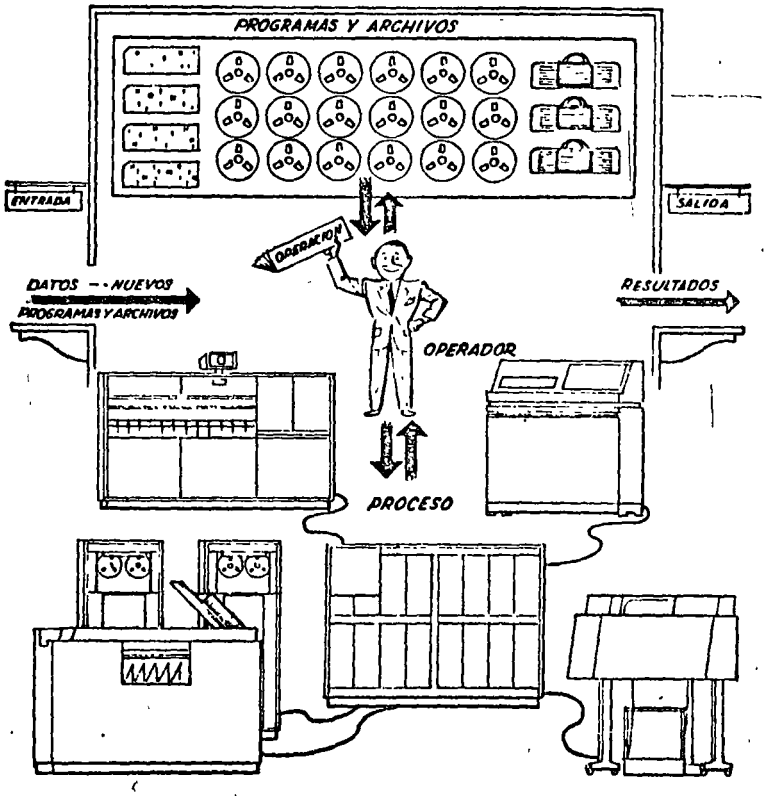


29

29 Volvamos, por un instante, al esquema de computadora que hablamos bosquejado en la figura 5.

Decíamos allí: "Obsérvese que la actuación del señor Control es puramente mecánica: sólo sigue las indicaciones de su manual... toma decisiones, pero solamente cuando se le señalan las alternativas que existen y con qué criterio debe elegir una de ellas".

3D Analicemos ahora una instalación de computadora de la segunda generación y comparemos esta figura con la anterior: el operador que maneja el sistema tiene una biblioteca de programas y archivos en forma de tarjetas perforadas, cintas magnéticas y discos magnéticos. A través de la "Entrada" se le entregan los datos a procesar y los nuevos programas y archivos a incorporar. En un Manual de Operación se indica al operador qué programa y qué archivos debe utilizar en cada proceso de computadora, cómo entran los datos, qué resultados debe entregar como "Salida" y cuáles son las medidas correctivas que debe adoptar cuando se detecta un error en el proceso. También aquí podemos observar que la actuación del operador es puramente mecánica.



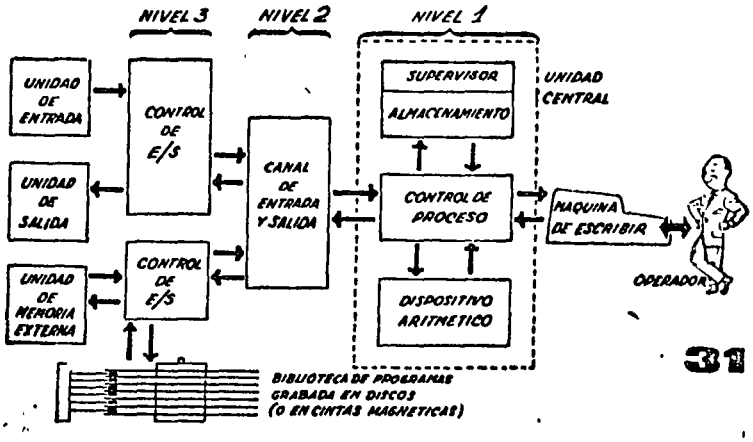
3D

31 Considerando que, en las computadoras de la tercera generación, los tiempos se miden en nanosegundos, se deduce la enorme importancia que asume el tiempo que el operador dedica a la "puesta en máquina" de cada proceso.

En efecto, tan sólo un minuto que se pierda en alistar la máquina significará 60.000.000.000 de nanosegundos perdidos, que equivale, por ejemplo, a 5.000.000 de sumas.

Puede observarse en la figura el esquema de una computadora de la tercera generación. El sistema se opera automáticamente: en una unidad de discos magnéticos (o de cintas magnéticas) está grabada la biblioteca de programas completa. Además hay una zona del almacenamiento destinada a un programa estable llamado "Supervisor", que gobierna todo el sistema a través del control de proceso. Por otra parte, el procesamiento se ha dividido en tres niveles: 1) Unidad Central, donde se realiza todo el proceso de cálculo y lógica; 2) Canales de Entrada/Salida, que manejan y controlan el tráfico de información desde y hasta la Unidad Central; 3) Unidades de Control de Entrada/Salida, que gobiernan y controlan las unidades de Entrada, Salida y Memoria Externa. Los tres niveles operan simultáneamente.

Finalmente, una máquina eléctrica de escribir, conectada al Control de Proceso, sirve como medio de comunicación entre el Operador y el Programa Supervisor. Cuando todo se realiza normalmente, la computadora se autogobierna. Si surge alguna falla en el proceso y el Programa Supervisor no puede superarla, solicita la presencia del Operador a través de la máquina de escribir, e indica además el problema detectado. Si, por otra parte, el operador quiere introducir una modificación en el programa de trabajo de la máquina, se comunica con el Programa Supervisor y así se lo indica.



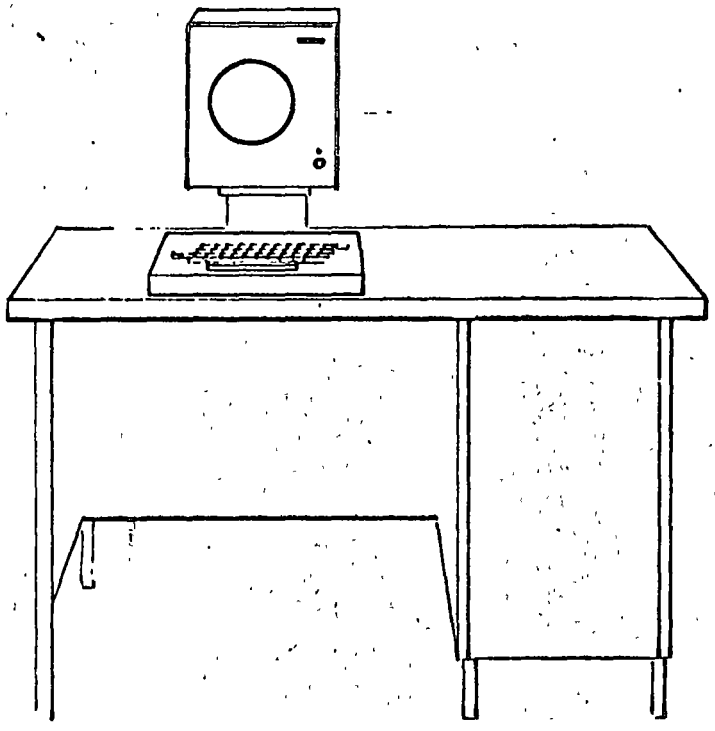
31

Enunciaremos brevemente los adelantos que esta tercera generación ha introducido con respecto a la tecnología anterior:

- La computadora se autogobierna y trabaja sin detenerse, pasando de un trabajo a otro sin demora alguna.
- El Operador interviene sólo cuando algún problema excepcional ocurre. La comunicación entre hombre y máquina se realiza sólo sobre la base de "Informes por Excepción".
- Si ocurre una falla en los circuitos o en la parte electromecánica, la máquina realiza un autodiagnóstico e indica cuál es la anomalía.
- La velocidad de Entrada-Proceso-Salida se ha incrementado extraordinariamente.
- Todas las operaciones del sistema se realizan en forma simultánea.
- Los lenguajes de programación han evolucionado de manera notable.
- El autocontrol y la autoverificación de operaciones han alcanzado niveles insospechados.
- Pueden realizarse, con máximo rendimiento, varios trabajos distintos simultáneamente.

REPRESENTACION VISUAL

La tercera generación de computadoras ha significado un enorme avance en cuanto a potencia de procesamiento de datos se refiere. Pero, tal vez, el adelanto más notable es el de las comunicaciones entre el hombre y la máquina. Veamos dos unidades que permiten dicha comunicación a través de imágenes:

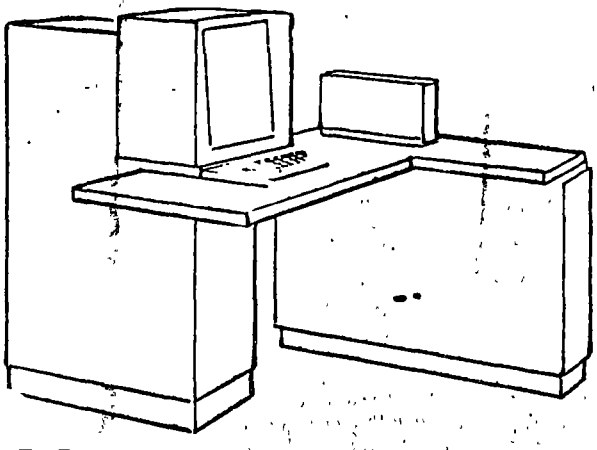


32

32 Esta unidad de entrada/salida sirve para hacer consultas a la computadora, por medio de un teclado de máquina de escribir, y obtener la respuesta reflejada en una pequeña pantalla de televisión. Se la denomina, por ello, "Unidad de Representación Visual".

Pueden ubicarse varias de estas estaciones de consulta en puntos cercanos a la computadora o en sitios remotos, de modo que muchas personas, desde distintos lugares, puedan consultar un archivo magnético simultáneamente. Este tipo de operación recibe el nombre de "Teleprocesamiento de Datos".

La imagen está formada por hasta 12 renglones de hasta 80 caracteres (letras, números o signos especiales) cada uno, es decir que, de una sola vez, pueden representarse 960 caracteres.



33

33 Vemos aquí otra Unidad de Representación Visual, más evolucionada que la anterior: la comunicación hombre-máquina puede establecerse en ella por medio de gráficos, es decir que la entrada y la salida de datos se hacen por medio de imágenes.

Cuenta esta unidad para ello con un dispositivo con forma de lápiz, que tiene en su punta una célula fotoeléctrica. Un delgado haz de luz parte en determinado momento de un punto de la pantalla y la recorre en forma de zig-zag. Si se apoya el "lápiz" en cualquier posición de la pantalla, su célula fotoeléctrica detectará en algún momento el haz de luz. Por el tiempo transcurrido desde que el haz de luz comenzó su "barrido" hasta que fue detectado, la computadora determina en qué punto de la pantalla se encuentra apoyado el "lápiz".

Como el barrido dura una fracción de segundo y se realizan muchos barridos por segundo, se puede "escribir" con el "lápiz" sobre la pantalla, y el dibujo "ingresa" en la memoria de la computadora como una sucesión de puntos codificados.

34 La pantalla está imaginariamente dividida en 1.040.576 puntos, de manera que los trazos que se obtienen son prácticamente continuos.

Pueden dibujarse así curvas, estructuras, letras, números y cualquier tipo de gráfico, y esa información ingresa automáticamente a la computadora.

Por otra parte, los resultados obtenidos por la computadora son representados en la pantalla también como curvas, letras, etc., bajo control del programa almacenado en la memoria.

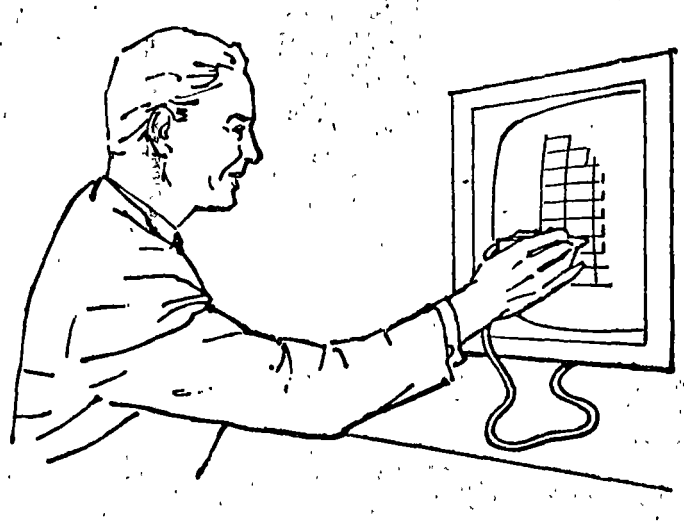
Si preparamos un programa de cálculo de estructuras de hormigón, por ejemplo, un arquitecto podrá bosquejar en la Unidad de Representación Visual la estructura de un edificio, y en pocos minutos tendrá un perfecto plano en escala, con indicación de las medidas de las columnas, vigas y losas, e incluso la cantidad de materiales que insumirá la construcción.

Sobre el mismo dibujo podrá realizar modificaciones y obtener luego los planos y cálculos modificados.

Además, podrá indicar a la computadora que proyecte sobre la pantalla un dibujo de la estructura en perspectiva, y que lo haga girar lentamente ante sus ojos como si fuera un modelo de tres dimensiones.

Esta comunicación visual entre el hombre y la máquina abre nuevos rumbos en el procesamiento de la información. Ya es una ayuda valiosa para la medicina, la automatización de procesos industriales, la ingeniería, la empresa y la investigación espacial.

34



DIGITALIZACION Y SIMULACION

Seguidamente trataremos estos dos conceptos, que han sido la base de increíbles realizaciones en el campo del procesamiento de datos.

Al analizar el funcionamiento de la Unidad de Representación Visual hemos visto que se podía "codificar" un dibujo, gráfico, curva, etc., descomponiendo la imagen en un conjunto suficientemente grande de puntos muy próximos entre sí, y representando luego cada punto por un código. Este proceso, denominado "digitalización de la imagen", brinda las siguientes posibilidades: 1) Almacenar una imagen en la Unidad Central de Procesamiento de la computadora o en cualquier unidad de Memoria Externa. 2) Transmitir una imagen por líneas telefónicas, telegráficas o de televisión, o bien por ondas electromagnéticas, a otros puntos cercanos o remotos. 3) "Procesar" una imagen matemáticamente y depurarla eliminando irregularidades producidas por inexactitud de un trazado a pulso, deficiencias de una placa fotográfica o radiográfica, e incluso interferencias o "ruidos" durante la transmisión de un gráfico a distancia. 4) Representar gráficamente el resultado de un cálculo o proceso.

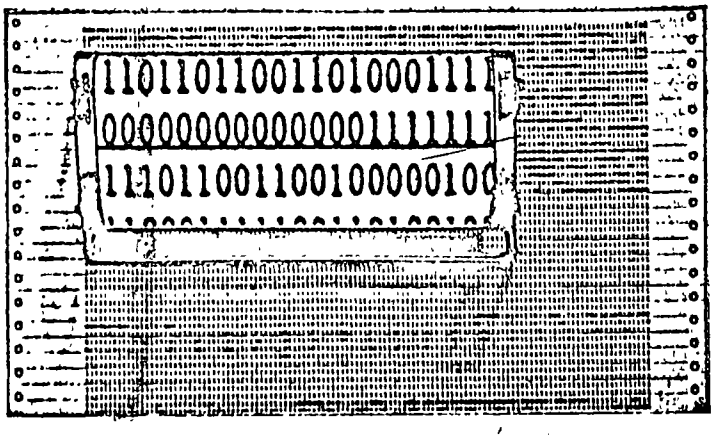
Gracias a la digitalización se han podido transmitir fotografías tomadas al planeta Marte por un vehículo espacial. Una pequeña computadora automática instalada en la cápsula tomó las exposiciones fotográficas y luego "dividió" cada imagen en 40.000 puntos. A continuación codificó cada punto según el color que en él se detectaba. El dispositivo tenía tal sensibilidad que reconocía 62 tonos de gris, además del blanco y el negro.

Se utilizó un código binario:

000000	blanco
000001	62 tonos de gris
000010	
000011	
000100	
000101	
.....	
111110	
111111	negro

Finalmente, el dispositivo transmitió a nuestro planeta el código correspondiente a cada punto. Esto significa que cada fotografía se transformó en 240.000 ceros y unos.

35

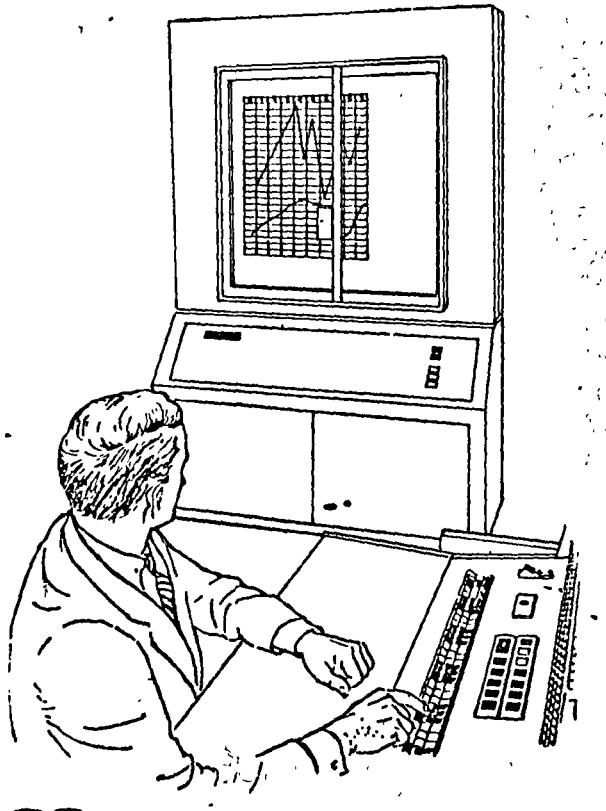


35 Vemos en la figura una de dichas fotos, ya digitalizada, transmitida a Tierra y depurada aquí por una computadora, es decir, lista para ser proyectada en una Unidad de Representación Visual.

Así como en la digitalización se transforman imágenes en códigos numéricos, puede también representarse mediante un adecuado "modelo matemático" un fenómeno físico, químico, económico, demográfico, etc. Esto permite representar un hecho de la vida real matemáticamente, y luego hacer "funcionar" el modelo, también matemáticamente, y en esa forma se puede prever el comportamiento futuro más probable del fenómeno real.

El proceso ha sido denominado "Simulación", y constituye una poderosa herramienta de predicción estadística, de inapreciable valor cuando encaramos un proyecto, pues nos brinda la posibilidad de conocer anticipadamente el efecto más probable de nuestras decisiones.

Por medio de la Digitalización y la Simulación, la computadora ha penetrado en los más intrincados problemas de la ciencia y la empresa.

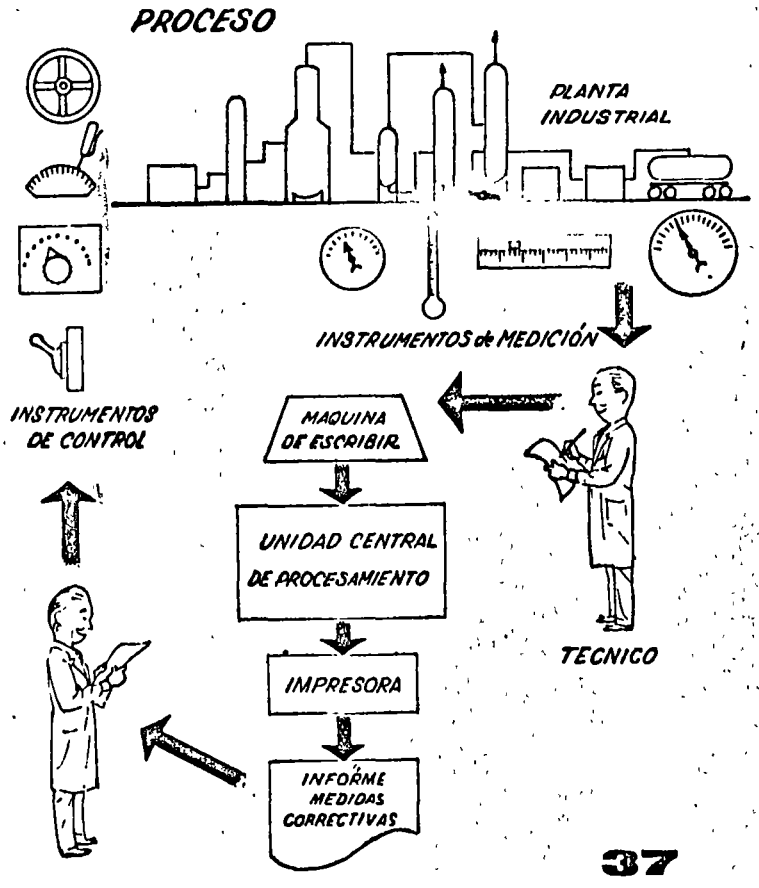


36

36 En la figura se puede observar una Unidad de Salida de datos íntimamente relacionada con esas dos técnicas: se trata de la "Unidad Trazadora de Gráficos", la cual dibuja sobre un plano fijo mediante una pluma comandada por el programa almacenado en la computadora. El funcionamiento de esta unidad se basa en tres movimientos: una guía vertical puede moverse hacia la izquierda o hacia la derecha; un dispositivo inscriptor se desplaza verticalmente a lo largo de la guía; finalmente, la pluma inscriptora se apoya sobre el papel o se separa de éste.

El tablero permite dibujar planos de hasta 74 cms. de lado, y pueden trazarse hasta 10.900 segmentos por minuto. Son muchas las aplicaciones que ya se están llevando a cabo con esta unidad: cartas de tiempo en meteorología, curvas estadísticas, diseño de estructuras o piezas a fabricar, gráficos para diagnóstico en medicina, y en general cualquier trazado que obedezca a una función matemática, o cualquier imagen digitalizada.

Veremos más adelante otras derivaciones de los conceptos analizados.



37

CONTROL AUTOMATICO DE PROCESOS

El gran desarrollo que los métodos de Simulación alcanzaron en los procesos industriales, generalizó el uso de computadoras para simular "En Tiempo Real" dichos procesos, es decir, que el modelo matemático de la planta industrial "funcionaba" en la computadora simultáneamente con el proceso de elaboración real. De esta forma, la computadora evaluaba constantemente, pronosticaba, y aconsejaba medidas correctivas tendientes a optimizar la producción.

37 Imaginemos el presente esquema: una planta de elaboración de productos químicos; una computadora en cuyo Almacenamiento se ha simulado el proceso químico, y un técnico que opera el sistema.

Periódicamente, el técnico anota las indicaciones de los instrumentos que miden presiones, temperaturas, reacciones químicas, velocidad de circulación de líquidos, densidades, etc., e introduce esos datos en la computadora por medio de la máquina de escribir. La computadora procesa entonces dicha información y compara la marcha real del proceso con el modelo en ella simulado. Como resultado entrega, a través de una impresora, un informe en el cual se indica la desviación con respecto a los planes trazados, una evaluación, y las medidas correctivas que deben adoptarse. Finalmente, el técnico realiza las correcciones indicadas, operando para ello los instrumentos de control que actúan directamente sobre el proceso variando presiones, caudales, temperaturas, cantidad de materias primas, etc.

Si bien este sistema de regulación significó un avance extraordinario en la automatización de plantas industriales, la incorporación de dos nuevas unidades a la computadora brindó aún más flexibilidad y velocidad de respuesta al conjunto. La primera de ellas es la "Unidad de Entrada Analógica", a la cual se pueden conectar directamente los instrumentos de medición.

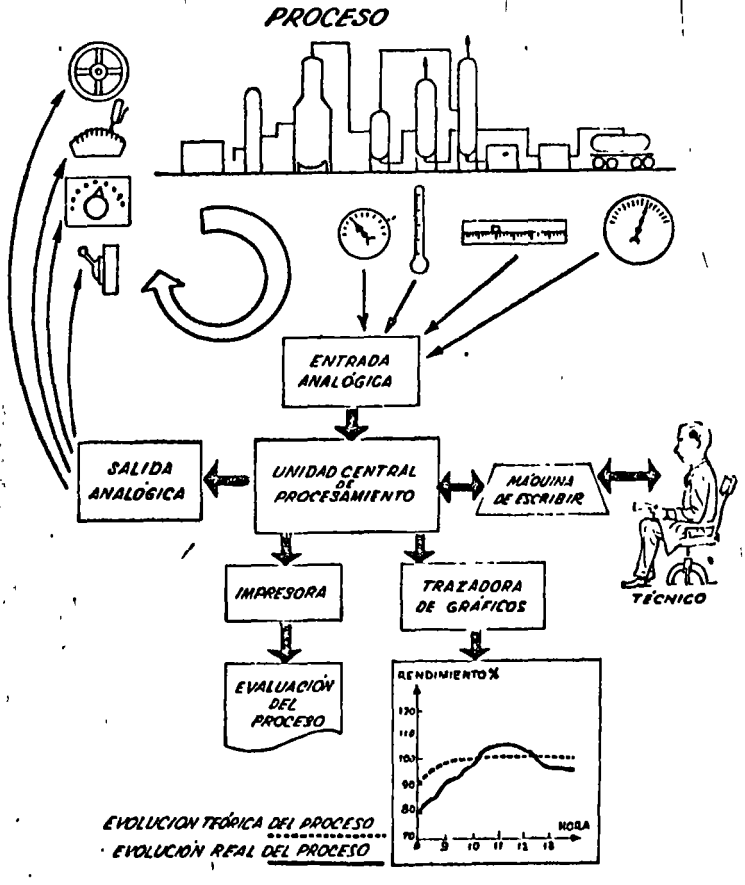
Dentro de la unidad, los impulsos eléctricos provenientes de los instrumentos son transformados en códigos numéricos y transmitidos a la Unidad Central de Procesamiento.

La segunda es la "Unidad de Salida Analógica", en la cual la información sigue un camino inverso al visto para la unidad anterior. Los impulsos eléctricos que ella emite actúan directamente sobre los instrumentos de control del proceso.

38 Vemos aquí el ejemplo anterior, pero con el agregado de las dos unidades descriptas: el sistema se regula automáticamente, y el técnico actúa sólo en casos excepcionales, comunicándose con el sistema a través de una máquina de escribir. Como todos los elementos están directamente comunicados entre sí, cuando algo anormal ocurre, y esa anomalía está prevista, la corrección se realiza casi inmediatamente, y por eso decimos que este conjunto tiene mayor "velocidad de respuesta" que el anterior.

Este nuevo esquema recibe el nombre de "Control de Procesos en Circuito Cerrado", y a continuación veremos, a través de algunos ejemplos, cómo se ha extendido este concepto a las más variadas aplicaciones de computadora.

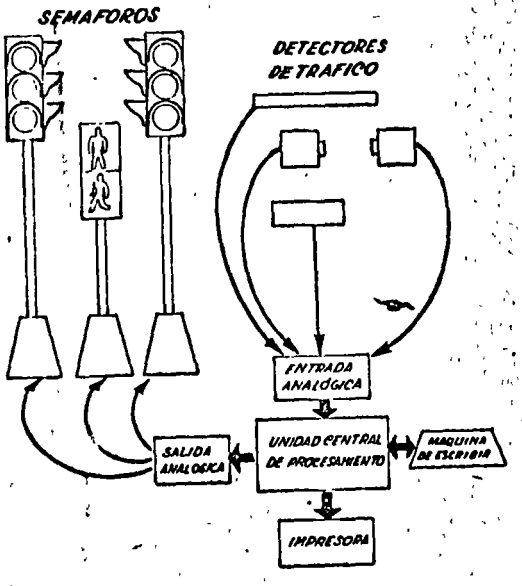
38

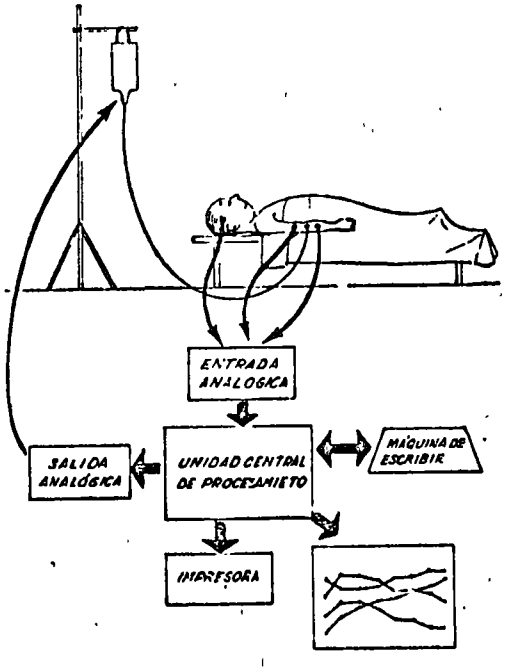


39 Este sistema dirige automáticamente el tránsito en un cruce de calles, una avenida, o bien una gran ciudad. Detectores especiales ubicados estratégicamente en el pavimento "miden" el tráfico. Se los llama "sensitivos", pues al actuar sobre ellos la presión ejercida por los neumáticos, pueden determinar la cantidad de vehiculos que circulan en cada tramo de la red vial, así como también las dimensiones, peso y velocidad de cada uno de ellos. Hay también detectores sónicos, magnéticos, infrarrojos, fotoeléctricos, y sistemas de radar.

Con esa información, la computadora actúa sobre los semáforos regulando el tránsito, mientras imprime informes estadísticos.

39

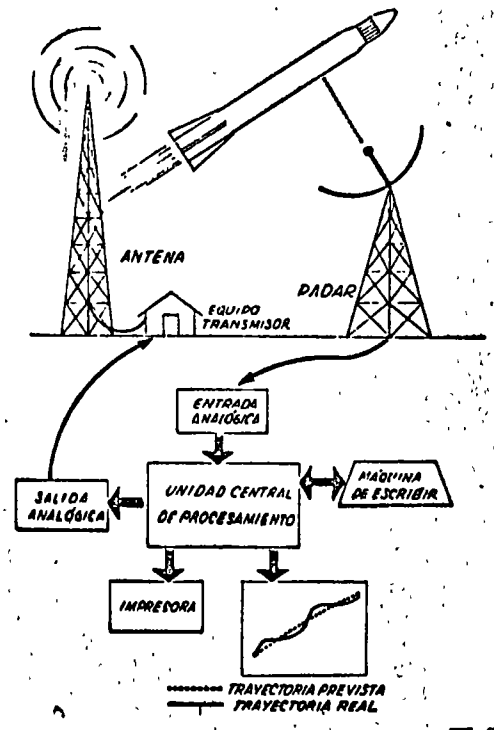




4.0

4.0 También puede regularse automáticamente la anestesia de un paciente durante una intervención quirúrgica: Los datos provienen de los instrumentos que miden presión sanguínea, temperatura, actividad cardíaca, etc., y la salida analógica gobierna el dosaje de las drogas anestésicas. Una computadora "atiende" simultáneamente veinte quirófanos, mientras registra mediante Trazadoras de Gráficos las reacciones de todos los pacientes.

— En una central eléctrica, el sistema permite regular el suministro de energía eléctrica de acuerdo con el consumo real de los usuarios, minimizando así los costos de producción.

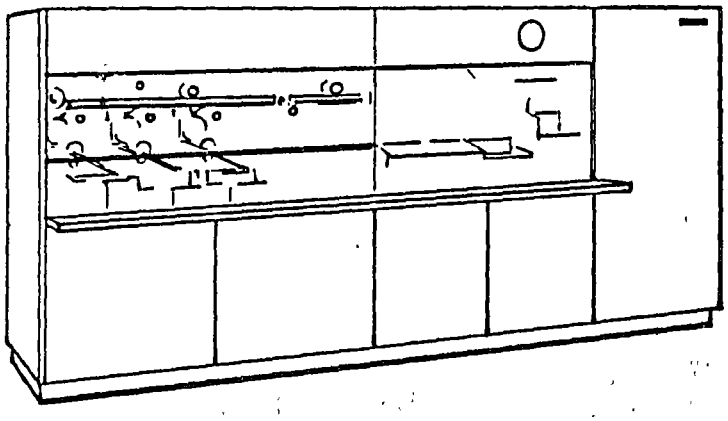


41 Un sistema análogo dirige los lanzamientos de vehículos espaciales: el radar informa en cada instante la posición alcanzada, y ante cualquier desviación detectada la computadora transmite las señales que harán modificar el empuje de los cohetes, corrigiendo la trayectoria.

En las aplicaciones de Control de Procesos, la computadora confronta constantemente el fenómeno real con el modelo matemático que lo simula. Esto le permite acumular "experiencia" acerca del fenómeno, y basándose en esa experiencia la computadora puede ir perfeccionando el modelo original. Esta forma de "aprendizaje", y la universalidad de posibilidades de aplicación, configuran las características fundamentales que distinguen a la computadora del resto de los dispositivos automáticos.

41-1

LECTORA OPTICA DE MANUSCRITOS



42

La posibilidad de leer automáticamente caracteres escritos a mano, ha permitido un mayor acercamiento entre el hombre y la computadora.

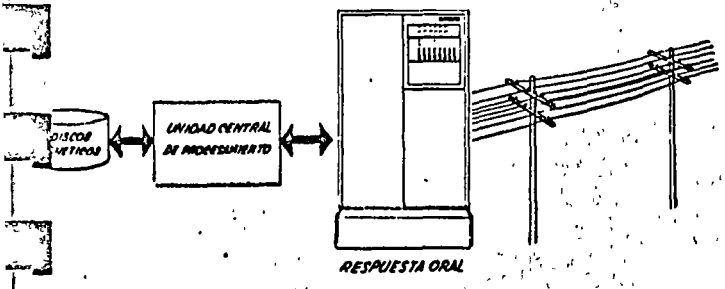
42 Esta Unidad de Entrada de Datos tiene todas las características de la Lectora Optica descripta anteriomente. Pero además, es una Lectora Optica de Manuscritos. Salvo algunas pequeñas restricciones en cuanto al formato de los caracteres, esta unidad puede "leer" documentos escritos por cualquier persona, y con cualquier elemento, a una velocidad de 30.000 caracteres por minuto.

COMUNICACION VERBAL ENTRE EL HOMBRE Y LA MAQUINA

Siguiendo con las posibilidades de comunicación entre el hombre y la computadora, veremos a continuación dos unidades que han significado un trascendental avance en dicho intercambio de información.

43 La Unidad de Respuesta Oral permite "hablar" a la computadora en todo el sentido de la palabra. Contiene una cinta magnetofónica en la cual un locutor —o locutora— ha grabado un diccionario de hasta 128 palabras, en cualquier idioma. La unidad puede conectarse a la red telefónica.

43



Supongamos una computadora instalada en un aeropuerto. En una Unidad de Discos Magnéticos se han almacenado las características de todos los vuelos de y hacia el aeropuerto. Hay además una Unidad de Respuesta Oral, cuyo diccionario contiene vocablos relacionados con dicha actividad, conectada a la Unidad Central de Procesamiento, y a la red telefónica.

Si una persona quiere conocer detalles acerca de un vuelo, llama al aeropuerto desde un aparato telefónico cualquiera, y la computadora "atiende" su llamado: un mensaje almacenado en la Unidad de Discos es enviado a la unidad de Respuesta Oral.

Esta interpreta cada palabra del mensaje y la "toma" luego de la cinta magnetofónica, transmitiéndola después por la línea telefónica. Quien ha llamado, escucha una voz que le indica: "Por favor disque usted el número de vuelo". La persona, sin cortar la comunicación disca el número en cuestión, y entonces la computadora busca la información relativa, en la Unidad de Discos, y la Unidad de Respuesta Oral compagina la respuesta y la transmite, como ya hemos visto.

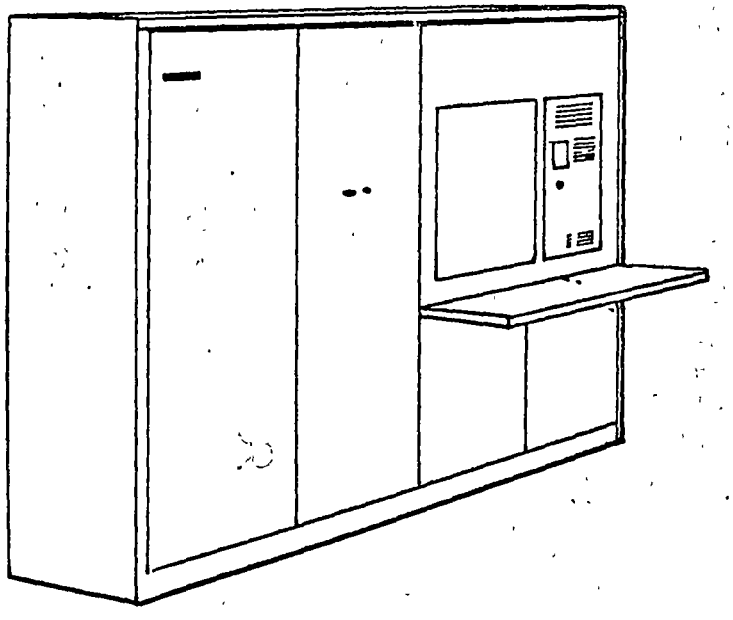
El sistema permite responder 48 llamados distintos automática y simultáneamente.

Hay otro modelo de Unidad de Respuesta Oral, más evolucionado que el anterior, en el cual la voz humana no está grabada sino digitalizada. Esto significa que el vocabulario es ilimitado. La digitalización es tan completa que se puede seleccionar el timbre de voz que se desee.

Existe también una Unidad de Entrada de Datos que recibe Ordenes Verbales, las codifica y las transmite a la Unidad Central de Procesamiento. Mediante esta unidad, se puede dirigir una computadora hablándole directamente por un micrófono. Si bien el dispositivo tiene un incalculable valor científico, en cambio su utilidad práctica es limitada, no habiéndose generalizado su aplicación por tal motivo.

REGISTRADOR/ANALIZADOR FOTOGRAFICO

Cuando se manejan masas de datos muy grandes, hay dos factores que asumen destacada importancia: la velocidad de entrada/salida y acceso a la información, y el volumen que ocupan los archivos. La microfotografía proporcionó una óptima solución al problema.



El Registrador/Analizador Fotográfico es una Unidad de Entrada/Salida de Datos que realiza las siguientes funciones:

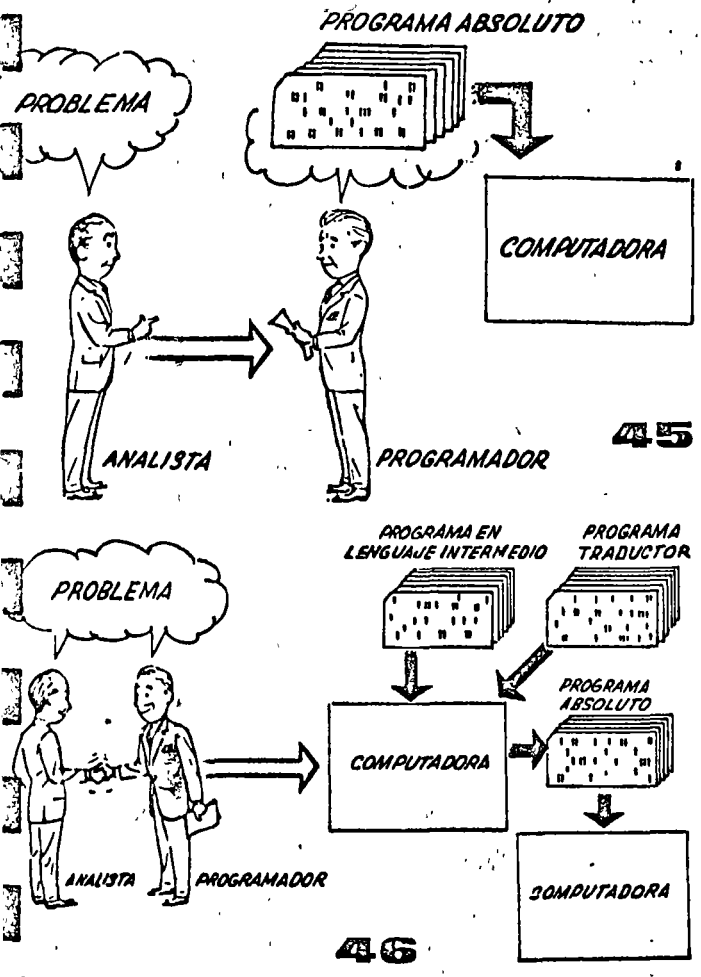
1) Registra los resultados de la computadora sobre microfotografías, mediante un tubo de rayos catódicos, que inciden sobre una película fotográfica, y cuyo haz electrónico actúa gobernado por el Programa Almacenado. La película se revela automáticamente dentro de la unidad, y 48 segundos después está lista para ser proyectada.

2) Proyecta sobre una pantalla translúcida las microfotografías registradas.

3) Analiza imágenes reproducidas en negativo sobre película transparente, las digitaliza y las transmite a la Unidad Central de Procesamiento.

La película utilizada tiene 30,5 milímetros de ancho y 120 metros de longitud. La Entrada o Salida de imágenes puede consistir en letras, números, símbolos, dibujos, gráficos, mapas, curvas, etc. En una microfotografía de 30,5 mm x 30,5 mm pueden registrarse hasta 30.600 letras y números, o hasta 16.777.216 puntos correspondientes a imágenes. La velocidad de Registración/Análisis es de 40.000 letras, números y símbolos por segundo, o su equivalente si se trata de imágenes.

Son innumerables las aplicaciones de esta unidad para cálculo, diseño, representación, diagnóstico, recuperación de la información, e investigación.



PROGRAMACION

Hasta ahora hemos visto muchas unidades que, en distintas combinaciones, configuran computadoras electrónicas para las más variadas aplicaciones. Ahora nos detendremos para analizar el manejo de dichos sistemas.

45 El Programa de Instrucciones almacenado en la Unidad Central de Procesamiento, consta de una secuencia de órdenes y comandos, expresados según una codificación especial denominada "Lenguaje Absoluto de Máquina". Las primeras computadoras se "programaban" en este complejo lenguaje. Habia entonces una enorme diferencia entre nuestro idioma y aquél según el cuál debíamos comunicarnos con la máquina. Esto obligaba a un gran esfuerzo común entre el analista que conocía el problema, y el programador que conocía la computadora, pues ambos hablaban del mismo proceso pero en distintos lenguajes.

46 Se crearon, para solucionar el problema, lenguajes intermedios cada vez más parecidos a nuestro idioma. Es decir que cada nuevo lenguaje intermedio se acercaba más al problema y se alejaba más de la máquina. Para cada uno de estos lenguajes se creó un programa traductor llamado "Compaginador" o "Compilador", que tenía la misión de traducir el lenguaje intermedio al absoluto de máquina. Ahora, el analista y el programador "hablan un mismo idioma": ambos conocen el problema y la solución.

Pero la computadora seguía desarrollándose, y pronto los lenguajes intermedios fueron insuficientes para formular intrincados problemas científicos o comerciales. Nacieron, entonces, lenguajes especializados: dos de ellos, el FORTRAN y el ALGOL, permiten programar problemas científico-técnicos utilizando una notación casi idéntica a la notación matemática común. El COBOL es un lenguaje comercial cuyas sen-

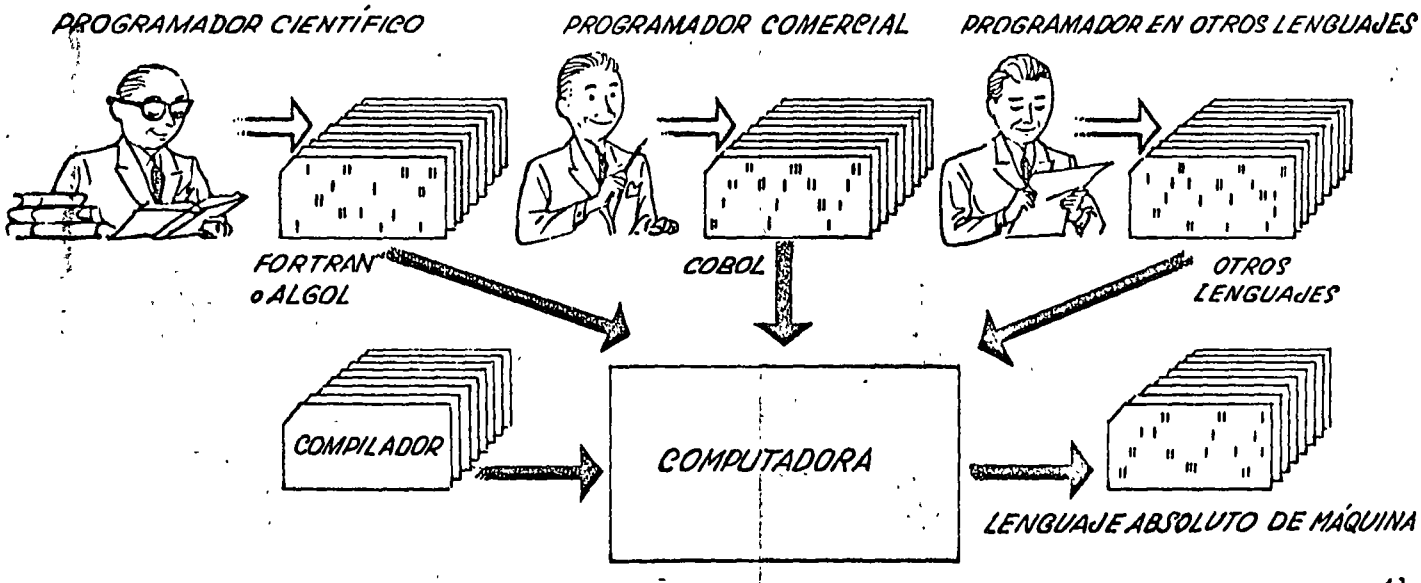
tencias configuran oraciones y frases en forma tal que una persona que no sabe qué es una computadora, puede leer un programa y entender perfectamente qué es lo que hará la máquina cuando lo tenga almacenado.

Cada uno de estos lenguajes tiene un programa Compilador para cada tipo distinto de computadora capaz de procesarlo. Esto significa que un programador que sabe FORTRAN, por ejemplo, puede programar una computadora aún sin conocerla. Es decir que estos tres lenguajes constituyen un "esperanto" de las máquinas.

47 La tercera generación de computadoras permitió abordar complejos problemas que incluían, entre otros, aspectos comerciales y científicos. No había un lenguaje que abarcara todas las especialidades.

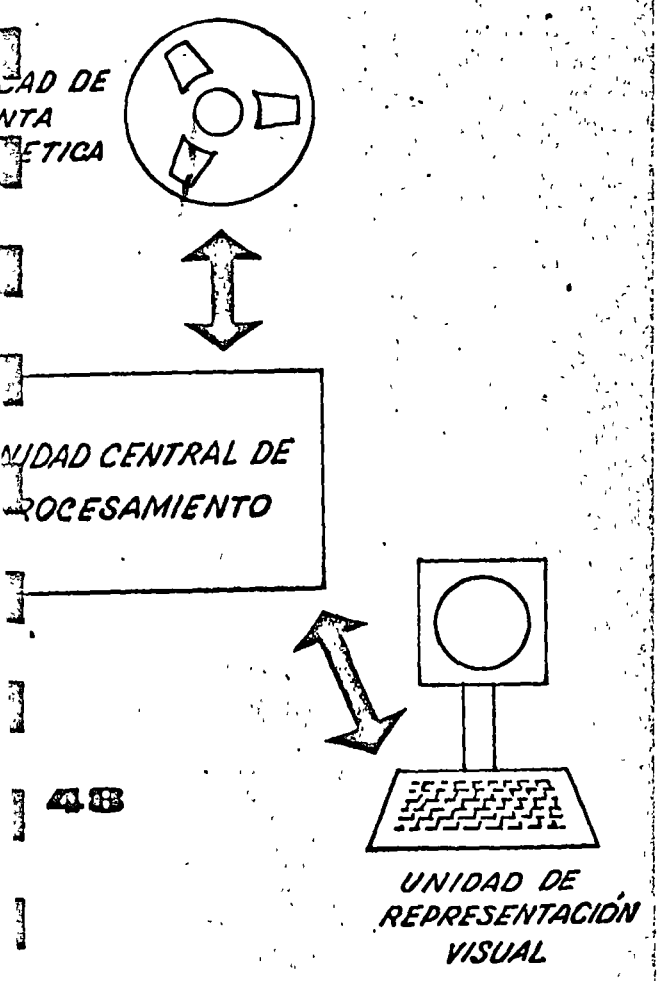
Entonces se reunieron todos los lenguajes conocidos en un superlenguaje llamado PL/I, cuyo compilador es tan poderoso que posibilita la sectorización de la programación en la forma que muestra el dibujo: varios programadores pueden programar distintas partes del proceso, incluso en diferentes lenguajes, y el programa compilador entregará como resultado las instrucciones del proceso completo, en Lenguaje Absoluto de Máquina.

Hemos llegado así a que la computadora nos "entienda", en lugar de que se limite a recibir órdenes en su idioma.



LA MAQUINA DE ENSEÑAR

Cuando analizamos el Control de Procesos mediante computadoras, hemos visto que la computadora puede "aprender". Veremos ahora que también puede "enseñar".



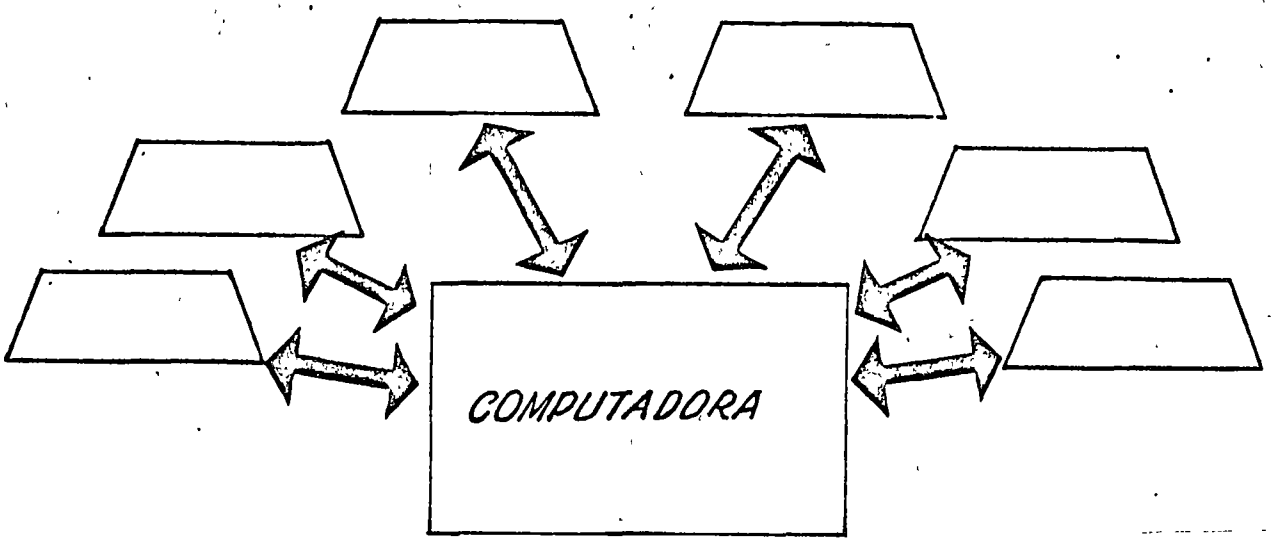
48 En un carrete de cinta magnética —o bien en un juego de discos magnéticos— se ha grabado un texto completo correspondiente a la enseñanza de una determinada materia.

El ordenamiento de todos los tópicos responde a una moderna técnica educativa: la Instrucción Programada. El alumno se ubica frente a una Unidad de Representación Visual en cuya pantalla se va desarrollando el texto, bajo control del Programa almacenado en la Unidad Central de Procesamiento (ver foto en la portada). Periódicamente, la computadora somete al alumno a un pequeño test que aquél responde por medio del teclado o del "lápiz" detector de gráficos. De acuerdo con el resultado de esas pruebas, el programa decide si se debe repasar el punto tratado, si conviene realizar un resumen de todo lo visto hasta ese momento, si el alumno domina ampliamente el tema, en cuyo caso el mismo será salteado, o si corresponde continuar normalmente con el tema.

Se deduce, entonces, que la computadora adecua el ritmo del curso a la velocidad de asimilación del alumno, realizando la enseñanza en tiempo optimo. Simultáneamente con el desarrollo de la materia, el sistema evalúa al estudiante: sus aciertos, sus equivocaciones, su velocidad de lectura, aprendizaje, asimilación y respuesta. Como resultado de esa evaluación, la computadora se comunica con el alumno mediante mensajes que lo orientan con el fin de obtener máximo rendimiento del tiempo utilizado. Al final del curso, el sistema entrega un informe y las calificaciones parciales y finales.

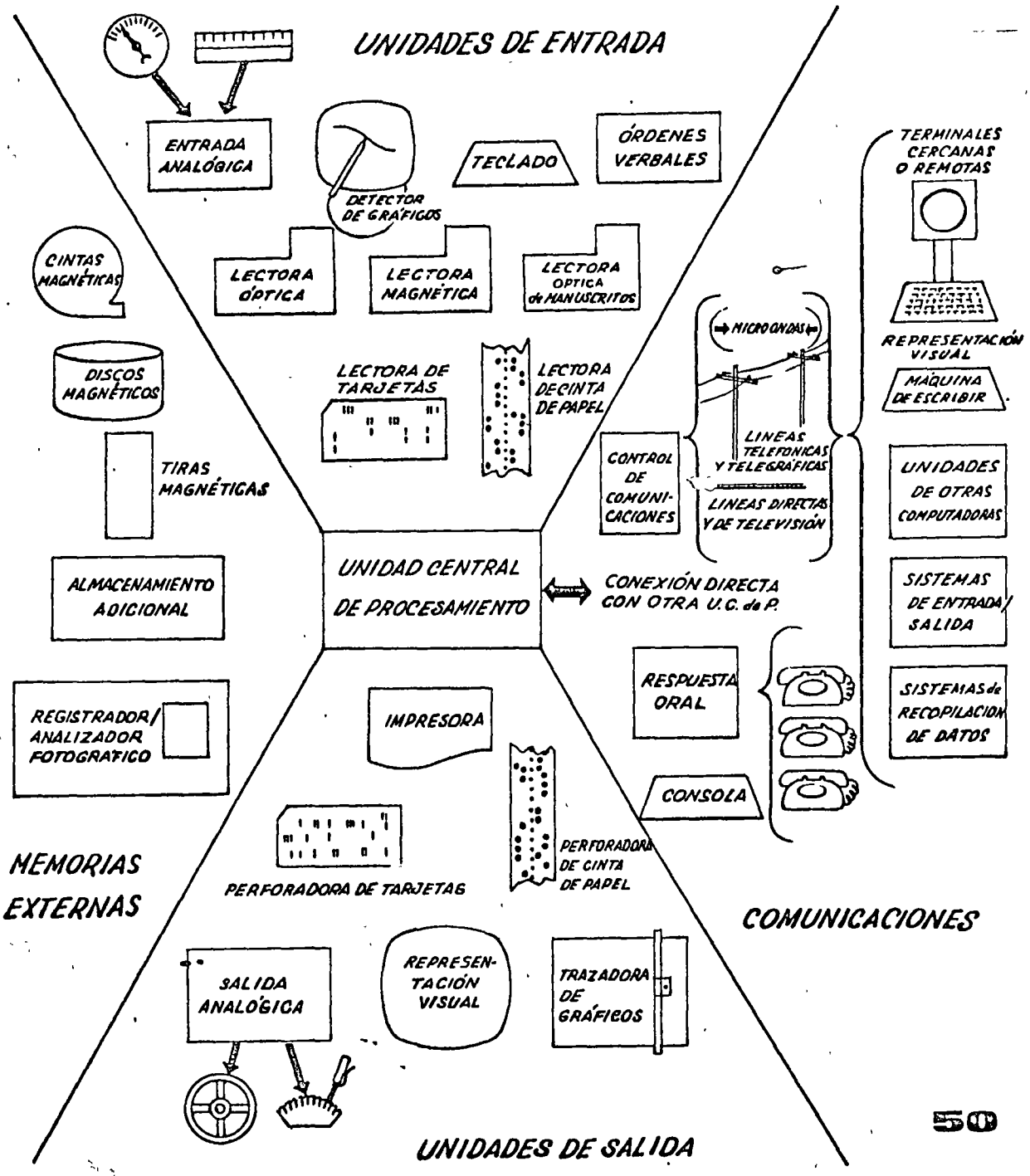
Un solo sistema puede atender simultáneamente a muchos alumnos que sigan iguales o diferentes cursos, tratando a todos ellos individualmente.

TELECOMPUTACION



Cuando una computadora se utiliza como máquina de enseñar, y uno de los alumnos plantea una duda a la computadora, el sistema reacciona modificando el programa de enseñanza en función de dicha consulta. Es decir que el estudiante formula un problema a la máquina, y ésta procesa el problema dando como resultado un diferente programa de estudio. Utilizando un principio análogo, se ha llegado a la siguiente realización:

4º El esquema muestra un sistema formado por una computadora, con su Unidad Central de Procesamiento y sus Unidades de Cintas o Discos Magnéticos conteniendo Programas Compiladores, y un Sistema Supervisor en el Almacenamiento. Hay muchas terminales, cercanas y remotas —en este caso máquinas de escribir— conectadas a la Unidad Central de Procesamiento. Desde cada terminal se puede formular un problema a la computadora, escribiendo el programa por medio del teclado. El sistema realizará la traducción al Lenguaje Absoluto de Máquina, procesará luego el programa, y entregará finalmente, los resultados a través de la máquina de escribir que planteó el problema.
Se pueden procesar varios programas, independientemente uno de otro, en forma simultánea.
Este principio recibe el nombre de "Telecomputación", y posibilita la Descentralización del Procesamiento de Datos en muchos casos, y con enormes ventajas cuando se trata de tareas no repetitivas.


UNIDADES DE ENTRADA
ENTRADA ANALÓGICA
DETECTOR DE GRÁFICOS
TECLADO
ÓRDENES VERBALES
LECTORA ÓPTICA
LECTORA MAGNÉTICA
LECTORA OPTICA de MANUSCRITOS
CINTAS MAGNÉTICAS
DISCOS MAGNÉTICOS
LECTORA DE TARJETAS
LECTORA DE CINTA DE PAPEL
TIRAS MAGNÉTICAS
ALMACENAMIENTO ADICIONAL
UNIDAD CENTRAL DE PROCESAMIENTO
CONEXIÓN DIRECTA CON OTRA U.C. de P.
REGISTRADOR/ ANALIZADOR FOTOGRÁFICO
IMPRESORA
PERFORADORA DE CINTA DE PAPEL
PERFORADORA DE TARJETAS
MEMORIAS EXTERNAS
SALIDA ANALÓGICA
REPRESENTACIÓN VISUAL
TRAZADORA DE GRÁFICOS
UNIDADES DE SALIDA
CONTROL DE COMUNICACIONES
MICROONDAS
LINEAS TELEFÓNICAS Y TELEGRÁFICAS
LINEAS DIRECTAS Y DE TELEVISIÓN
RESPUESTA ORAL
CONSOLA
COMUNICACIONES
TERMINALES CERCANAS O REMOTAS
REPRESENTACIÓN VISUAL
MÁQUINA DE ESCRIBIR
UNIDADES DE OTRAS COMPUTADORAS
SISTEMAS DE ENTRADA/ SALIDA
SISTEMAS de RECOPILACIÓN DE DATOS

RESUMEN

Como breve reseña de todo lo visto, haremos un esquema que muestra todas las unidades que pueden integrar una computadora electrónica.

50 De cada unidad existen numerosos modelos de distintas capacidades, velocidades y otras características. Esto da una idea de la gran cantidad de combinaciones posibles, de lo cual se deduce la notable flexibilidad de la computadora para adaptarse a las más diversas aplicaciones.

Pueden observarse en el gráfico algunas unidades que no han sido analizadas detenidamente, debido a que su funcionamiento se basa en principios análogos a los que rigen la operación de otros dispositivos ya vistos.

EPILOGO

Tal vez se habrá preguntado el lector cómo es posible diseñar, construir y probar una máquina tan compleja como lo es la computadora. El problema es similar, por ejemplo, al de la construcción de una enorme prensa hidráulica. Se fabrican las piezas de la prensa con otras máquinas menores, cuyas piezas a su vez fueron fabricadas por otras máquinas. La primera máquina herramienta fue construida totalmente por las manos del hombre.

Lo mismo ha ocurrido con las computadoras: las primeras fueron diseñadas con papel y lápiz, fabricadas a mano, y probadas por operadores. Las computadoras actuales han sido diseñadas, construidas y probadas automáticamente por otras computadoras.

Si bien muchas de las ideas en las cuales se basan estos sistemas fueron concebidas hace ya mucho tiempo, las increíbles realizaciones que hemos analizado datan de poco más de veinte años. En tan corto plazo, este "amplificador de inteligencia" se ha convertido en parte de la vida del hombre, como inapreciable aliado en todas sus actividades. Incluso, la creación de dispositivos que imitan nuestras funciones, ha llevado a la formulación de teorías que arrojan nueva luz en los más complicados estudios sobre neurología y fisiología, es decir que la computadora nos ha ayudado a conocernos mejor a nosotros mismos.

Los laboratorios siguen investigando nuevas tecnologías para que los más ambiciosos sueños continúen transformándose en realidades. El futuro nos depara aún muchas sorpresas. Tenemos el privilegio de ser testigos de un trascendental momento de la historia.

Este material fue preparado por el Centro Educativo IBM, Argentina.

INDICE

PASOS PARA "RESOLVER UN PROBLEMA" CON COMPUTADORA

a. IDENTIFICACION DEL PROBLEMA Y DEFINICION DE OBJETIVOS

b. DESCRIPCION MATEMATICA

c. ANALISIS NUMERICO

d. PROGRAMACION DE LA COMPUTADORA

   d.1 DIAGRAMA DE BLOQUES

   d.2 CODIFICACION (FORTRAN)

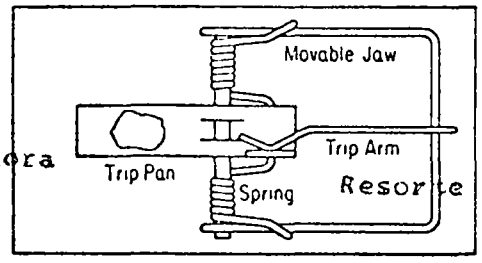
e. COMPROBACION DEL PROGRAMA

f. PRODUCCION

g. INTERPRETACION DE RESULTADOS

PROBLEMA.

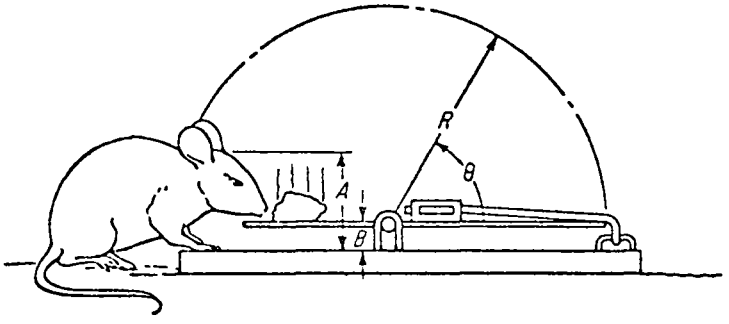
Un fabricante de ratoneras, ha recibido numerosas quejas sobre el comportamiento de las ratoneras que él fabrica. Los clientes se-quejan de que el tiempo que toma la trampa para golpear al animal, es lo suficientemente grande como para permitir que un gran núme-ro de animales escapen. Se desea calcular la constante del re-- sorte para la trampa, que haga posible que el tiempo requerido -- para golpear al animal sea la mitad del tiempo que necesita la -- ratonera actual, manteniendo las características restantes iguales a las de la actual ratonera.



Brazo Móvil

Charola Disparadora

Seguro del disparador



Trampa para ratones en posición abierta

$$\theta = \frac{T_o}{K}\left(1 - \cos\sqrt{\frac{K}{I_o}}\,t\right)$$

$$T_o = T_c + 3.27\,K$$

θ = Desplazamiento angular

T_o = Par del resorte $\theta_o = 0$, lb-ft

K = Constante torsional del resorte lb-ft/rad.

I_o = Momento de inercia lb-ft-sec^2

t = tiempo en segundos.

CONDICIONES DEL PROBLEMA

$A = 1.125$ in $\quad B = 0.5$ in $\quad R = 3.75$ in

$\theta_k = 2.97$ rad $= 170.4°$ (PRIMER CONTACTO)

$\theta_c = 3.27$ rad $= 187.7°$ (POSICION CERRADA)

CARACTERISTICAS DE LA ROTONERA ACTUAL

TIEMPO = 0.0382 SEGUNDOS PARA PRIMER CONTACTO

$T_c = 0.625$ lb-ft (PAR EN LA POSICION CERRADA)

$I_o = 0.0006$ lb-ft-sec^2

CARACTERISTICAS QUE VARIAN PARA LA NUEVA SAT.

$t = 0.0191$ segundos.

$\theta = 2.97$ radianes.

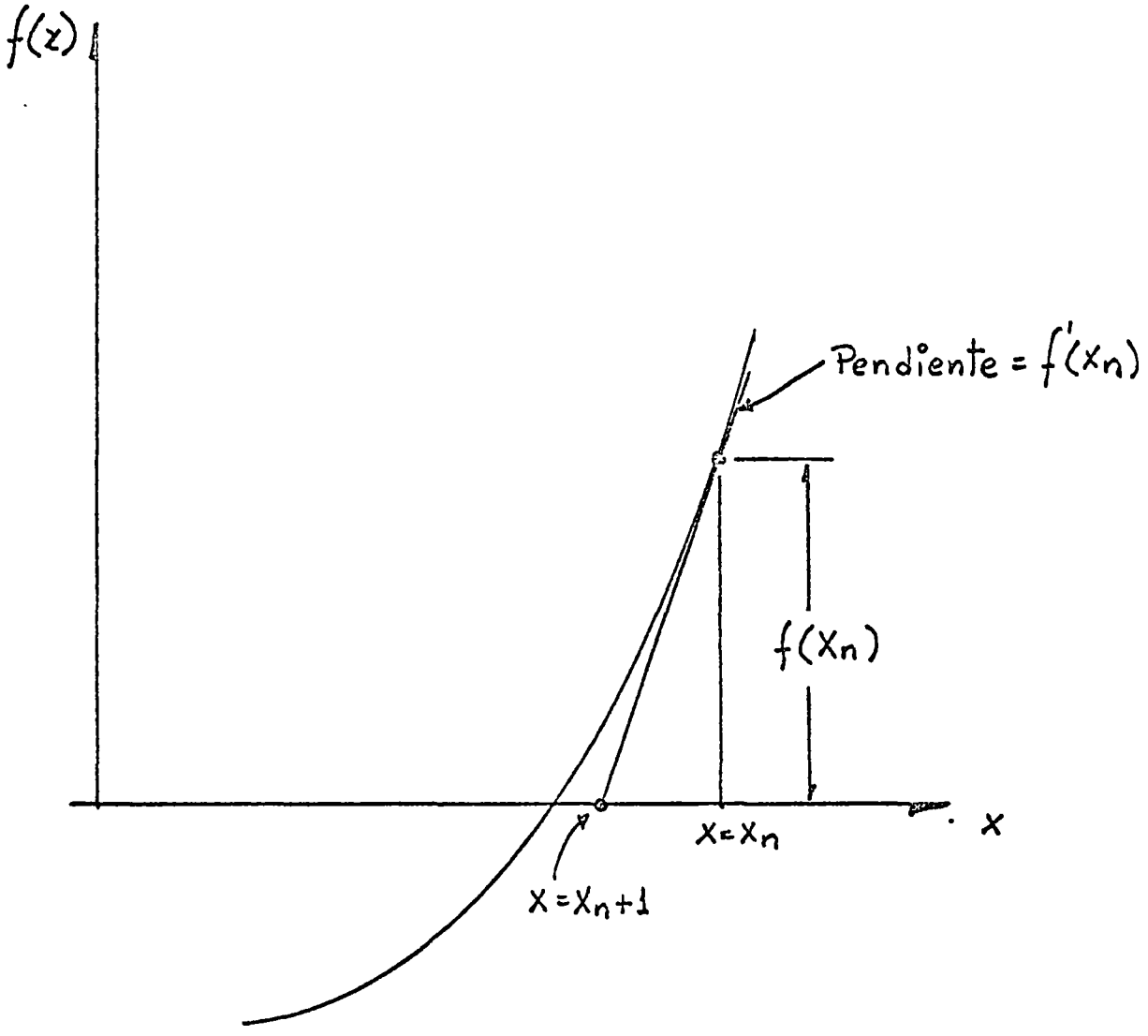
$$\frac{0.625 + 0.30K}{0.625 + 3.27K} - \cos\left(\sqrt{\frac{K}{0.0006}} \times 0.0191\right) = 0$$

$$f(K) = \frac{0.625 + 0.30K}{0.625 + 3.27K} - \cos\left(\sqrt{\frac{K}{0.0006}} \times 0.0191\right)$$

$$f'(K) = \frac{(0.625 + 3.27K) \times 0.3 - (0.625 + 0.3K) \times 3.27}{(0.625 + 3.27K)^2} +$$

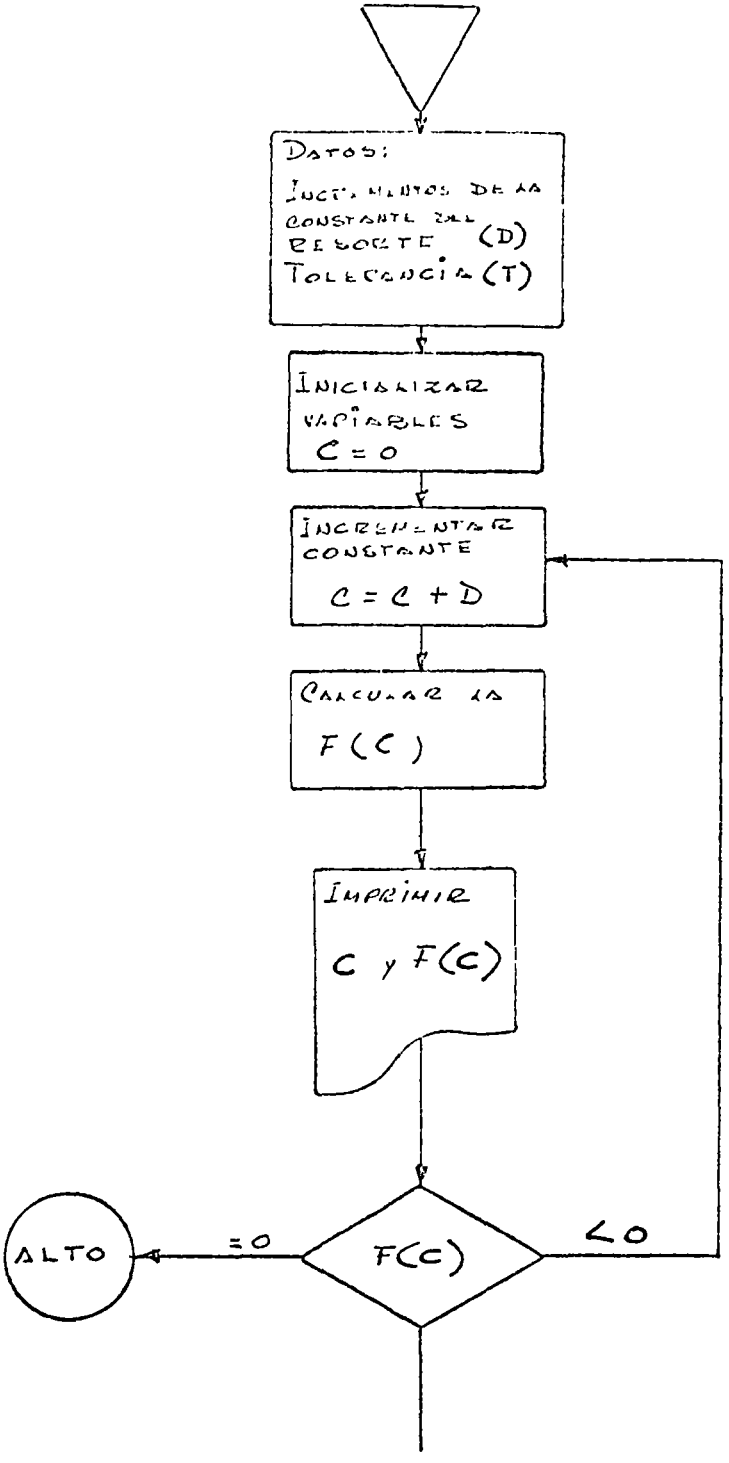
$$\frac{0.0191}{2 \times 0.0006} \times \sqrt{\frac{0.0006}{K}} \times \operatorname{sen}\left(\sqrt{\frac{K}{0.0006}} \times 0.0191\right)$$

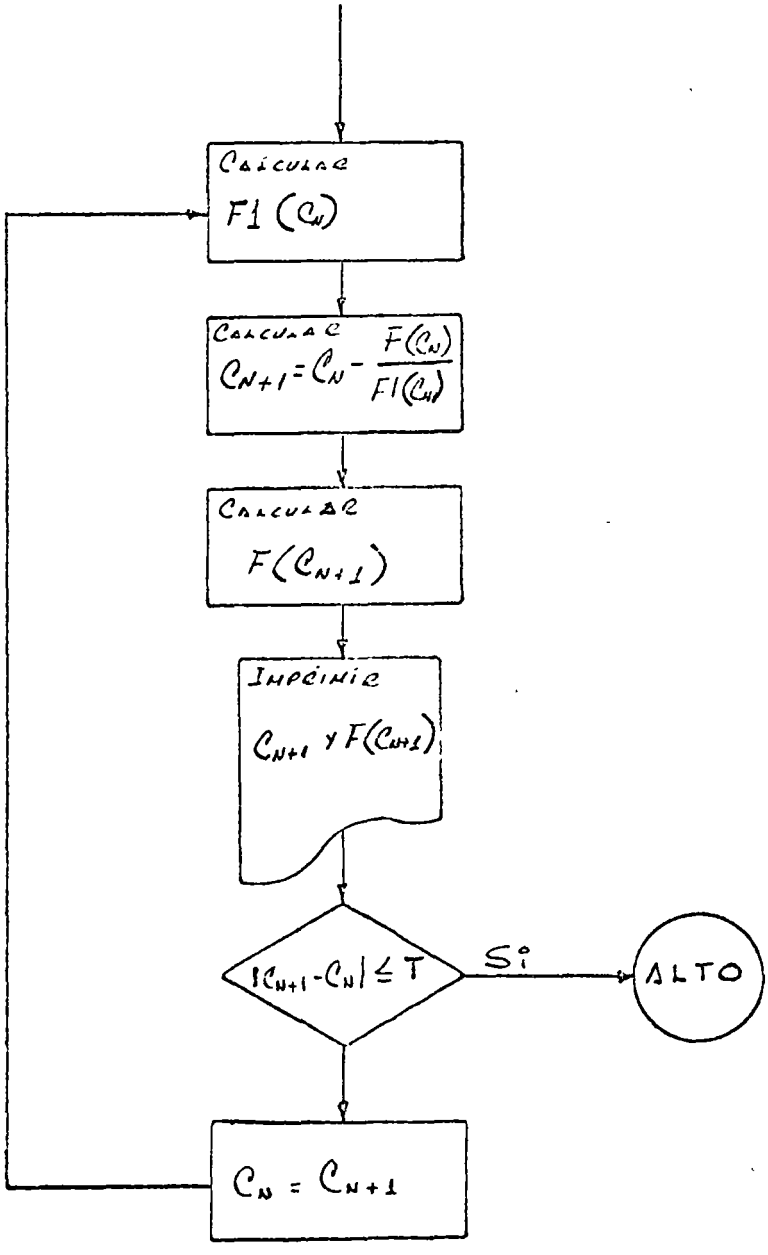
Metodo de Newton



$$f'(X_n) = \frac{f(X_n)}{X_n - X_{n+1}}$$

$$X_{n+1} = X_n - \frac{f(X_n)}{f'(X_n)}$$


DATOS:
INCREMENTOS DE LA CONSTANTE DEL RESORTE (D)
TOLERANCIA (T)
INICIALIZAR VARIABLES
C = 0
INCREMENTAR CONSTANTE
C = C + D
CALCULAR LA
F(C)
IMPRIMIR
C y F(C)
F(C)
=0
ALTO
<0

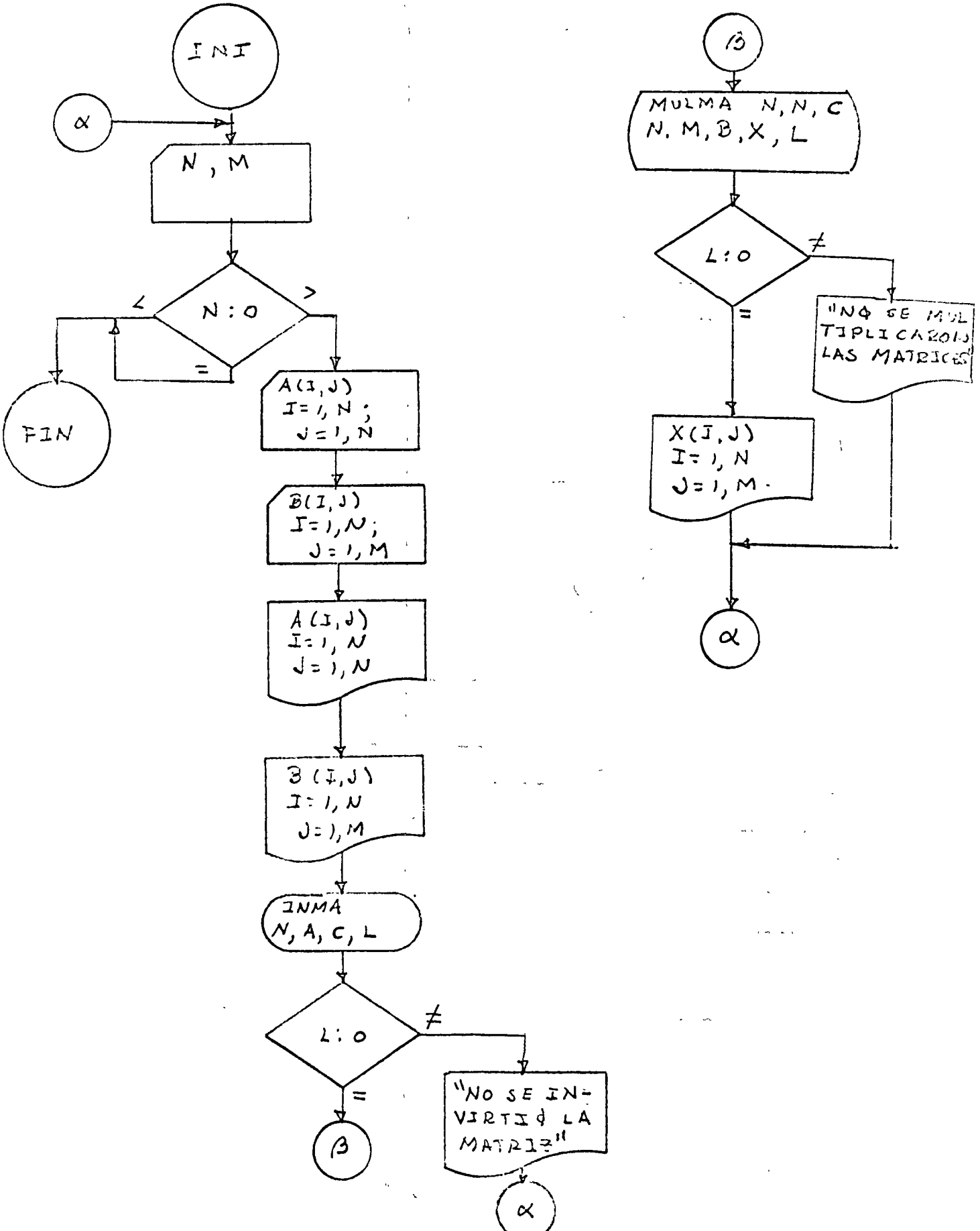

CALCULAR
F1 (CN)
CALCULAR
CN+1 = CN - F(CN) / F1(CN)
CALCULAR
F(CN+1)
IMPRIMIR
CN+1 Y F(CN+1)
|CN+1 - CN| ≤ T
SI
ALTO
CN = CN+1

```
10  READ D,T
20  C=0
30  C=C+D
40  F=(.625+.3*C)/(.625+3.27*C)-COS(SQR(C/.0006)*.0191)
50  PRINT C,F
60  IF F<0 THEN 30
70  IF F=0 THEN 180
80  IF F>0 THEN 90
90  A=((.625+3.27*C)*.3-(.625+.3*C)*3.27)/(.625+3.27*C)↑2
100  B=.0191/(2*.0006)*SQR(.0006/C)*SIN(SQR(C/.0006)*.0191)
110  F1=A+B
120  C1=C-F/F1
130  F=(.625+.3*C1)/(.625+3.27*C1)-COS(SQR(C1/.0006)*.0191)
140  PRINT C1,F
150  IF (ABS(C1-C) <= T) THEN 180
155  C=C1
160  GOTO 90
170  DATA .2,.00001
180  END
```

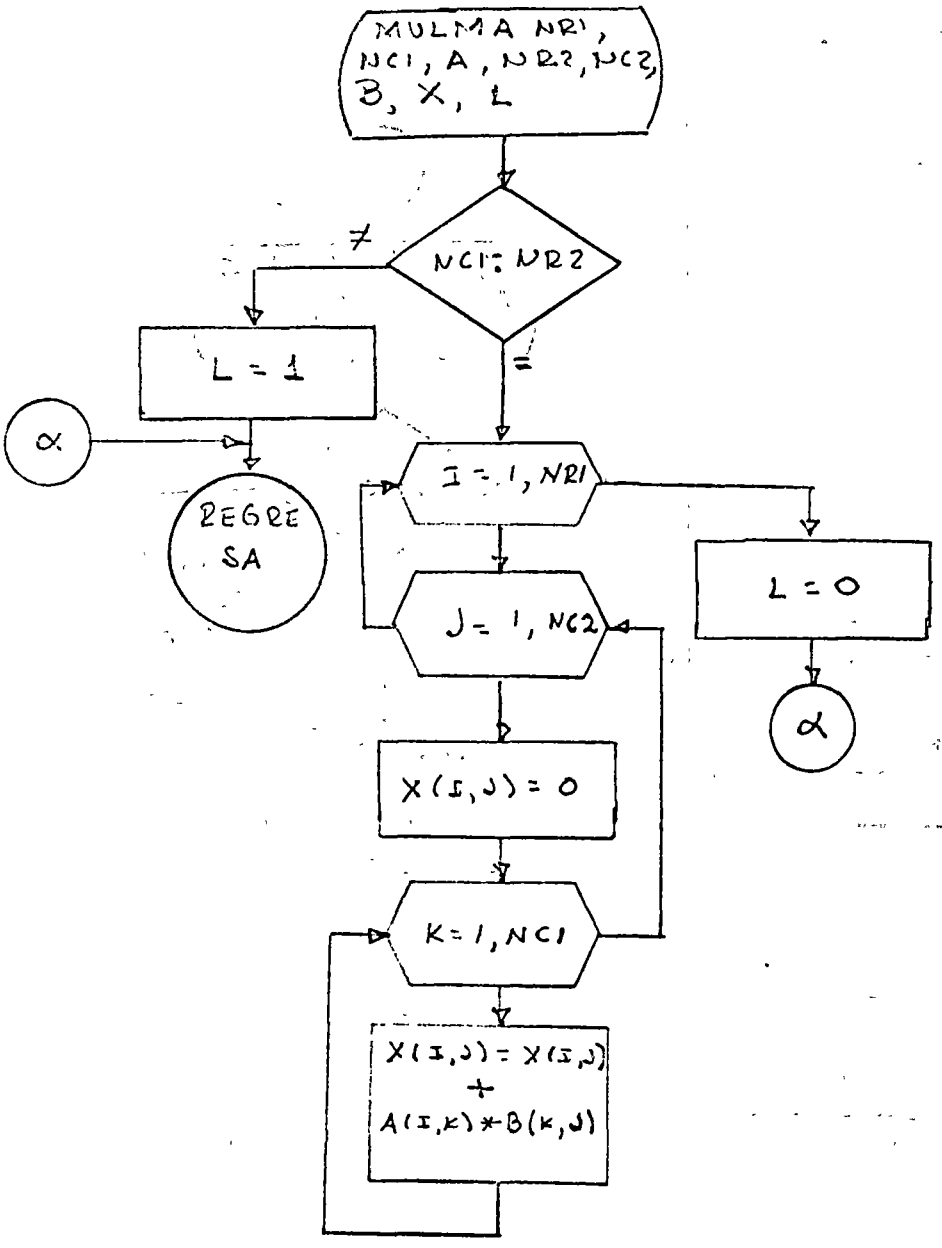
C	F(C)
.2	-.404237
.4	-.49543
.6	-.511902
.8	-.4996
1	-.473602
1.2	-.440307
1.4	-.40289
1.6	-.3631
1.8	-.321979
2	-.280183
2.2	-.238145
2.4	-.19616
2.6	-.154432
2.8	-.113111
3.	-.072304
3.2	-.032089
3.4	7.47430E-03
3.36189	-1.20997E-05
3.36195	2.98023E-08
3.36195	2.98023E-08

SOLUCION DE SISTEMAS DE ECUACIONES

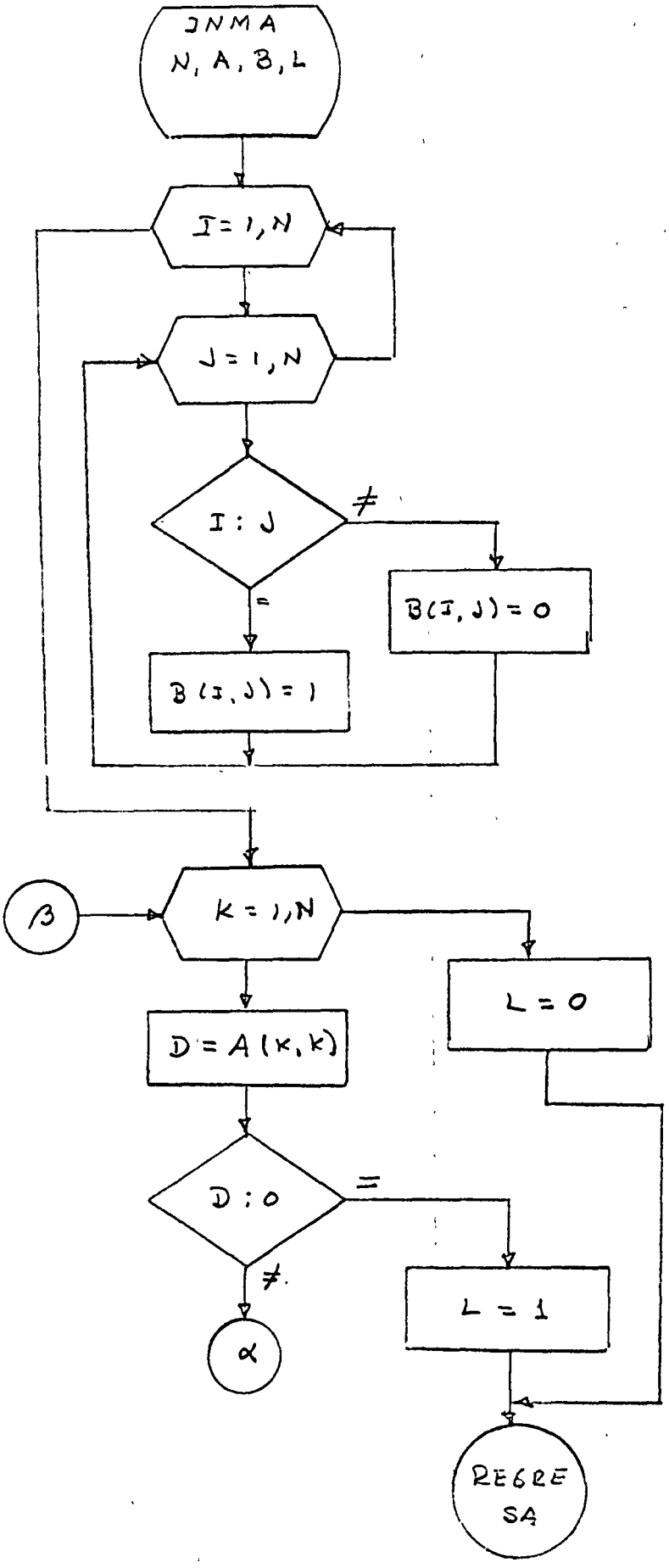
I- PROGRAMA PRINCIPAL.

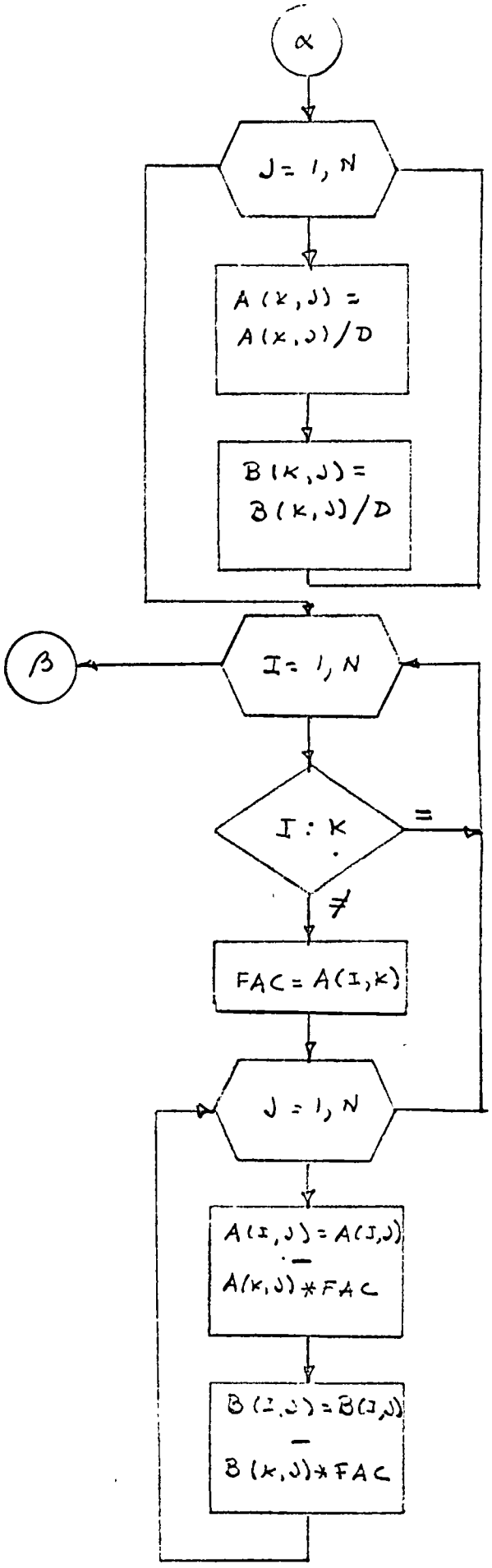


II - Subrutina de Multiplicación de Matrices



III Subrutina de Inversion de matrices.




α
J = 1, N
A(K,J) = A(K,J)/D
B(K,J) = B(K,J)/D
β
I = 1, N
I : K
=
≠
FAC = A(I,K)
J = 1, N
A(I,J) = A(I,J) − A(K,J) * FAC
B(I,J) = B(I,J) − B(K,J) * FAC

```
// JOB SOLUC SISTEMAS DE ECUACIONES
// OPTION LINK
 ACTION NOMAP
// EXEC FORTRAN
C       CENTRO DE EDUCACION CONTINUA DE LA F.I. U.N.A.M.
C       SUBRUTINA PARA MULTIPLICAR MATRICES
        SUBROUTINE MULMA (NR1,NC1,A,NR2,NC2,B,C,L)
        DIMENSION A(10,10),B(10,5),C(10,5)
        IF(NC1-NR2)1,2,1
      1 L=1
        GO TO 10
      2 DO 3 I=1,NR1
        DO 3 J=1,NC2
        C(I,J)=0
        DO 3 K=1,NC1
      3 C(I,J)=C(I,J)+A(I,K)*B(K,J)
        L=0
     10 RETURN
        END
/*
```

```
// EXEC FORTRAN
C       SUBRUTINA PARA INVERTIR MATRICAS
        SUBROUTINE INMA (N,A,B,L)
        DIMENSION A(10,10),B(10,10)
        DO 1 I=1,N
        DO 1 J=1,N
        IF(I-J)2,3,2
     2  B(I,J)=0
        GO TO 1
     3  B(I,J)=1
     1  CONTINUE
        DO 8 K=1,N
        D=A(K,K)
        IF(D)4,11,4
     4  DO 5 J=1,N
        A(K,J)=A(K,J)/D
     5  B(K,J)=B(K,J)/D
        DO 8 I=1,N
        IF(I-K)6,8,6
     6  FAC=A(I,K)
        DO 7 J=1,N
        A(I,J)=A(I,J)-A(K,J)*FAC
     7  B(I,J)=B(I,J)-B(K,J)*FAC
     8  CONTINUE
        L=0
        GO TO 9
    11  L=1
     9  RETURN
        END
/*
```

```
// EXEC FORTRAN
C      SOLUCION DE SISTEMAS DE ECUACIONES ( PROGRAMA PRINCIPAL )
       DIMENSION A(10,10),B(10,5),C(10,10),X(10,5)
     1 FORMAT(2I2)
     2 FORMAT(10F8.2)
     3 FORMAT(1H1,3X,@MATRIZ DE COEFICIENTES@/)
     4 FORMAT(//3X,@MATRIZ DE TERMINOS INDEPENDIENTES@/)
     5 FORMAT(//3X,@MATRIZ SOLUCION@/)
     6 FORMAT(10(1X,F8.2))
     7 FORMAT(//3X,@NO SE INVIRTIO LA MATRIZ@)
     8 FORMAT(//3X,@NO SE MULTIPLICARON LAS MATRICES@)
    13 READ(1,1)N,M
       IF(N)100,100,9
     9 READ(1,2)((A(I,J),J=1,N),I=1,N)
       READ(1,2)((B(I,J),J=1,M),I=1,N)
       WRITE(3,3)
       DO 10 I=1,N
    10 WRITE(3,6)(A(I,J),J=1,N)
       WRITE(3,4)
       DO 11 I=1,N
    11 WRITE(3,6)(B(I,J),J=1,M)
       CALL INMA (N,A,C,L)
       IF(L)12,14,12
    12 WRITE(3,7)
       GO TO 13
    14 CALL MULMA(N,N,C,N,M,B,X,L)
       IF(L)15,16,15
    15 WRITE(3,8)
       GO TO 13
    16 WRITE(3,5)
       DO 17 I=1,N
    17 WRITE(3,6)(X(I,J),J=1,M)
       GO TO 13
   100 STOP
       END
/*
// EXEC LNKEDT
// EXEC
          D A T O S
 3 1
    2.0    -3.0     1.0     1.0    -1.0     1.0     3.0     5.0    -2.0
   -1.0     2.0     7.0
 6 3
    3.2     4.3     1.0    -3.2     1.2     6.7     1.7     8.2     9.2
    7.2     8.3    -4.1     5.2     6.4     1.3     2.5    -3.2     4.6
    2.3     6.2    -1.5     2.4     7.2     5.3    -4.3     5.2     7.4
   -8.2     1.4     1.6     3.2    -4.1     1.7
    4.5    -2.3     6.4     7.2    -1.6     2.7    -8.4     1.8    -9.5
    1.6     6.4     1.3     2.7     5.3    -8.7     2.5    -3.4
 0
```

MATRIZ DE COEFICIENTES

2.00	-3.00	1.00
1.00	-1.00	1.00
3.00	5.00	-2.00

MATRIZ DE TERMINOS INDEPENDIENTES

-1.00
2.00
7.00

MATRIZ SOLUCION

1.00
2.00
3.00

MATRIZ DE COEFICIENTES

3.20	4.30	1.00	-3.20	1.20	6.70
1.70	8.20	9.20	6.50	7.20	8.30
-4.10	5.20	6.40	1.30	2.50	-3.20
4.60	-1.30	2.30	6.20	-1.50	2.40
7.20	5.30	-4.30	5.20	7.40	2.30
-8.20	1.40	1.60	3.20	-4.10	1.70

MATRIZ DE TERMINOS INDEPENDIENTES

4.50	-2.30	6.40
7.20	-1.60	2.70
-8.40	1.80	-9.50
3.60	1.60	6.40
1.30	2.70	5.30
-8.70	2.50	-3.40

MATRIZ SOLUCION

0.33	0.10	0.73
-1.75	0.62	-0.23
0.48	-0.27	-0.27
-0.35	0.46	0.11
1.27	-0.52	-0.34
1.17	-0.44	0.91

C.E.C
15/V/72.

1.- INTRODUCCION

MODELOS DETERMINISTICOS NO SON ADECUADOS PARA EL DISEÑO NI LA EVALUACIÓN.

METODOS MUY CONSERVADORES DE COMBINACION DE LAS PEORES SITUACIONES.

2.- APLICACIONES DE PROBABILIDAD Y ESTADISTICA.

2.1 ESTABLECIMIENTO DE MUESTREOS PARA CONTROL DE CALIDAD

2.2 DETERMINACION ECONOMICA DE REDUNDANCIA EN COMPONENTES

2.3 UTILIZACION DE DATOS HISTORICOS PARA SELECCIONAR PERSONAL

2.4 DESARROLLO DE MUESTREOS PARA DETECTAR ERRORES DE FACTURACIÓN

2.5 UTILIZAR MEDICIONES DE RASTREO PARA LOCALIZAR OBJETOS.

2.6 RELACIONAR LAS VARIABLES DE UN PROCESO PARA ESTABLECER LIMITES EN LAS ESPECIFICACIONES DE LA PRODUCCIÓN.

2.7 ESTIMAR LAS FLUCTUACIONES EN EL COMPORTAMIENTO DE UN GRAN SISTEMA.

2.8 ESTABLECER ESTRATEGIAS OPTIMAS BAJO CONDICIONES DE INCERTIDUMBRE.

2.9 ENCONTRAR EL NIVEL ADECUADO POR INSTALAR EN UN SISTEMA ELECTRICO.

2.10 COMPARACION DE LAS CONSECUENCIAS DE PERIODOS DE GARANTIA ALTERNATIVOS

2.11 ESTABLECIMIENTO DE PROGRAMAS OPTIMOS DE MANTENIMIENTO Y REMPLAZO DE LOS EQUIPOS.

2.12 CALIBRACION DE INSTRUMENTOS.

2.- Interpretaciones de probabilidad.

2.1 Interpretacion clasica.

Si un evento puede ocurrir en N diferentes modos igualmente posibles y si n de estos modos tienen un atributo A, entonces la probabilidad de ocurrencia de A escrita como $Pr(A)$ se define como n/N.

2.2 Interpretacion empírica de probabilidad.

Si un experimento se lleva a cabo N veces y un atributo particular A ocurre n veces, entonces el límite de n/N, cuando N es suficientemente grande, se define como la probabilidad del evento A, $Pr(A)$.

2.3 Interpretacion subjetiva de probabilidad.

La probabilidad $Pr(A)$ es una medida del grado de confianza que uno tiene en una proposicion especifica A.

3 Teoría de conjuntos.

3.1 Un conjunto es una colección finita o infinita de objetos distintos ó elementos con una o varias características comunes que los distinguen.

Un subconjunto es una parte del conjunto a la que algunas características adicionales hace diferentes a sus elementos del resto del conjunto.

El conjunto que contiene todos los elementos se denomina universo.

El conjunto que no contiene elementos se denomina vacío o nulo.

3.2 Operaciones de conjuntos.

La unión (OR)

$$C_1 = A \cup B$$

La intersección (AND)

$$C_2 = A \cap B$$

El complemento (NOT)

$$C_3 = \bar{A}$$

3.3 Conjuntos mutuamente exclusivos.

4. Probabilidad y teoría de conjuntos.

4.1 En teoría de probabilidades el conjunto de elementos son los resultados de un experimento.

$$Pe(A) = \frac{m(A)}{m(\Omega)}$$

4.2 Dos eventos son independientes si la ocurrencia de uno no altera la probabilidad de ocurrencia del otro.

4.3 Leyes probabilísticas.

4.3.1 $Pe(\bar{A}) = 1 - Pe(A)$

4.3.2 Si A y B son dos eventos independientes, entonces la probabilidad de que ambos sucedan es el producto de sus probabilidades.

$$Pe(A \text{ y } B) = Pe(A \cap B) = Pe(A)\ Pe(B)$$

4.3.3 Si A y B son dos eventos mutuamente exclusivos esto es si $m(A \cap B) = 0$ y $Pe(A \cap B) = 0$ entonces

$$Pe(A \text{ or } B) = Pe(A \cup B) = Pe(A) + Pe(B)$$

4.3.4 Si A y B son dos eventos cualquiera, esto es si $Pe(A \cap B) \neq 0$ entonces la probabilidad de ocurrencia de uno de ellos es

$$Pe(A \text{ and/or } B) = Pe(A \cup B) = Pe(A) + Pe(B) - Pe(A \cap B)$$

Ejemplo pag 17

5.- PROBABILIDAD CONDICIONAL Y TEOREMA DE BAYES.

5.1 LA PROBABILIDAD CONDICIONAL $P_e(B|A)$ DE UN EVENTO B CON RESPECTO A ALGUN OTRO EVENTO A ES LA PROBABILIDAD QUE B OCURRA, DADO QUE A HA OCURRIDO.

$$P_e(B|A) = \frac{P_e(A \cap B)}{P_e(A)}$$

TEOREMA DE BAYES

$$P_e(A_i|B) = P_e(A_i)\frac{P_e(B|A_i)}{\sum_{i=1}^{n} P_e(B|A_i)\,P_e(A_i)} \quad \text{para } i = 1, 2, \ldots n$$

6.- Variables aleatorias y funciones probabilisticas. 6

6.1 Una variable aleatoria es un funcion definida en un espacio muestra.

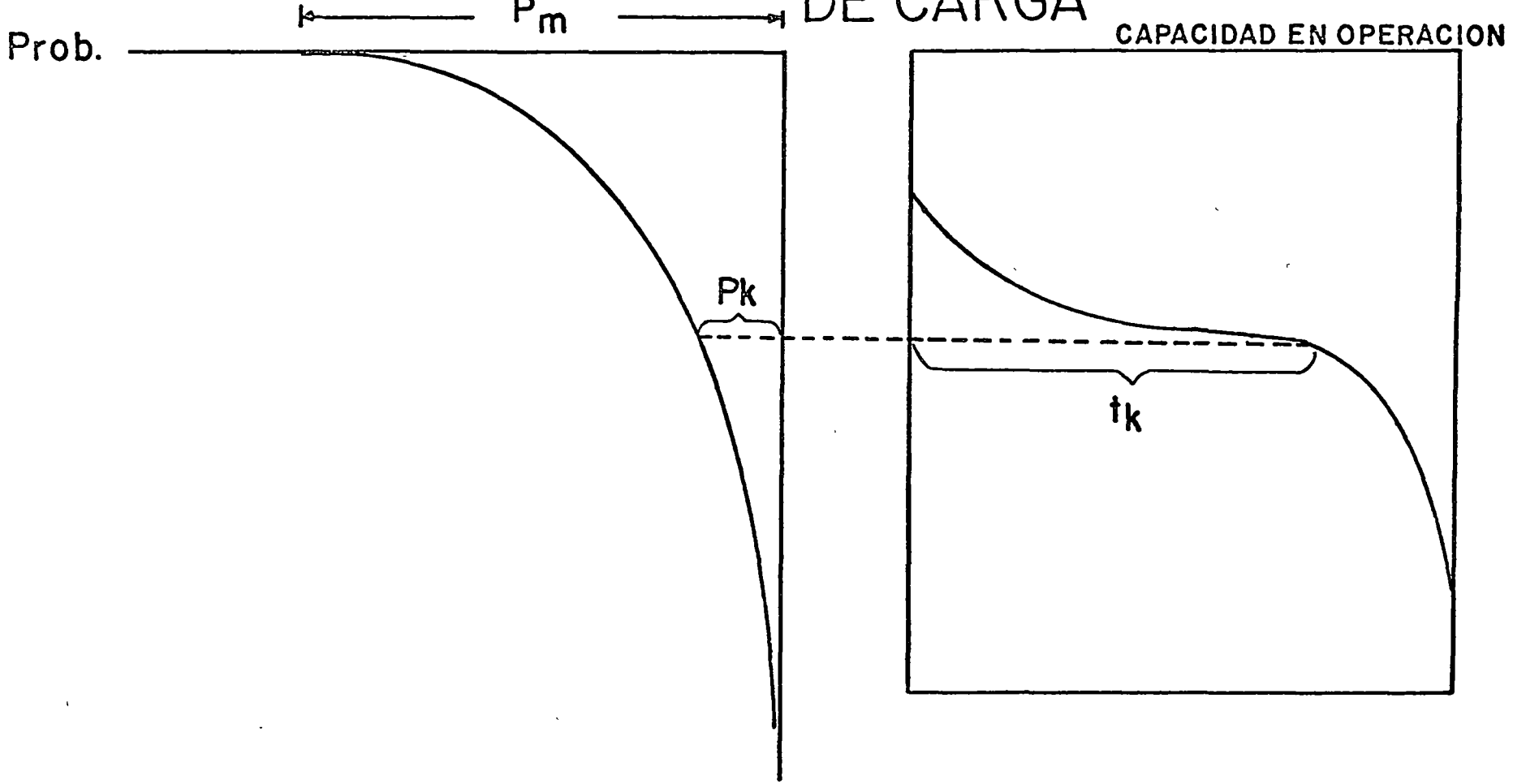
6.2 Una variable aleatoria discreta es una que puede tener solo un número contable de valores.

6.3 La función de probabilidad de una variable aleatoria discreta x es una expresion $p(x_i)$ que da la probabilidad asociada con todos los posibles valores de la variable aleatoria

6.4 Funcion de distribucion acumulativa.

$$P_e(x \leq x_i) = F(x_i) = \sum_{x \leq x_i} p(x_i)$$

CALCULO DE LA PROBABILIDAD DE PERDIDA DE CARGA



$$(PPC)_k = P_k t_k$$

$$\sum_k (PPC)_k = \sum_k P_k t_k$$

8.- Medidas de dispersion

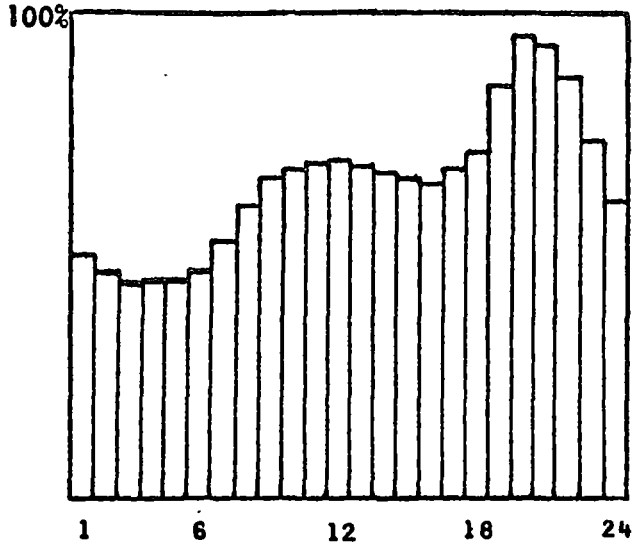
La varianza

$$\mu_2 = Var(x) = E[x-\mu_1]^2 = \begin{cases} \int_{-\infty}^{\infty} (x-\mu_1)^2 f(x)\,dx \\ \sum_i (x_i-\mu_1)^2 p(x_i) \end{cases}$$

Desviacion estandar

$$\sigma = \sqrt{\mu_2} = \sqrt{Var(x)}$$

$$\sigma = \frac{n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2}{n(n-1)}$$



CURVAS TIPICAS DE CARGA
EN % DE LA DEMANDA MAXIMA DEL MES

SISTEMA EJEMPLO

CURVA TIPICA DE LUNES A VIERNES

Fig. 2.2

CURVA TIPICA DE SABADOS

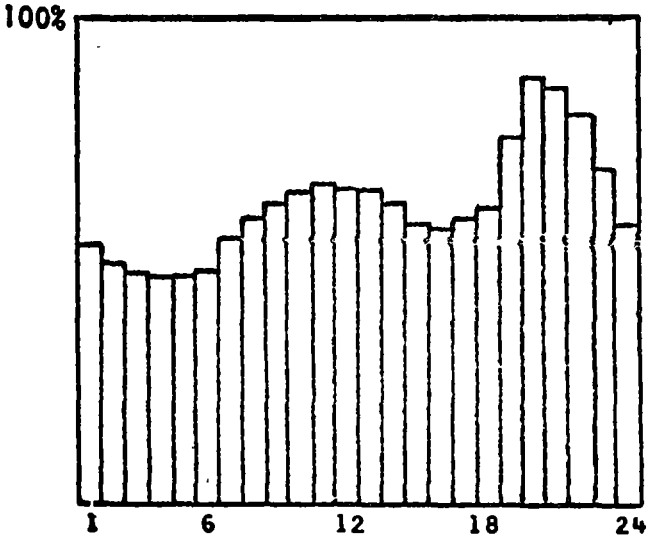
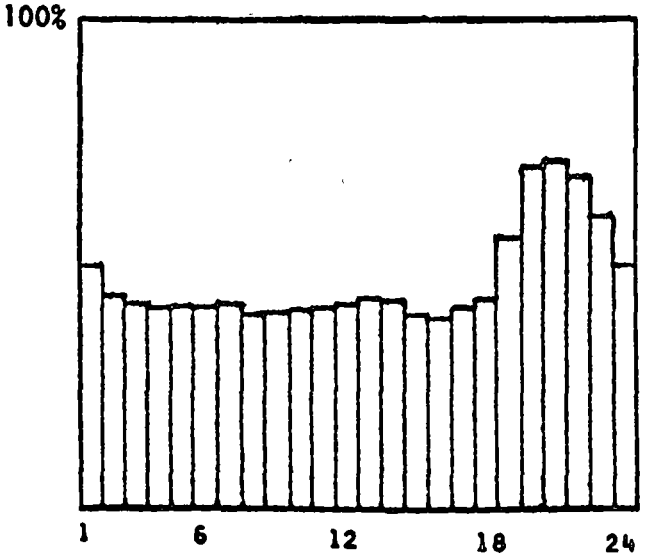


Fig. 2.3



CURVA TIPICA DE DOMINGOS

Fig. 2.4

CURVA DE CARGA HORARIA

SISTEMA EJEMPLO

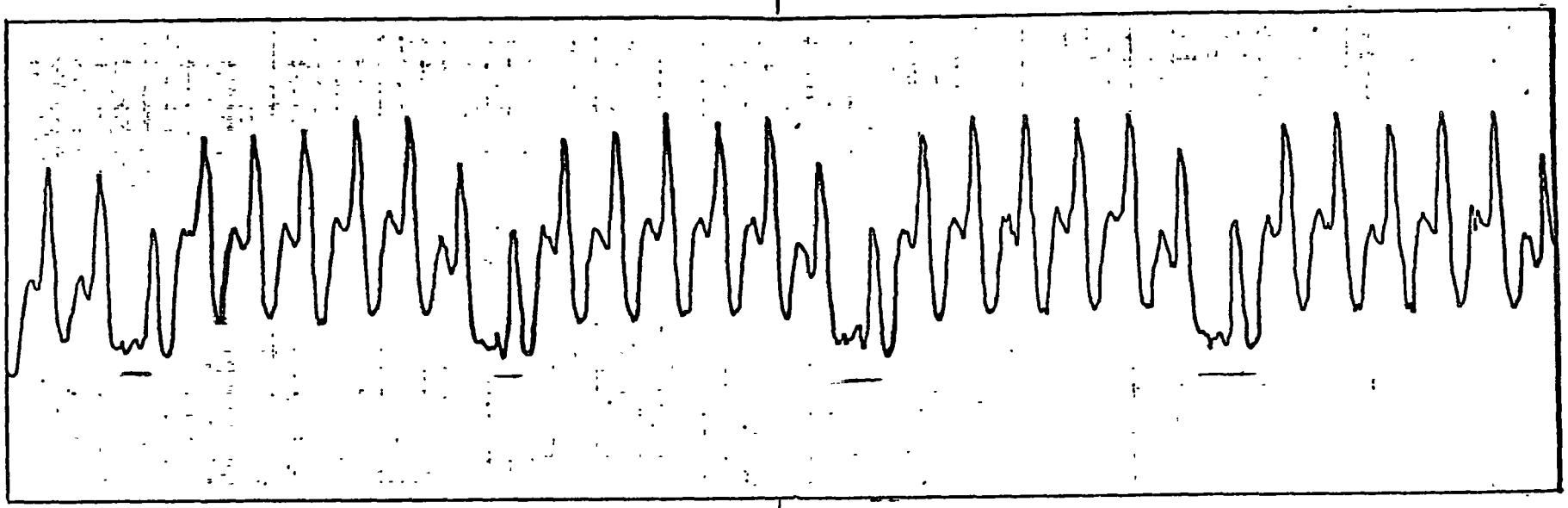


Figura 2.1

UNIDAD	CAPACIDAD EN MW. FUERA.	PROBABILIDAD DE EXISTENCIA DE LA FALLA.
1	0	q_1
	C_1	p_1
2	0	q_2
	C_2	p_2
3	0	q_3
	C_3	p_3
1,2 y 3	0	$q_1\ q_2\ q_3$
	C_1	$p_1\ p_2\ q_3$
	C_2	$q_1\ p_2\ q_3$
	C_3	$q_1\ q_2\ p_3$
	$C_1 + C_2$	$p_1\ p_2\ q_3$
	$C_2 + C_3$	$q_1\ p_2\ p_3$
	$C_1 + C_3$	$p_1\ q_2\ q_3$
	$C_1 + C_2 + C_3$	$p_1\ p_2\ p_3$

Cuando tenemos el sistema compuesto por los generadores 1,2 y 3, vemos de la tabla, que lo único que nos puede ocurrir es que nos fallen.

0
C_1
C_2
C_3
$C_1 + C_2$
$C_2 + C_3$
$C_1 + C_3$
$C_1 + C_2 + C_3$

Con probabilidades

$q_1\ q_2\ q_3$

$p_1\ q_2\ q_3$

$q_1\ p_2\ q_3$

$q_1\ q_2\ p_3$

$p_1\ p_2\ q_3$

$q_1\ p_2\ p_3$

$p_1\ q_2\ p_3$

$p_1\ p_2\ p_3$ respectivamente, entonces, para estar de acuerdo con el cálculo de probabilidades, la suma de las probabilidades de cada uno de los posibles sucesos debe ser 1:

$$q_1\ q_2\ q_3 + p_1\ q_2\ q_3 + q_1\ p_2\ q_3 + q_1\ q_2\ p_3 + p_1\ p_2\ q_3 + q_1\ p_2\ p_3 + p_1\ q_2\ p_3 + p_1\ p_2\ p_3 =$$

$$q_1\ (q_2\ q_3 + p_2\ q_3 + q_2\ p_3 + p_2\ p_3) + p_1\ (q_2\ q_3 + p_2\ q_3 + q_2\ p_3 + p_2\ p_3\ ) =$$

$$q_1\ (q_2 + p_2) + p_1\ (q_2 + p_2\ ) =$$

$$q_1 + p_1 = 1$$

El procedimiento esbozado puede ampliarse de modo que agregando un generador cada vez a un sistema existente cuyas características se conocen, es posible obtener, para cualquier número de generadores, un listado que muestre cada indisponibilidad posible que pudiera presentarse y la correspondiente probabilidad de su existencia.

Con lo hecho hasta ahora hemos obtenido una expresión cuantitativa de la confiabilidad del suministro de potencia del sistema, ahora nos interesa calcular la probabilidad de pérdida de carga.

CURVA DE DURACION DE CARGA

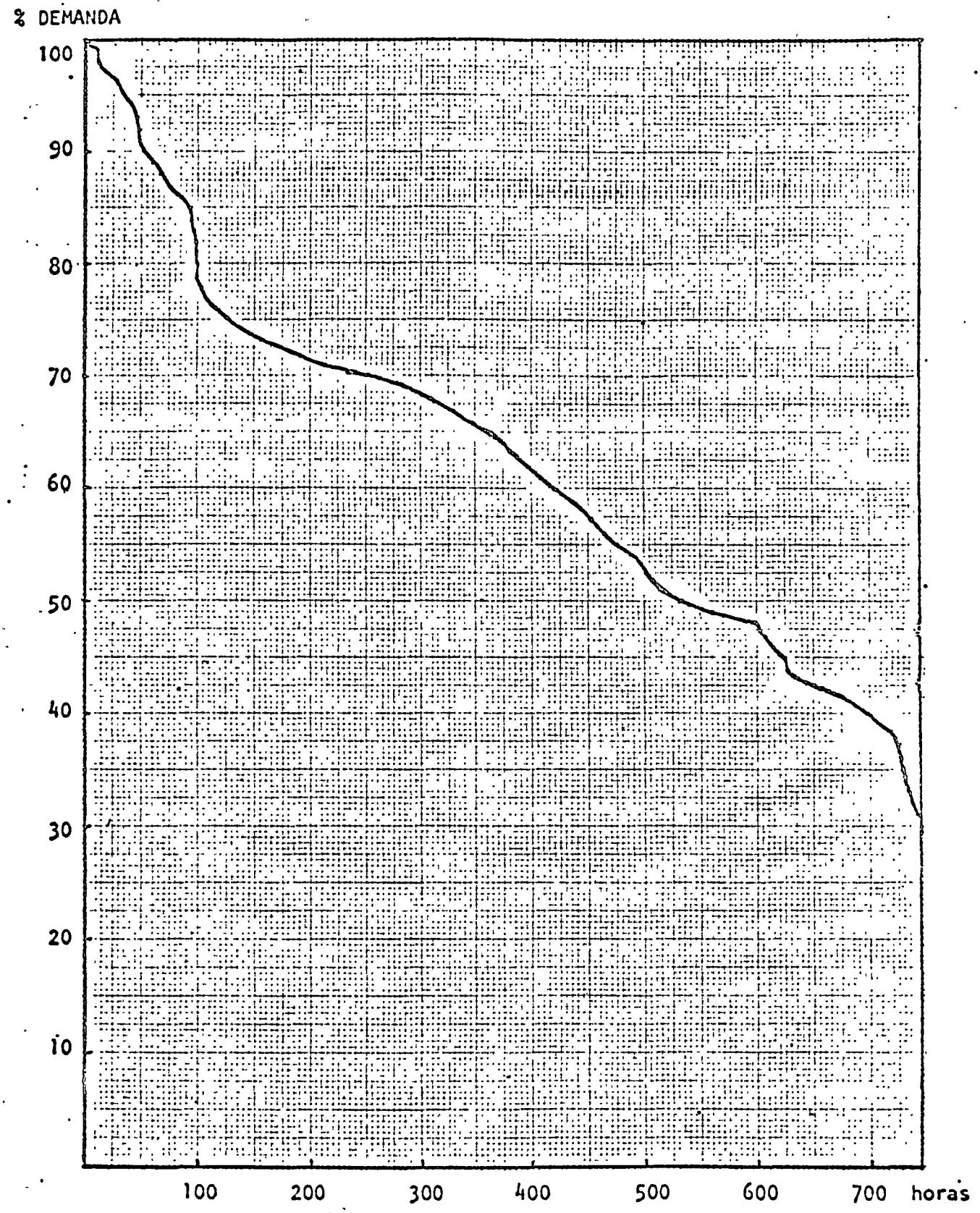
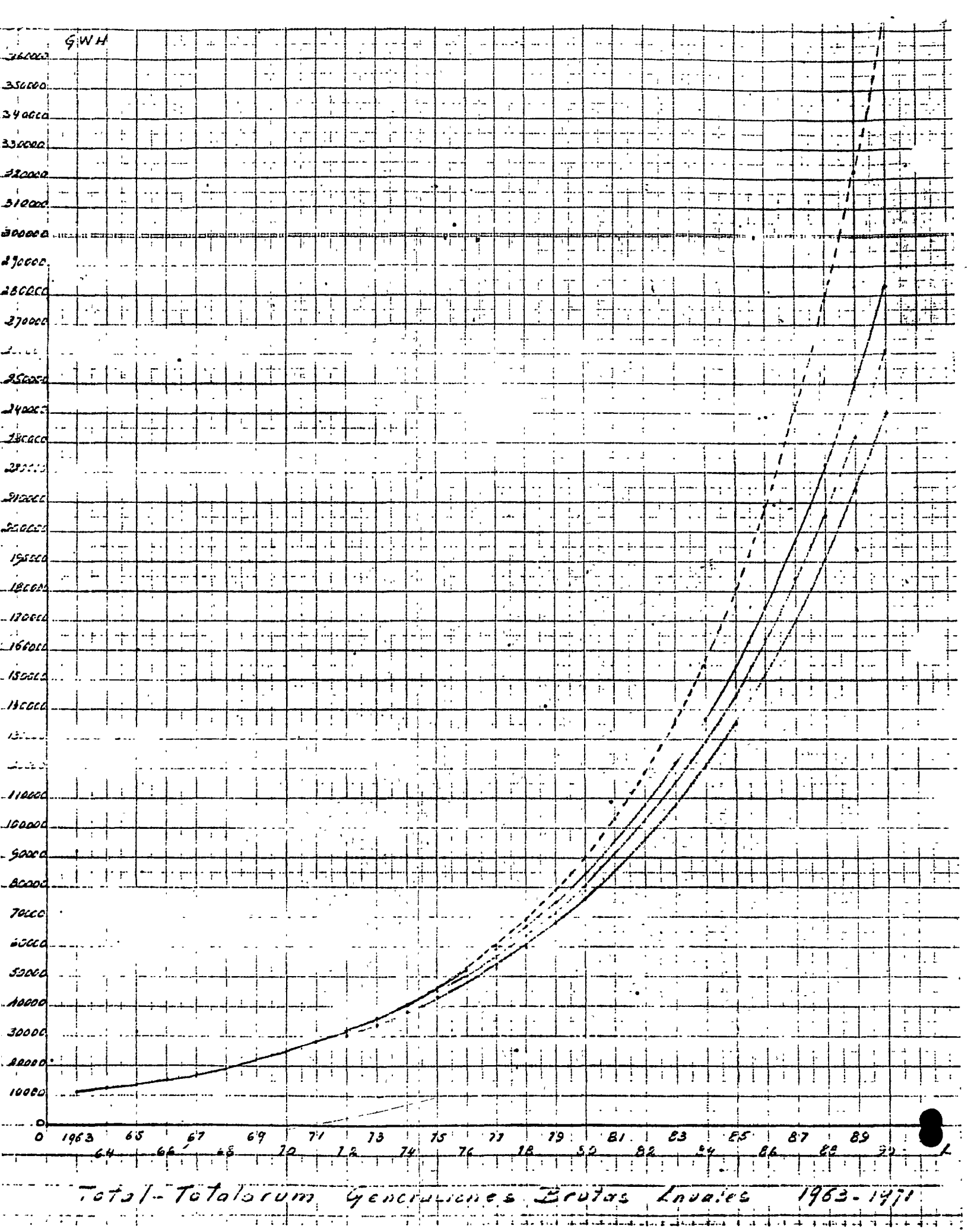


Fig. 2.5



Total - Totalorum Generationes Brutas Anuales 1963-1991

SISTEMA EJEMPLO AÑOS 1954 A 1966

Fig. 2.10 DATOS HISTORICOS

AÑO	CARGA
1954	499.0
1955	543.0
1956	602.0
1957	681.0
1958	720.0
1959	782.0
1960	821.0
1961	894.0
1962	948.0
1963	1037.0
1964	1159.0
1965	1270.0
1966	1356.0

Fig.2.11 PRONOSTICOS PARA EL SISTEMA EJEMPLO

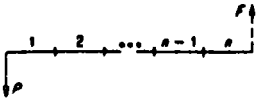
AÑO	DEMANDA MAXIMA MW	DESVIACION ESTANDAR
1967	1468.929	0.2642012E-01
1968	1593.480	0.2721690E-01
1969	1728.589	0.2809259E-01
1970	1875.157	0.2904008E-01
1971	2034.150	0.3005257E-01
1972	2206.625	0.3112371E-01
1973	2393.726	0.3224767E-01
1974	2596.687	0.3341911E-01
1975	2816.861	0.3463323E-01
1976	3055.703	0.3588568E-01
1977	3314.794	0.3717259E-01
1978	3595.855	0.3849050E-01
1979	3900.745	0.3983635E-01
1980	4231.488	0.4120739E.01
1981	4590.277	0.4260119E-01
1982	4979.484	0.4401558E-01
1983	5401.695	0.4544866E-01
1984	5859.707	0.4689869E-01
1985	6356.547	0.4836415E-01
1986	6895.520	0.4984370E-01
1987	7480.184	0.5133610E-01
1988	8114.430	0.5284027E-01
1989	8802.453	0.5435524E-01
1990	9548.805	0.5588011E-01
1991	10358.449	0.5741412E-01
1992	11236.742	0.5895653E-01
1993	12189.496	0.6050672E-01
1994	13223.047	0.6206409E-01
1995	14344.219	0.6362803E-01
1996	15560.465	0.6519830E-01

EVALUACION DE PROYECTOS

Los proyectos de grandes inversiones, generalmente se evalúan en moneda presente comparando los beneficios que se esperan obtener con el costo del proyecto.

Este programa encuentra las tablas de amortización de los créditos que financían el proyecto y hace la evaluación del mismo. En los costos del proyecto no se incluyen los intereses. A continuación proporcionamos las fórmulas de interés compuesto y enseguida ilustramos el programa con un ejemplo.

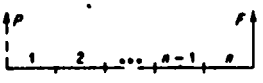
The cash-flow diagrams represent the position of the person owning the cash who deposits in or withdraws from an interest bearing fund. Downward is outgo, or deposit, and upward is income or withdrawal. Solid arrows represent known cash flows and dashed arrows represent unknown cash flows, or solutions of the equations.



Compound amount at the end of n periods to which a single deposit will accumulate

$$F = P(1+i)^n; \quad CA = (1+i)^n$$

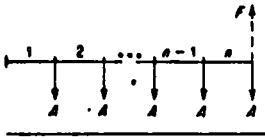
FSP (6-1)



Present worth of a single sum to be withdrawn at the end of n periods in the future:

$$P = F\frac{1}{(1+i)^n}; \quad PW = \frac{1}{(1+i)^n}$$

PSF (6-2)



Compound amount at the end of n periods to which a series of n uniform period-end deposits will accumulate

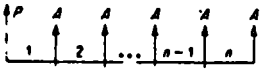
$$F = A\frac{(1+i)^n - 1}{i}; \quad SCA = \frac{(1+i)^n - 1}{i}$$

FSA (6-3)

Sinking fund uniform period-end deposit which will accumulate to a given sum at the end of n periods

$$A = F\frac{i}{(1+i)^n - 1}; \quad SF = \frac{i}{(1+i)^n - 1}$$

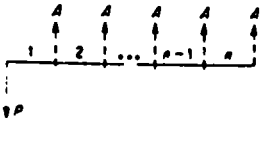
ASF (6-4)



Present worth of a series of n uniform period end withdrawals

$$P = A\frac{(1+i)^n - 1}{i(1+i)^n}; \quad SPW = \frac{(1+i)^n - 1}{i(1+i)^n}$$

PSA (6-5)



Capital recovery with interest, annuity which will return in n period end uniform receipts a given present deposit plus interest on the unreturned portion

$$A = P\frac{i(1+i)^n}{(1+i)^n - 1}; \quad CR = \frac{i(1+i)^n}{(1+i)^n - 1}$$

ASP (6-6)

FIG. 6-1. Cash-flow diagrams and the six standard compound interest equations based on the period-end step convention.

EJEMPLO. EVALUACION DE UN PROYECTO DE RIEGO

Se tiene en proyecto la construcción de una Presa para riego, la cual se piensa financiar con fondos fiscales y fondos del Banco Interamericano de Desarrollo. Las caracte - rísticas de los créditos se muestran en el Cuadro I y los datos del proyecto se presentan en el Cuadro II.

Terminadas las obras se espera tener los siguientes beneficios: los 3 primeros años 32.39 millones anuales, los años siguientes 34.14 millones. La vida útil del proyecto es de 50 años.

C U A D R O I

	Plazo años	Gracia	Interés %	Com. Servicio	Comisión Compromiso
REG. FISC.	20	0	6	0	0
BID (ROC)	20	3	7	1	1
BID (ROE)	20	3	3	1	1

C U A D R O II. PROYECTO

	AVANCE DEL PROYECTO					COSTO EN	F I N A N C I A M I E N T O		
	1 9 7 3		1 9 7 4		1975	MILLONES	GOB.	BID	BID
	I	II	III	IV	V	DE PESOS	FED.	ROC	ROE
Vaso	0.24	0.24	0.24	0.28		50.0	100%		
Canal principal			.376	.376	.248	26.6		100%	
Estructuras del C.P.			.357	.357	.286	7.4		100%	
Canales de distribución			.333	.333	.333	9.0			100%
Estructuras de los Canales			.316	.316	.368	7.9			100%
Drenaje			.33	.33	.33	1.5			100%
Caminos	.576	.424				5.2			100%
Imprevisto					1.00	3.1		100%	

DISTRIBUCION PERIODICA POR FUENTES

PERIODO	1	2	3	4	5
1	12.00	12.00	12.00	14.00	3.10
2	0.00	0.00	12.64	12.64	8.71
3	3.00	2.20	5.99	5.99	6.41

FUENTES Y USOS DE FONDOS

	GOB.FED.	ROC-BID	ROE-BID
PRESA	50.00	0.00	0.00
CANAL PPAL.	0.00	26.60	0.00
ESTR. CANAL P.	0.00	7.40	0.00
CANALES DISTRIB.	0.00	0.00	9.00
EST.CAN. DISTRIB	0.00	0.00	7.90
DRENAJE	0.00	0.00	1.50
CAMINUS	0.00	0.00	5.20
IMPREVISTOS	3.10	0.00	0.00

DISTRIBUCION PERIODICA POR USOS

PERIODO	1	2	3	4	5
PRESA	12.00	12.00	12.00	14.00	0.00
CANAL PPAL.	0.00	0.00	10.00	10.00	6.60
ESTR. CANAL P.	0.00	0.00	2.64	2.64	2.12
CANALES DISTRIB.	0.00	0.00	3.00	3.00	3.01
EST.CAN. DISTRIB	0.00	0.00	2.50	2.50	2.91
DRENAJE	0.00	0.00	0.50	0.50	0.50
CAMINUS	3.00	2.20	0.00	0.00	0.00
IMPREVISTOS	0.00	0.00	0.00	0.00	3.10

CONDICIONES DE LOS CREDITOS

	PLAZO	GRACIA	INTERES	COMISION SERVICIO	COMISION COMPROMISO
GOB.FED.	20.00	0.00	6.00	0.00	0.00
ROC-BID	20.00	3.00	7.00	1.00	1.00
ROE-BID	20.00	3.00	3.00	1.00	1.00

ROE-BID

INTERES	3.00
PLAZO	20.00
PERIODO DE GRACIA	3.00
COM. DE COMPROMISO	1.00
COM. DE SERVICIO	1.00

CREDITO 23,600 MILL

PERIODO	COMISION COMPROMISO	COMISION SERVICIO	INTERESES	AMORTI-IZACION	PAGO
1	0.04	0.01	0.04	0.00	0.10
2	0.04	0.03	0.08	0.00	0.14
3	0.02	0.06	0.17	0.00	0.25
4	0.01	0.09	0.26	0.00	0.36
5	0.00	0.12	0.35	0.00	0.47
6	0.00	0.00	0.00	0.00	0.00
7	0.00	0.12	0.35	0.49	0.96
8	0.00	0.12	0.35	0.50	0.96
9	0.00	0.11	0.34	0.51	0.96
10	0.00	0.11	0.33	0.52	0.96
11	0.00	0.11	0.32	0.53	0.96
12	0.00	0.11	0.32	0.54	0.96
13	0.00	0.10	0.31	0.55	0.96
14	0.00	0.10	0.30	0.56	0.96
15	0.00	0.10	0.29	0.58	0.96
16	0.00	0.09	0.28	0.59	0.96
17	0.00	0.09	0.27	0.60	0.96
18	0.00	0.09	0.26	0.61	0.96
19	0.00	0.09	0.26	0.62	0.96
20	0.00	0.08	0.25	0.64	0.96
21	0.00	0.08	0.24	0.65	0.96
22	0.00	0.08	0.23	0.66	0.96
23	0.00	0.07	0.22	0.67	0.96
24	0.00	0.07	0.21	0.69	0.96
25	0.00	0.07	0.20	0.70	0.96
26	0.00	0.06	0.19	0.72	0.96
27	0.00	0.06	0.17	0.73	0.96
28	0.00	0.05	0.16	0.74	0.96
29	0.00	0.05	0.15	0.76	0.96
30	0.00	0.05	0.14	0.77	0.96
31	0.00	0.04	0.13	0.79	0.96
32	0.00	0.04	0.12	0.81	0.96
33	0.00	0.04	0.11	0.82	0.96
34	0.00	0.03	0.09	0.84	0.96
35	0.00	0.03	0.08	0.86	0.96
36	0.00	0.02	0.07	0.87	0.96
37	0.00	0.02	0.06	0.89	0.96
38	0.00	0.01	0.04	0.91	0.96
39	0.00	0.01	0.03	0.93	0.96
40	0.00	0.00	0.01	0.94	0.96

ROC-BID

INTERES	7.00
PLAZO	20.00
PERIODO DE GRACIA	3.00
COM.DE COMPROMISO	1.00
COM.DE SERVICIO	1.00

CREDITO 34,000 MILL

PERIODO	COMISION COMPROMISO	COMISION SERVICIO	INTERESES	AMORTI-IZACION	PAGO
1	0.07	0.00	0.00	0.00	0.07
2	0.07	0.00	0.00	0.00	0.07
3	0.04	0.06	0.44	0.00	0.55
4	0.02	0.13	0.89	0.00	1.03
5	0.00	0.17	1.19	0.00	1.36
6	0.00	0.00	0.00	0.00	0.00
7	0.00	0.17	1.19	0.49	1.85
8	0.00	0.17	1.17	0.51	1.85
9	0.00	0.17	1.16	0.53	1.85
10	0.00	0.16	1.14	0.55	1.85
11	0.00	0.16	1.12	0.57	1.85
12	0.00	0.16	1.10	0.59	1.85
13	0.00	0.15	1.08	0.62	1.85
14	0.00	0.15	1.06	0.64	1.85
15	0.00	0.15	1.03	0.67	1.85
16	0.00	0.14	1.01	0.69	1.85
17	0.00	0.14	0.99	0.72	1.85
18	0.00	0.14	0.96	0.75	1.85
19	0.00	0.13	0.93	0.78	1.85
20	0.00	0.13	0.91	0.81	1.85
21	0.00	0.13	0.88	0.84	1.85
22	0.00	0.12	0.85	0.88	1.85
23	0.00	0.12	0.82	0.91	1.85
24	0.00	0.11	0.79	0.95	1.85
25	0.00	0.11	0.75	0.99	1.85
26	0.00	0.10	0.72	1.03	1.85
27	0.00	0.10	0.68	1.07	1.85
28	0.00	0.09	0.65	1.11	1.85
29	0.00	0.09	0.61	1.15	1.85
30	0.00	0.08	0.57	1.20	1.85
31	0.00	0.07	0.52	1.25	1.85
32	0.00	0.07	0.48	1.30	1.85
33	0.00	0.06	0.44	1.35	1.85
34	0.00	0.06	0.39	1.40	1.85
35	0.00	0.05	0.34	1.46	1.85
36	0.00	0.04	0.29	1.52	1.85
37	0.00	0.03	0.23	1.58	1.85
38	0.00	0.03	0.18	1.64	1.85
39	0.00	0.02	0.12	1.71	1.85
40	0.00	0.01	0.06	1.78	1.85

GUB.FED.

INTERES	6.00
PLAZO	20.00
PERIODO DE GRACIA	0.00
COM.DE COMPROMISO	0.00
COM.DE SERVICIO	0.00

CREDITO 53.100 MILL

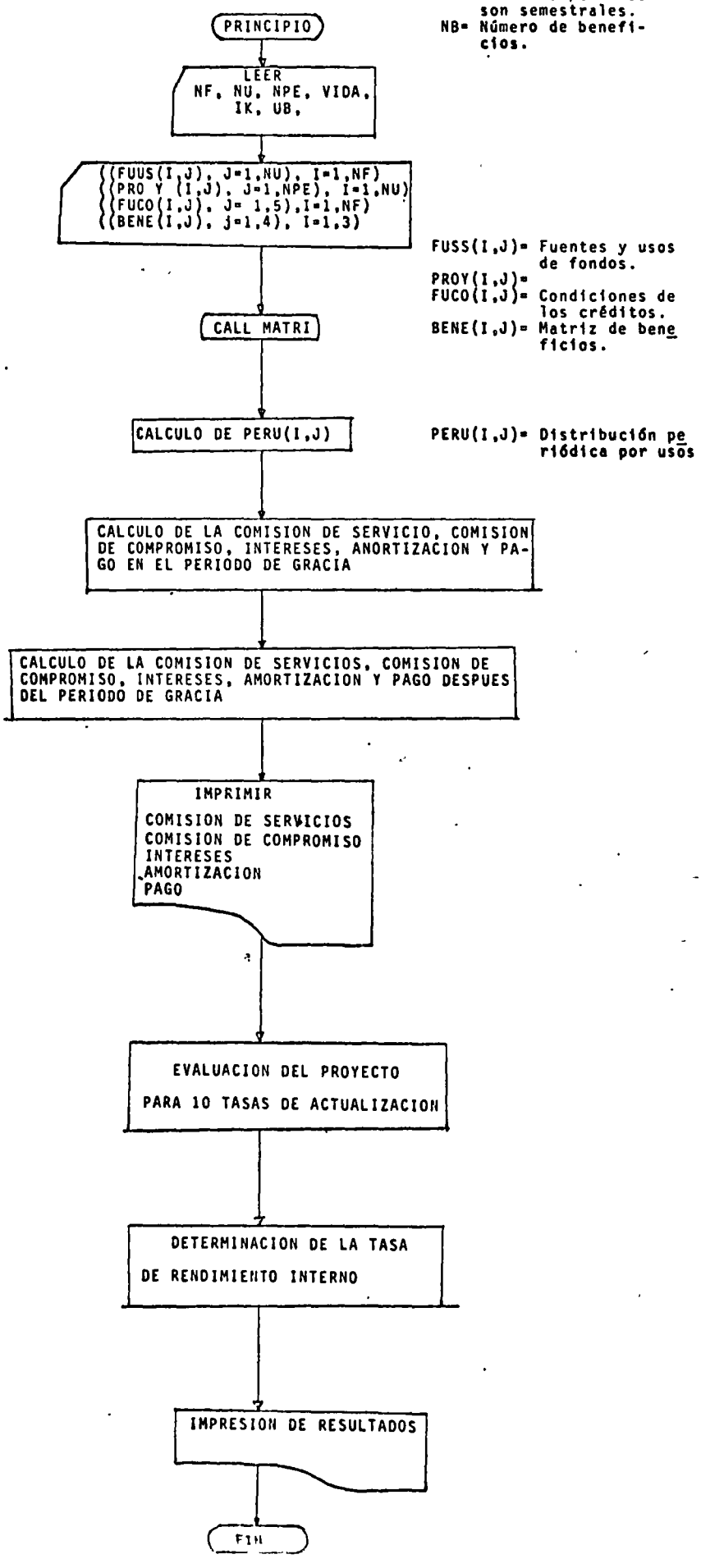
PERIODO	COMISION COMPROMISO	COMISION SERVICIO	INTERESES	AMORTI-IZACION	PAGO
1	0.00	0.00	1.59	0.70	2.30
2	0.00	0.00	1.57	0.73	2.30
3	0.00	0.00	1.55	0.75	2.30
4	0.00	0.00	1.53	0.77	2.30
5	0.00	0.00	1.50	0.79	2.30
6	0.00	0.00	1.48	0.82	2.30
7	0.00	0.00	1.46	0.84	2.30
8	0.00	0.00	1.43	0.87	2.30
9	0.00	0.00	1.41	0.89	2.30
10	0.00	0.00	1.38	0.92	2.30
11	0.00	0.00	1.35	0.95	2.30
12	0.00	0.00	1.32	0.97	2.30
13	0.00	0.00	1.29	1.00	2.30
14	0.00	0.00	1.26	1.03	2.30
15	0.00	0.00	1.23	1.07	2.30
16	0.00	0.00	1.20	1.10	2.30
17	0.00	0.00	1.17	1.13	2.30
18	0.00	0.00	1.13	1.16	2.30
19	0.00	0.00	1.10	1.20	2.30
20	0.00	0.00	1.06	1.23	2.30
21	0.00	0.00	1.03	1.27	2.30
22	0.00	0.00	0.99	1.31	2.30
23	0.00	0.00	0.95	1.35	2.30
24	0.00	0.00	0.91	1.39	2.30
25	0.00	0.00	0.87	1.43	2.30
26	0.00	0.00	0.82	1.47	2.30
27	0.00	0.00	0.78	1.52	2.30
28	0.00	0.00	0.73	1.56	2.30
29	0.00	0.00	0.69	1.61	2.30
30	0.00	0.00	0.64	1.66	2.30
31	0.00	0.00	0.59	1.71	2.30
32	0.00	0.00	0.54	1.76	2.30
33	0.00	0.00	0.48	1.81	2.30
34	0.00	0.00	0.43	1.87	2.30
35	0.00	0.00	0.37	1.92	2.30
36	0.00	0.00	0.32	1.98	2.30
37	0.00	0.00	0.26	2.04	2.30
38	0.00	0.00	0.19	2.10	2.30
39	0.00	0.00	0.13	2.17	2.30
40	0.00	0.00	0.07	2.23	2.30

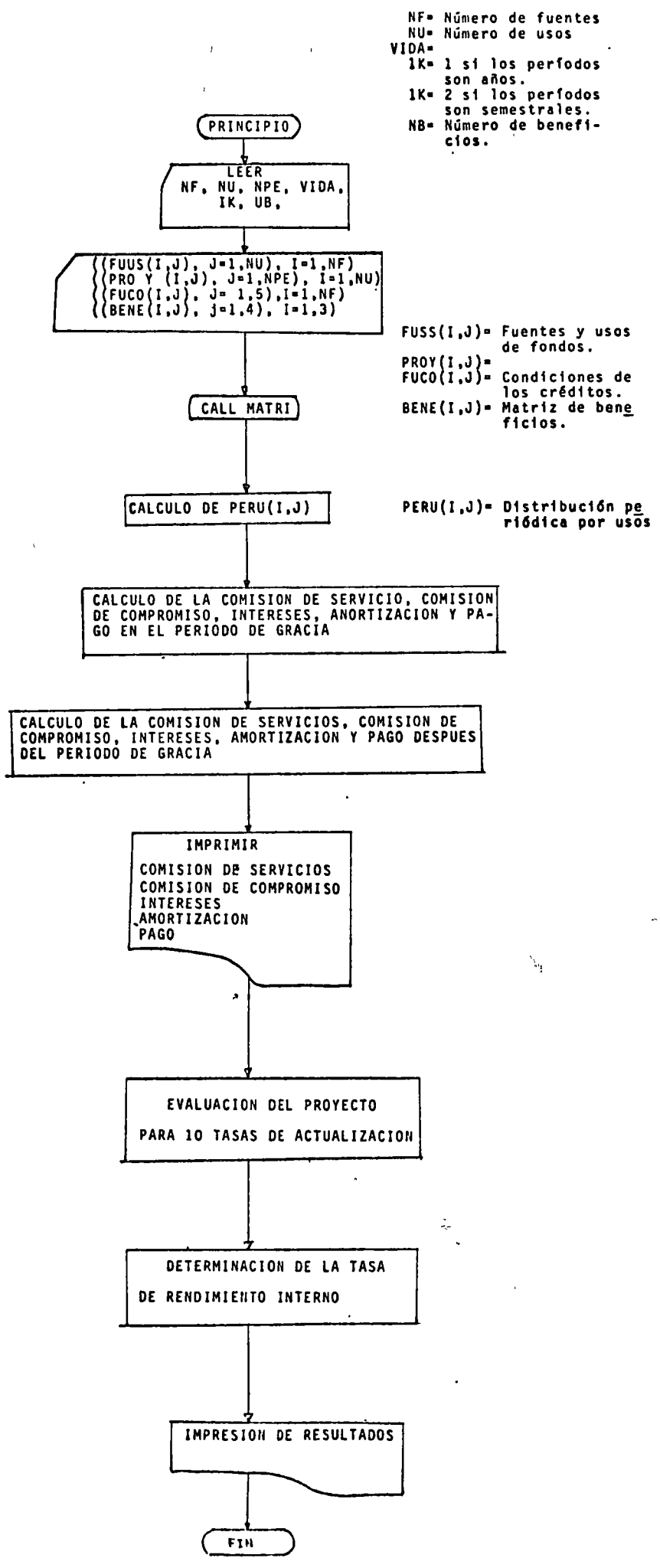
EDUCACION CONTINUA

E V A L U A C I O N

TASA DE RENDIMIENTO INTERNO 0.278

TASA	COSTO	BENEF	B-C	B/C	APROV.
0.050	109.06	581.23	472.17	5.33	19.
0.060	106.40	494.25	387.86	4.65	22.
0.070	104.01	425.99	321.98	4.10	24.
0.080	101.84	371.51	269.67	3.65	27.
0.090	99.84	327.36	227.52	3.28	30.
0.100	97.96	291.05	193.09	2.97	34.
0.110	96.20	260.81	164.61	2.71	37.
0.120	94.52	235.30	140.77	2.49	40.
0.130	92.93	213.55	120.63	2.30	44.
0.140	91.39	194.83	103.44	2.13	47.


NF= Número de fuentes
NU= Número de usos
VIDA=
1K= 1 si los períodos son años.
1K= 2 si los períodos son semestrales.
NB= Número de beneficios.
PRINCIPIO
LEER
NF, NU, NPE, VIDA, IK, UB,
((FUUS(I,J), J=1,NU), I=1,NF)
((PRO Y (I,J), J=1,NPE), I=1,NU)
((FUCO(I,J), J= 1,5),I=1,NF)
((BENE(I,J), J=1,4), I=1,3)
FUSS(I,J)= Fuentes y usos de fondos.
PROY(I,J)=
FUCO(I,J)= Condiciones de los créditos.
BENE(I,J)= Matriz de beneficios.
CALL MATRI
CALCULO DE PERU(I,J)
PERU(I,J)= Distribución periódica por usos
CALCULO DE LA COMISION DE SERVICIO, COMISION DE COMPROMISO, INTERESES, ANORTIZACION Y PAGO EN EL PERIODO DE GRACIA
CALCULO DE LA COMISION DE SERVICIOS, COMISION DE COMPROMISO, INTERESES, AMORTIZACION Y PAGO DESPUES DEL PERIODO DE GRACIA
IMPRIMIR
COMISION DE SERVICIOS
COMISION DE COMPROMISO
INTERESES
AMORTIZACION
PAGO
EVALUACION DEL PROYECTO
PARA 10 TASAS DE ACTUALIZACION
DETERMINACION DE LA TASA
DE RENDIMIENTO INTERNO
IMPRESION DE RESULTADOS
FIN


NF= Número de fuentes
NU= Número de usos
VIDA=
IK= 1 si los períodos son años.
IK= 2 si los períodos son semestrales.
NB= Número de beneficios.
PRINCIPIO
LEER
NF, NU, NPE, VIDA,
IK, UB,
((FUUS(I,J), J=1,NU), I=1,NF)
((PRO Y (I,J), J=1,NPE), I=1,NU)
((FUCO(I,J), J= 1,5),I=1,NF)
((BENE(I,J), j=1,4), I=1,3)
FUSS(I,J)= Fuentes y usos de fondos.
PROY(I,J)=
FUCO(I,J)= Condiciones de los créditos.
BENE(I,J)= Matriz de beneficios.
CALL MATRI
CALCULO DE PERU(I,J)
PERU(I,J)= Distribución periódica por usos
CALCULO DE LA COMISION DE SERVICIO, COMISION DE COMPROMISO, INTERESES, ANORTIZACION Y PAGO EN EL PERIODO DE GRACIA
CALCULO DE LA COMISION DE SERVICIOS, COMISION DE COMPROMISO, INTERESES, AMORTIZACION Y PAGO DESPUES DEL PERIODO DE GRACIA
IMPRIMIR
COMISION DE SERVICIOS
COMISION DE COMPROMISO
INTERESES
AMORTIZACION
PAGO
EVALUACION DEL PROYECTO
PARA 10 TASAS DE ACTUALIZACION
DETERMINACION DE LA TASA
DE RENDIMIENTO INTERNO
IMPRESION DE RESULTADOS
FIN

```
B 6 7 0 0   F O R T R A N   C O M P I L A T I O N   M A R K   II.01.99     FRIDAY,  05/19/72,  02:49 PM
                                                                                FIB IS 0006 LONG
                                                                                FIB IS 0006 LONG
                                                                           START OF SEGMENT 002
$ SET SINGLE LIST                                                       C  002:000:0
$ SET BCD                                                               C  002:000:0
FILE  5=CARD,UNIT=READER                                                C  002:000:0
FILE  6=PRT,UNIT=PRINT                                                  C  002:000:0
      DIMENSION BENE(10,4)                                              C  002:000:0
      DIMENSION RES(10,3)                                               C  002:000:0
      DIMENSION TITULO(10),FUUS(10,15),PROY(15,20),FUCO(10,5),FUTE(10,4)C  002:000:0
     *,USO(10,4),PROINV(20,20),FUUST(10,10),PERU(10,10),COMCOM(10,10),FI C  002:000:0
     *NTE(10,10),COMSER(10,10),UNI(5),Q(50)                             C  002:000:0
      REAL INV(20)                                                      C  002:000:0
      DIMENSION A(50,5)                                                 C  002:000:0
   80 FORMAT(10A4)                                                      C  002:000:0
   81 FORMAT(5I2,F10.0)                                                 C  002:000:0
   82 FORMAT(8F10.0)                                                    C  002:000:0
   83 FORMAT(I2,2X,4A4,5F5.0)                                           C  002:000:0
   85 FORMAT(15H          CREDITO,F10.3,2A4)                            C  002:000:0
   86 FORMAT(10X,7HPERIODO,/,16X,10(I2,8X),/,2X,7HFUENTES,/)            C  002:000:0
   87 FORMAT(4X,I2,4X,10F10.2)                                          C  002:000:0
   88 FORMAT(21X,10(2A4,2X))                                            C  002:000:0
   89 FORMAT(1X,4A4,10F10.2)                                            C  002:000:0
   90 FORMAT(17X,7HPERIODO,/,14X,10(I2,8X),/,2X,4HUSOS,/)               C  002:000:0
   91 FORMAT(22X,49HPLAZO     GRACIA   INTERES   SERVICIO  COMPROMISO,/) C  002:000:0
   92 FORMAT(51X,18HCOMISION  COMISION)                                 C  002:000:0
   93 FORMAT(13X,I2,1X,5F10.2)                                          C  002:000:0
   95 FORMAT(17HINTERES          ,F10.2/17HPLAZO              ,F10.2/   C  002:000:0
     *17HPERIODO DE GRACIA  ,F10.2/17HCOM.DE COMPROMISO ,F10.2/         C  002:000:0
     *17HCOM.DE SERVICIO    ,F10.2/)                                    C  002:000:0
   97 FORMAT(21X,18HCOMISION  COMISION,13X,7HAMORTI-,/,10X,56HPERIODO  C  C  002:000:0
     *OMPROMISO SERVICIO  INTERESES  IZACION    PAGO  ,/)               C  002:000:0
   99 FORMAT(///)                                                       C  002:000:0
  100 FORMAT(1H1)                                                       C  002:000:0
  111 FORMAT(9X,60H************************************************************C  002:000:0
     ********)                                                          C  002:000:0
  112 FORMAT(22X,34HDISTRIBUCION PERIODICA POR FUENTES)                 C  002:000:0
  113 FORMAT(19X,30H******************************)                     C  002:000:0
  114 FORMAT(22X,24HFUENTES Y USOS DE FONDOS)                           C  002:000:0
  115 FORMAT(20X,58H**********************************************************C  002:000:0
     *******)                                                           C  002:000:0
  116 FORMAT(33X,31HDISTRIBUCION PERIODICA POR USOS)                    C  002:000:0
  117 FORMAT(17X,7HPERIODO,/,23X,10(I2,8X),/,2X,4HUSOS,/)               C  002:000:0
  118 FORMAT(19X,50H**************************************************)     C  002:000:0
  119 FORMAT(31X,27HCONDICIONES DE LOS CREDITOS)                        C  002:000:0
  908 FORMAT (8F10.0)                                                   C  002:000:0
C FUUS(I,J)= CANTIDAD DE DINERO DE LA FUENTE I EN EL USO J              C  002:000:0
C PROY(I,J) = PORCIENTO DE AVANCE DE LA ACTIVIDAD I EN EL PERIODO J     C  002:000:0
C PARA LA FUENTE DE FINANCIAMIENTO I                                    C  002:000:0
C FUCO(I,1)=PLAZO                                                       C  002:000:0
C FUCO(I,2)=PERIODO DE GRACIA                                           C  002:000:0
C FUCO(I,3)=INTERES                                                     C  002:000:0
C FUCO(I,4)=COMISION DE COMPROMISO                                      C  002:000:0
C FUCO(I,5)=COMISION DE SERVICIO                                        C  002:000:0
C FUTE(I,K),K=1,4 NOMBRE DE LA FUENTE I(4 PALABRAS)                     C  002:000:0
C USO(I,K),K=1,4 NOMBRE DE LA ACTIVIDAD I (4 PALABRAS)                  C  002:000:0
C     EN SEGUIDA SE PROCEDE A LA LECTURA DE DATOS                       C  002:000:0
      READ(5,80)(TITULO(I),I=1,10)                                      C  002:000:0
      READ(5,81)NF,NU,NPE,ND,IK,VIDA                                    C  002:001:2
```

```
      READ(5,82) NOPE ,OPMA                                              C 002:0029:0
      READ(5,80)UNI(1),UNI(2)                                            C 002:0038:3
      DO 1 I=1,NF                                                        C 002:0048:3
    1 READ(5,82)(FUUS(I,J),J=1,NU)                                       C 002:0049:1
      DO 2 I=1,NU                                                        C 002:005E:0
    2 READ(5,82)(PROY(I,J),J=1,NPE)                                      C 002:005E:4
      DO 3 I=1,NF                                                        C 002:0073:3
    3 READ(5,82)(FUCO(I,J),J=1,5)                                        C 002:0074:1
      NP=NPE                                                             C 002:0089:0
      DO 4 I=1,NF                                                        C 002:0089:5
    4 READ(5,80)(FUTE(I,K),K=1,4)                                        C 002:008A:3
      DO 5 I=1,NU                                                        C 002:009F:0
    5 READ(5,80)(USO(I,K),K=1,4)                                         C 002:009F:4
C     SE SUMAN LOS VALORES EN CADA UNO DE LOS RENGLONES DE LA MATRIZ     C 002:00B4:3
C     FUENTES Y USOS DE FONDOS,COLOCANDO EL RESULTADO EN UNA NUEVA       C 002:00B4:3
C     COLUMNA                                                            C 002:00B4:3
      DO 25 J=1,NF                                                       C 002:00B4:3
      FUUS(J,NU+1)=0                                                     C 002:00B5:1
      DO 25 I=1,NU                                                       C 002:00B8:1
   25 FUUS(J,NU+1)=FUUS(J,NU+1)+FUUS(J,I)                                C 002:00B8:5
      DO 26 J=1,NU                                                       C 002:00C5:1
      FUUS(NF+1,J)=0                                                     C 002:00C5:5
      DO 26 I=1,NF                                                       C 002:00C8:5
   26 FUUS(NF+1,J)=FUUS(NF+1,J)+FUUS(I,J)                                C 002:00C9:3
C     SE EFECTUA EL PRODUCTO DE LAS MATRICES FUENTES Y USOS DE FONDOS    C 002:00D5:5
C     Y LA MATRIZ DE PORCENTAJES,OBTENIENDOSE LA MATRIZ DISTRIBUCION     C 002:00D5:5
C     PERIODICA POR FUENTES                                              C 002:00D5:5
      CALL MATRI(FUUS,PROY,PROINV,NF,NP,NU)                              C 002:00D5:5
      WRITE(6,99)                                                        C 002:00DA:0
      WRITE(6,111)                                                       C 002:00DF:1
      WRITE(6,112)                                                       C 002:00E4:3
      WRITE(6,111)                                                       C 002:00E9:5
      WRITE(6,86) (I,I=1,NPE)                                            C 002:00EF:1
      DO 6 I=1,NF                                                        C 002:00FF:5
    6 WRITE(6,87)I,(PROINV(I,J),J=1,NPE)                                 C 002:0100:3
      WRITE(6,99)                                                        C 002:0117:3
      DO 7 I=1,NF                                                        C 002:011C:4
      PI=0                                                               C 002:011D:2
      DO 7 J=1,NPE                                                       C 002:011E:0
      PI=PI+PROINV(I,J)                                                  C 002:011E:4
      COMCOM(I,J)=(FUUS(I,NU+1)-PI)*FUCO(I,5)/(K*100)                    C 002:0122:0
      FINTE(I,J)=PI*FUCO(I,3)/200                                        C 002:012A:1
      COMSER(I,J)=PI*FUCO(I,4)/200                                       C 002:012F:0
    7 CONTINUE                                                           C 002:0133:5
      WRITE(6,113)                                                       C 002:0138:1
      WRITE(6,114)                                                       C 002:013D:3
      WRITE(6,113)                                                       C 002:0142:5
      WRITE(6,88)(FUTE(I,1),FUTE(I,2),I=1,NF)                            C 002:0148:1
      DO 9 I=1,NF                                                        C 002:015C:5
      DO 9 J=1,NU                                                        C 002:015D:3
    9 FUUST(J,I)=FUUS(I,J)                                               C 002:015E:1
      DO 8 I=1,NU                                                        C 002:0167:2
    8 WRITE(6,89)(USO(I,K),K=1,4),(FUUST(I,J),J=1,NF)                    C 002:0168:0
      WRITE(6,99)                                                        C 002:0184:3
C     OBTENCION DE LA MATRIZ DISTRIBUCION PERIODICA POR USOS             C 002:0189:4
      DO 10 J=1,NU                                                       C 002:0189:4
      DO 10 M=1,NP                                                       C 002:018A:2
   10 PERU(J,M)=FUUS(NF+1,J)*PROY(J,M)                                   C 002:018B:0
      WRITE(6,115)                                                       C 002:0197:0
      WRITE(6,116)                                                       C 002:019C:2
```

```
      WRITE(6,115)                                                    C  002:01A1:4
      WRITE(6,117) (I,I=1,NP)                                         C  002:01A7:0
      DO 11 I=1,NU                                                    C  002:01B7:5
   11 WRITE(6,89)(USO(I,K),K=1,4),(PERU(I,K),K=1,NP)                  C  002:01B8:3
      WRITE(6,99)                                                     C  002:01D5:0
      WRITE(6,118)                                                    C  002:01DA:1
      WRITE(6,119)                                                    C  002:01DF:3
      WRITE(6,118)                                                    C  002:01E4:5
      WRITE(6,92)                                                     C  002:01EA:1
      WRITE(6,91)                                                     C  002:01EF:2
      DO 12 I=1,NF                                                    C  002:01F4:3
   12 WRITE(6,89)(FUTE(I,K),K=1,4),(FUCO(I,K),K=1,5)                  C  002:01F5:1
      DO 60 J=1,NF                                                    C  002:0211:3
      WRITE(6,100)                                                    C  002:0212:1
      WRITE(6,99)                                                     C  002:0217:3
      NGR=FUCO(J,2)                                                   C  002:021C:4
      WRITE(6,80)(FUTE(J,K),K=1,4)                                    C  002:021E:4
      WRITE(6,95) FUCO(J,3),FUCO(J,1),FUCO(J,2),FUCO(J,5),FUCO(J,4)   C  002:0231:2
      WRITE(6,85) FUUS(J,NU+1),UNI(1),UNI(2)                          C  002:024C:0
      IF (NGR)79,78,79                                                C  002:0260:3
   79 CONTINUE                                                        C  002:0261:4
C     CALCULO DE LA COMISION DE COMPROMISO,COMISION DE SERVICIO,INTERESE S,  C  002:0261:4
C     AMORTIZACION Y PAGO EN EL PERIODO DE GRACIA                     C  002:0261:4
      DO 61 I=1,NGR*IK                                                C  002:0261:4
      A(I,1)=COMCOM(J,I)                                              C  002:0262:2
      A(I,2)=COMSER(J,I)      A(I,3)=FINTE(J,I)                       C  002:0266:0
      A(I,4)=0.               A(I,5)=A(I,1)+A(I,2)+A(I,3)             C  002:026D:4
   61 CONTINUE                                                        C  002:0275:0
   78 CONTINUE                                                        C  002:0277:4
C     CALCULO DE LA COMISION DE COMPROMISO,COMISION DE SERVICIO,INTERESES  C  002:0277:4
CAMORTIZACION Y PAGO DESPUES DEL PERIODO DE GRACIA                    C  002:0277:4
      NZ=(FUCO(J,1)-FUCO(J,2))*IK                                     C  002:0277:4
      AI=(FUCO(J,3)+FUCO(J,4))/(IK*100)                               C  002:027B:3
      FF=ASP(NZ,AI)                                                   C  002:027F:5
      DO 62 I=1,NZ                                                    C  002:0282:0
      I1=I+NGR*IK                                                     C  002:0282:4
      Q(I)=FUUS(J,NU+1)*((1+AI)**(I))*((1+AI)**(NZ-I)-1)/((1+AI)**NZ-1)  C  002:0284:4
      IF(I.NE.1) GO TO 222                                            C  002:0290:2
      CK=FUUS(J,NU+1)-Q(I)                                            C  002:0291:3
      GO TO 223                                                       C  002:0296:0
  222 CK=Q(I-1)-Q(I)                                                  C  002:0296:3
  223 A(I1,5)=FF*FUUS(J,NU+1)                                         C  002:0299:5
      CIN=A(I1,5)-CK                                                  C  002:029E:3
      A(I1,1)=0                                                       C  002:02A0:5
      A(I1,2)=CIN*FUCO(J,4)/(FUCO(J,3)+FUCO(J,4))                     C  002:02A2:2
      A(I1,3)=CIN-A(I1,2)                                             C  002:02A8:4
      A(I1,4)=CK                                                      C  002:02AC:0
   62 CONTINUE                                                        C  002:02AD:5
      WRITE(6,97)                                                     C  002:02B0:0
C     SE IMPRIMEN LOS VALORES CALCULADOS                              C  002:02B5:2
      DO 63 I=1,FUCO(J,1)*IK                                          C  002:02B5:2
   63 WRITE(6,93) I,(A(I,K),K=1,5)                                    C  002:02B6:0
   60 CONTINUE                                                        C  002:02CE:2
      DO 501 I=1,NB                                                   C  002:02D0:3
  501 READ(5,908)(BENE(I,J),J=1,4)                                    C  002:02D1:1
 1002 FORMAT(8F12.7)                                                  C  002:02E6:0
      DO 510 J=1,NP                                                   C  002:02E6:0
      INV(J)=0.0                                                      C  002:02E6:4
      DO 510 I=1,NF                                                   C  002:02E8:1
      INV (J) = INV(J) + PROINV(I,J)                                  C  002:02E8:5
```

```
  510 CONTINUE                                                          C  002:02ED:5
      WRITE(6,100)                                                      C  002:02F2:1
C*************************************************************** ********  C  002:02F7:3
C                                                                       C  002:02F7:3
C     EN LA PRIMERA PARTE ENCONTRAMOS BENEFICIOS Y COSTOS               C  002:02F7:3
C     PARA SEIS TASAS                                                   C  002:02F7:3
      RET=0,0                                                           C  002:02F7:3
      TAS=0,05                                                          C  002:02F8:1
      DO 610 L=1,10                                                     C  002:02FA:3
      C1=PSA(VIDA,TAS)*OPMA*PSF(NOPE,TAS)                               C  002:02FB:1
      CALL COSTO (INV,IK,NP,TAS,C)                                      C  002:02FF:4
      CALL BENEF(BENE,NB,TAS,RET,B)                                     C  002:0303:0
      RES(L,1)=TAS                                                      C  002:0306:2
      RES(L,2)=C+C1                                                     C  002:0308:0
      RES(L,3)=B                                                        C  002:030A:2
      TAS=TAS+0,01                                                      C  002:030C:1
  610 CONTINUE                                                          C  002:030E:4
C*************************************************************** ********  C  002:0310:5
C                                                                       C  002:0310:5
C     AHORA SE DETERMINA LA TASA DE RENDIMIENTO INTERNO                 C  002:0310:5
C     CON APROXIMACION DE UNO PORCIENTO                                 C  002:0310:5
      DO 640 LL=1,9                                                     C  002:0310:5
      L=11-LL                                                           C  002:0311:3
      Y2=RES(L,2)/RES(L,3)                                              C  002:0313:0
      Y1=RES(L-1,2)/RES(L-1,3)                                          C  002:0316:2
      IF(Y2-1,0)652,652,620                                             C  002:031A:2
  620 IF(Y1-1,0)651,651,640                                             C  002:031D:0
  640 CONTINUE                                                          C  002:0320:0
  652 Y=Y2                                                              C  002:0322:1
      T=RES(L,1)                                                        C  002:0323:0
      GO TO 660                                                         C  002:0324:4
  651 Y=Y1                                                              C  002:0325:1
      T=RES(L-1,1)                                                      C  002:0326:0
  660 T2=RES(L,1)                                                       C  002:0328:0
      T1=RES(L-1,1)                                                     C  002:0329:4
      DO 670 L=1,100                                                    C  002:032B:4
      IF(Y.LE.1.01 .AND. Y .GE.0.99) GO TO 680                          C  002:032C:2
      T=(T2-T1)*(1,0-Y1)/(Y2-Y1)+T1                                     C  002:0330:5
      CALL COSTO (INV,IK,NP,T,C)                                        C  002:0335:4
      C=C+OPMA*PSA(VIDA,T)*PSF(NOPE,T)                                  C  002:0339:0
      CALL BENEF (BENE,NB,T,RET,B)                                      C  002:033E:0
      Y = C/B                                                           C  002:0341:2
           WRITE(6,1002) Y,T1,T2,T                                      C  002:0342:4
      D1=ABS(1,0-Y1)                                                    C  002:0356:0
      D2=ABS(1,0-Y2)                                                    C  002:0359:2
      IF(D1.GT.D2)GO TO 665                                             C  002:035C:2
      Y2=Y                                                              C  002:035D:4
      T2=T                                                              C  002:035E:3
      GO TO 670                                                         C  002:035F:2
  665 T1=T                                                              C  002:035F:5
      Y1=Y                                                              C  002:0360:4
  670 CONTINUE                                                          C  002:0361:3
      WRITE(6,100)                                                      C  002:0363:4
      WRITE(6,900)                                                      C  002:0369:0
  900 FORMAT (13H1 NO CONVERGE )                                        C  002:036E:2
      GO TO 681                                                         C  002:036E:2
C*************************************************************** ********  C  002:036E:5
C                                                                       C  002:036E:5
C     LA SUBRUTINA PINTA IMPRIME LOS RESULTADOS DE LA EVALUACION        C  002:036E:5
  680 WRITE(6,100)                                                      C  002:036E:5
```

```
681 WRITE(6,99)                                          C  002:0374:1
    WRITE(6,80) TITULO                                   C  002:0379:2
    WRITE(6,99)                                          C  002:0388:3
    CALL PINTA (RES,T)                                   C  002:038D:4
    WRITE(6,100)                                         C  002:038F:3
    CALL EXIT                                            C  002:0394:5
    END                                                  C  002:0395:4
                                  REENTRANT FORMAT SEGMENT 003 IS 0018 LONG
                                    NONREENTRANT FORMAT ARRAY IS 009A LONG
                                                 SEGMENT 002 IS 03EF LONG
```

```
      FUNCTION PSA(AN,TA)
C
C     TA=INTERES,AN=TIEMPO EN NUMERO DE PERIODOS
C      ACTUALIZA AL PRINCIPIO DEL PRIMER PERIODO
      IF(TA.LE.0.0001) GO TO 15
      PSA=(1.0-1.0/(1.0+TA)**AN)/TA
      RETURN
   15 PSA=AN
      RETURN
      END
```

```
      FUNCTION FSP(AN,TA)
      IF(AN.LE.0.00001)GO TO 5
      FSP=(1.0+TA)**AN
      RETURN
    5 FSP=1.0
      RETURN
      END
```

```
      FUNCTION PSF(AN,TA)
      IF(AN.LE.0.00001) GO TO 5
      PSF=1.0/(1.0+TA)**AN
      RETURN
    5 PSF=1.0
      RETURN
      END
```

```
      SUBROUTINE MATRI(A,B,C,NF,NP,NU)
      DIMENSION A(10,15),B(15,20) ,C(20,20)
C     10 FUENTES,15 USOS, 20 PERIODOS
      DO 20 I=1,NF
      DO 20 J=1,NP
      C(I,J)=0.0
      DO 2 K=1,NU
    2 C(I,J)= C(I,J)+ A(I,K)*B(K,J)
   20 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE COSTO(INV,K,NP,TASA,COACT)
      REAL INV(20)
C      LAS INVERSIONES SE SUPONEN AL PRINCIPIO DEL PERIODO
C     INV(J)=VALOR DE LA INVERSION EN EL PERIODO J
C      COACT=COSTO ACTUALIZADO
C     TASA=INTERES DE ACTUALIZACION
C     NP=NO. DE PERIODOS
C     K=ESCALA DE TIEMPO, PARA K=1 LOS PERIODOS SON ANOS PARA
C     K=2 LOS PERIODOS SON SEMESTRES, ETC.
      WRITE (6,901)
      COACT=0.0
      DO 20 J=1,NP
      AN=J
      IF(AN.LE.0.0001) GO TO 20
      FAC=PSF(AN,TASA/K)
      COACT=COACT+INV (J)*FAC
      WRITE(6,900) NP,AN,INV(J),FAC,COACT
  900 FORMAT (I10,4F12.2)
  901 FORMAT(20H   CALCULO DE COSTOS)
   20 CONTINUE
      RETURN
      END
```

REENTRANT FOR

```
      SUBROUTINE BENEF(BENE,NB,TISA,RET,BEN)
      DIMENSION BENE(10,4)
C     BEN=BENEFICIO ACTUALIZADO
C     PARA EL BENEFICIO I
C     BENE(I,1)= PERIODO EN EL QUE SE OBTIENE EL PRIMER BENEFICIO
C     BENE(I,2)= NO. DE BENEFICIOS IGUALES INCLUYENDO EL PRIMERO
C     BENE(I,3)= IMPORTE DEL BENEFICIO
C     RET =RETRAZO
C     NB = NO. DE BENEFICIOS
C     TASA=INTERES DE ACTUALIZACION
C     LOS BENEFICIOS SE SUPONEN AL FINAL DEL PERIODO
C     PARA FRACCIONES DE PERIODO SE HACE UNA INTERPOLACION LINEAL
      WRITE(6,901)
      BEN=0
      DO 20 I=1,NB
      TASA=TISA/BENE(I,4)
      TIEM=BENE(I,2)
      PRO=PSA(TIEM,TASA)*BENE(I,3)
      PRI=PRO
      ANV=BENE(I,1)+RET
      PRO=PRO*PSF(ANV,TASA)
      BEN=BEN+PRO
      WRITE(6,900) I,TASA,TIEM,PRI,PRO,BEN,BENE(I,3)
  900 FORMAT(I10,6F12.3)
  901 FORMAT(24H   CALCULO DE BENEFICIOS)
   20 CONTINUE
      RETURN
      END
```

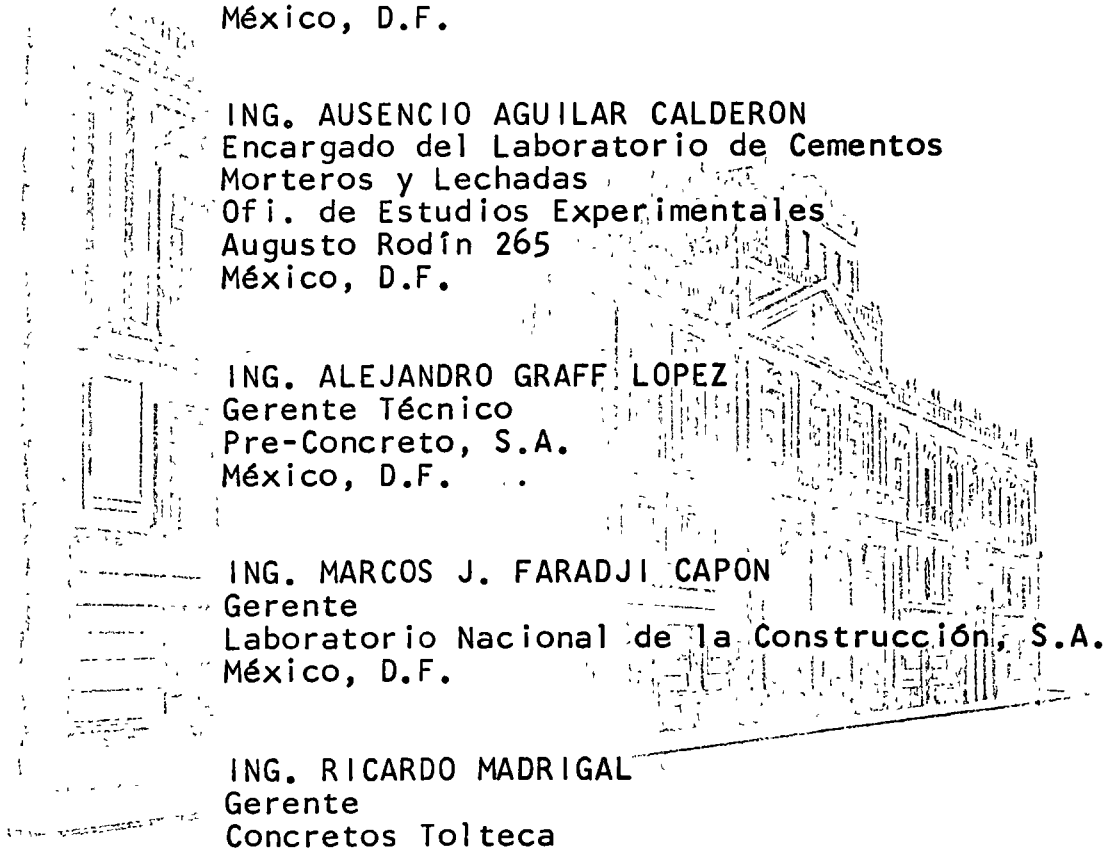
REENTRANT FO
NONREENTR

```
      SUBROUTINE PINTA(RES,RI)
      DIMENSION RES(10,3)
C     RES(I,1)=TASA
C     RES(I,2)=COSTO
C     RES(I,3)=BENEFICIO
      WRITE(6,901)
      WRITE(6,902)
      WRITE(6,904)
      WRITE(6,902)
      WRITE(6,905)RI
      WRITE(6,902)
      WRITE(6,901)
      WRITE(6,906)
      WRITE(6,901)
      DO 100 I=1,10
      A=RES(I,3)-RES(I,2)
  901 FORMAT(5X,45H*********************************************)
  902 FORMAT(5X,1H*,43X,1H*)
  903 FORMAT(5X,1H*,F5.3,F7.2,F8.2,F7.2,F7.2,F8.0,1X,1H*)
  904 FORMAT(5X,1H*,8X,19HE V A L U A C I O N,16X,1H*)
  905 FORMAT (5X,1H*,5X,27HTASA DE RENDIMIENTO INTERNO,F6.3,1H ,4X,1H*)
  906 FORMAT(5X,1H*, 44H TASA  COSTO   BENEF    B-C     B/C   APROV.*)
      B=RES(I,3)/RES(I,2)
      C=RES(I,2)/RES(I,3)*100
      WRITE(6,902)
  100 WRITE(6,903) (RES(I,J),J=1,3),A,B,C
      RETURN
      END
```

NONREENT-

```
      FUNCTION ASP(AN,TA)
      IF(TA.LE.0.00001) GO TO 10
      NA=AN
      AM=NA
      IF(NA.NE.0) GO TO 30
      F1=1.0
      GO TO 11
   30 CONTINUE
      F1=TA/(1.0-1.0/(1.0+TA)**NA)
   11 CONTINUE
      F2=TA/(1.0-1.0/(1.0+TA)**AM)
      ASP=F1+(F2-F1)*(AN-AM)
      RETURN
   10 ASP=0
      RETURN
      END
```

PROFESORES DEL CURSO FABRICACION, COLOCACION Y CONTROL DEL CONCRETO



ING. PEDRO LUIS BENITEZ ESPARZA
Jefe de la Div. Trituración y Asfalto
Industria del Hierro, S.A.
Minería 145 Edif. A 3 Piso
México, D.F.

ING. AUSENCIO AGUILAR CALDERON
Encargado del Laboratorio de Cementos
Morteros y Lechadas
Ofi. de Estudios Experimentales
Augusto Rodín 265
México, D.F.

ING. ALEJANDRO GRAFF LOPEZ
Gerente Técnico
Pre-Concreto, S.A.
México, D.F.

ING. MARCOS J. FARADJI CAPON
Gerente
Laboratorio Nacional de la Construcción, S.A.
México, D.F.

ING. RICARDO MADRIGAL
Gerente
Concretos Tolteca
Div. Bombeo
México, D.F.

ING. VICTOR MANUEL MENA FERRER
Jefe de la Sección de Materiales
Oficina de Estudios Experimentales
C. F. E.
México, D.F.

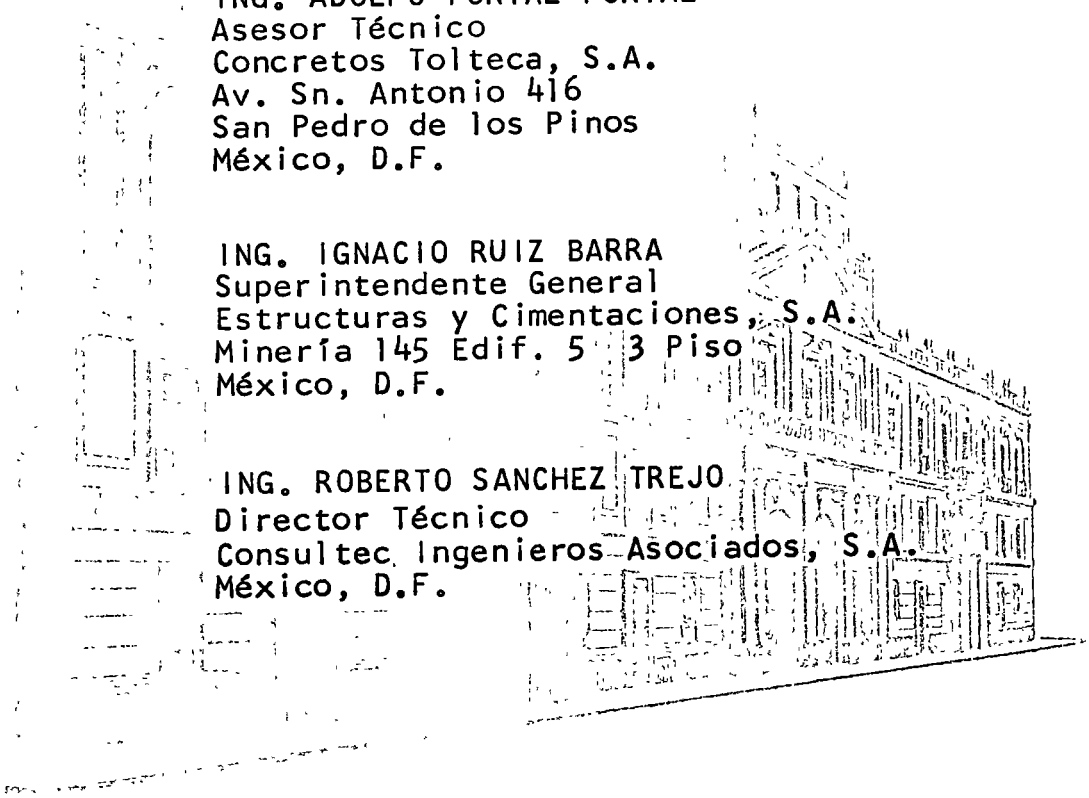
PROFESORES DEL CURSO DE FABRICACION, COLOCACION Y CONTROL DEL CONCRETO

ING. CARLOS JAVIER MENDOZA ESCOBEDO
Jefe de Laboratorio de Materiales
Fac. de Ingeniería
Investigador y Profesor

ING. ADOLFO PORTAL PORTAL
Asesor Técnico
Concretos Tolteca, S.A.
Av. Sn. Antonio 416
San Pedro de los Pinos
México, D.F.

ING. IGNACIO RUIZ BARRA
Superintendente General
Estructuras y Cimentaciones, S.A.
Minería 145 Edif. 5 3 Piso
México, D.F.

ING. ROBERTO SANCHEZ TREJO
Director Técnico
Consultec Ingenieros Asociados, S.A.
México, D.F.



DIRECTORIO DE ASISTENTES DEL CURSO DE FABRICACION, COLOCACION Y CONTROL DEL CONCRETO (del 5 de junio al 5 de julio de 1972)

	NOMBRE Y DIRECCION	EMPRESA Y DIRECCION
1.	ING. GUILLERMO CAMPOS IBARRA Rafael Buelna No. 383 Ote. Culiacán, Sin	UNIVERSIDAD AUTONOMA DE SINALOA Andrade y Constitución Culiacán, Sin
2.	ARQ. PEDRO CORDERO RAMIREZ Norte 81-A No. 374 Col. Electricistas México 16, D. F.	CONTRATISTAS DE INGENIERIA Y ARQUITECTURA, S. A. Martín Mendalde No. 1260 Col. del Valle México 12, D. F.
3.	C.P. MARCO ANTONIO DAMIAN ADAN Av. Oriente 83 No. 3720 Col. La Joya México, D. F.	ECISA CONSTRUCCIONES, S. A. Av. Chapultepec No. 511-105 México 6, D. F.
4.	ING. LUIS AMANDO GARCIA CHOWELL Edif. 22 Depto 301 Villa Olímpica, Tlalpam, D. F.	SECRETARIA DE OBRAS PUBLICAS Xola y Av. Universidad México, D. F.
5.	ING. JOSE GARCIA LOZANO Calle 10-B No. 13 México, D. F.	TUNEL, S. A. Minería No. 145 Col. Escandón México 18, D. F.
6.	SR. JUAN GAUTIER RODRIGUEZ Sur 73-A No. 133 Col. Prado México, D. F.	LG. CONSTRUCTORA, S. A. DE C. V. Liverpool No. 89-602 Col. Juárez México, D. F.
7.	ING. JORGE GIL MERINO Guanajuato No. 143-302 México 4, D. F.	COMISION FEDERAL DE ELECTRICIDAD Augusto Rodín No. 265 México 19, D. F.
8.	SR. PEDRO GOMEZ COLIO Unidad Cuitlahuac Edif.75 D-201 México 16, D. F.	SECRETARIA DE OBRAS PUBLICAS Xola y Av. Universidad México, D. F.
9.	ING. FRANCISCO HERMOSILLO SILVA P. Elias Calles No. 1328-403 México, D. F.	PRODUCTOS DE BASALTO, S. A. DE C.V. Laminador No. 37 Col. Bellavista México, D. F.

DIRECTORIO DE ASISTENTES DEL CURSO DE FABRICACION, COLOCACION Y CONTROL DEL CONCRETO (del 5 de junio al 5 de julio de 1972)

	NOMBRE Y DIRECCION	EMPRESA Y DIRECCION
10.	ING. FERNANDO IBARRA NUÑEZ Lago Cuitzeo No. 212-17 México, D. F.	INSTITUTO MEXICANO DEL SEGURO SOCIAL Toledo 21 P.Baja México, D. F.
11.	ING. ALBERTO LOPEZ CASTAÑON 5 de Mayo No. 35-A Tepeji del Rio,Hgo.	TUNEL, S. A. Minería No. 145 Col. Escandón México 18, D. F.
12.	ING. FRANCISCO JAVIER LOPEZ D. Real del Monte No. 22 México 2, D. F.	C.T.S. CONCRETOS, S. A. Manzanillo No. 109 Col. Roma Sur México 7, D. F.
13.	ING. DAVID MARTINEZ EGUILUZ Florencio Antillon No. 23 México 14, D. F.	CONCRETOS TOLTECA, S. A. Prolongación Avenida San Antonio No.461 Col. San Pedro de los Pinos México, D. F.
14.	SR. ARMANDO M. MARTINEZ GARAY Las Flores No. 319 Tlacopac, San Angel México, D. F.	MAQUINARIA PANAMERICANA, S. A. DE C.V. Blud. M. Avila Camacho No. 245-E México, D. F.
15.	ING. IGNACIO PEDROZA SERRANO San Francisco No. 1814-C México, D. F.	CONDOMINIOS E INVERSIONES, S. A. Cumbres Acultzingo No. 67-101 México 12, D. F.
16.	ING. FRANCISCO PUERTOS CERVANTES Uxmal No. 388-9 Col. Narvarte México 12, D. F.	CONSTRUCCIONES CONDUCCIONES Y PAVIMENTOS Minería No. 145 Col. Escandón México 18, D. F.
17.	ING. FCO. ARMANDO RANGEL ORDOÑEZ Calle E No. 1 Col. Educación México, D. F.	SECRETARIA DE OBRAS PUBLICAS Xola y Av. Universidad México, D. F.
18.	ING. ANTONIO ROLON ORTIZ Millel No. 45 México 12, D. F.	TUNEL, S. A. Minería No. 145 Col. Escandón México 18, D. F.

DIRECTORIO DE ASISTENTES DEL CURSO DE FABRICACION, COLOCACION Y CONTROL DEL CONCRETO (del 5 de junio al 5 de julio de 1972)

	NOMBRE Y DIRECCION	EMPRESA Y DIRECCION
19.	ING. LEOPOLDO RODRIGUEZ GOMEZ Rastro 22 México 22, D. F.	POR CUENTA PROPIA Rastro 22 México 22, D. F.
20.	SR. ADOLFO PORTAL DE LOS RIOS Carlos J. Meneses No. 4 Gto. Musicos Cd. Satélite México, D. F.	CONCRETOS TOLTECA, S. A. Av. San Antonio No. 461 Col. Mixcoac México, D. F.
21.	ING. RAUL RUIZ PEREZ Libertad No. 10-107 Col. Portales México, D. F.	CONSTRUCTORA TECNICA EDIFICADORA Insurgentes Sur No. 1160-302 México, D. F.
22.	ING. LEOPOLDO SALAZAR TORRES Fresno No. 91 México, D. F.	TECNICOS ASOCIADOS, S. A. Av. Juárez No. 157-105 México, D. F.
23.	SR. JORGE SALAZAR TERRON Av. Universidad 1810 Edificio H-Dpto- 4 México, D. F.	CONSTRUCCIONES CONDUCCIONES Y PAVIMENTOS, S. A. Minería No. 145 Edificio 4 2o. Piso Col. Escandón México 18, D. F.
24.	ING. JULIO FRANCISCO SALINAS V. Marti No. 125-201 México, D. F.	CAMINOS Y VIAS, S. A. Benjamin Franklin No. 235-4 México, D. F.
25.	ING. CARLOS TAGLE M. Cantil No. 52 Lomas de Bellavista Cd. Satélite Edo. de México	ADICIONANTES PARA CONCRETO, S. A. Salamanca No. 102-602 Col. Roma México 7, D. F.
26.	ING. MARIO TENA BERNAL Cda. de Miguel Negrete No. 5 México, D. F.	SECRETARIA DE OBRAS PUBLICAS Xola y Av. Universidad México, D. F.