DOCUMENT RESUME

ED 050 592 EM 008 952

TITLE	Computer Club Information Kit for Schools.
INSTITUTION	Association for Educational Data Systems, Washington, D.C.
SPONS AGENCY	National Education Association, Washington, D.C.
PUB DATE	[71]
NOTE	32p.
AVAILABLE FROM	The Association for Educational Data Systems, 1201 Sixteenth Street, N.W., Washington, D.C. 20036 ($1.00)
EDRS PRICE	EDRS Price MF-$0.65 HC Not Available from EDRS.
DESCRIPTORS	After School Activities, *Clubs, *Cocurricular Activities, Computers, *Computer Science Education, *Guides, High School Students, Programing

ABSTRACT

A computer club can provide a group of high school students with an opportunity to satisfy their curiosity about computers and computer programing. This guide offers suggestions for organizing and running such a club. The objectives of a computer club and the duties of the club's sponsor are outlined. Some activities for the first few meetings are suggested. An elementary discussion of the basic concepts of computers and computer programing is presented along with a set of programing problems which require a rudimentary knowledge of FORTRAN. A bibliography of books about computers and FORTRAN programing and a sample computer club constitution are also included. (JY)

TABLE OF CONTENTS

STATEMENT OF PURPOSE OF INFORMATION KIT

In the mid-1950's, computers began to make their appearance on college campuses. Today computers are starting to appear on our high school campuses. Aiding and abetting this emergence of computers in the high schools are the National Science Foundation, sponsoring summer and in-service institutes on computer science for teachers; and local business firms, making Saturdays and evening time on their computers available to high schools at nominal costs. Regional educational computer centers and some state universities are providing free computer time for high school computer science classes and/or organized clubs. Moreover, commercial organizations, and universities with large computer systems are offering time-sharing systems with typewriter-type terminals located in high school buildings.

It is the purpose of this information kit to provide suggestions, materials, and encouragement in seeking how every high school can bring, to those students interested, an opportunity to explore computer science.

Marion Vande Wall

I. DEFINITION OF A COMPUTER CLUB

What is a computer club? A computer club can be defined as a group of students, whose common interest is to seek answers to the following questions :

1. What is a computer?
2. How do humans make a computer do the things they want it to do?
3. Can I learn how to program a computer?
4. What effects does the computer presently have upon society? and how about the future?
5. Would I like to consider computer science as a vocation?

To make a definition by a series of questions may seem confusing. If there is a sufficient number of students interested in the answers to the above questions, and no formal computer science course is offered, a computer club in your school can fill the void.

II. STATEMENT OF OBJECTIVES OF CLUB

The objectives of a school computer club are:

1. To know by name and understand the function of the components of a typical computer system.
2. To learn an orderly process of preparing a problem for a computerized solution.

3. To become knowledgeable of the major rules of a programming language.

4. To become aware of the use of computers in society, and in the school's immediate community.

III. DUTIES OF THE SPONSOR

Usually the Mathematics Department will assume the sponsorship of a computer club. It can easily involve more than one person; business education teachers should also be considered.

A sponsor should be someone with an interest in the subject, and should not be afraid to admit his limitations. The best position a sponsor can take is one of an activities coordinator; maybe doing a little teaching, but mainly helping to make materials, personnel, and a computer facility available to the members. There will be club members that will quickly gain expertise in programming language, and the sponsor should not take exception to this. He should stand back and give these people encouragement and suggest new and challenging programs for the members to write.

Contact with local businesses or school facilities for use of equipment will be one of the largest responsibilities of the sponsor.

IV. SUGGESTED ACTIVITIES

The initial meeting of the club should be well planned and filled with activity--a guest speaker from a local computer center would add greatly to help give the membership a perspective of computer science. Computer vendors are helpful in providing speakers and films with no obligation, and companies are most cooperative in the educational field. Area educational computer center personnel are excellent resource people to call upon to kick-off the first club meeting. An outstanding film, "The Computer Revolution," a product of the TV series Twenty-First

Century, would be a fine start if there are no available guest speakers. This film is available through Marion Vande Wall, ACCESS Computer Center, 112 Eleventh Street, Des Moines, Iowa, 50309.

A visit to a computer center could follow the first meeting. Most businesses operating a computer center are happy to have young people interested in computers, and will usually provide an information tour of their facility. Here again, do not overlook an area educational computer center for a tour.

The third meeting could be dedicated to the analysis of a problem for a computerized solution. At this point, a professional programmer, or systems analyst, could be used as a guest lecturer and give the club members an insight into how they approach using a computer to perform the tasks which have confronted them.

The interest of the group and success of the first three meetings will give direction to the activities from this point forward. Instruction of a programming language could begin at this point through programmed-instruction materials or lectures. Possible instructors could come from an earlier established club in a neighboring school. Do not overlook the utility of students teaching other students. As has been indicated, the club sponsor need not be an authority. More practically, he is a talent merchant to bring to this club the opportunity for exploring the field of computer science.

The following article "What is a Computer? And Programming" is intended to give club sponsors a working definition of a computer and the steps involved in preparing the instructions for a computer to follow in solving a problem or processing data.

V. WHAT IS A COMPUTER? AND PROGRAMMING

This unit is designed to present the basic concepts of what a computer is and how it can be instructed to perform the operations we so choose.

What is a Computer?

Undoubtedly a computer is different things to different people. Let us say that a computer is an unusually fast and versatile machine for processing data. It can accept both the data to be processed and instruction to direct the processing.

In short, a computer is something that will accept information, process it, and give results.

The types and number of computers in existence seem almost endless. They vary in size, cost and type of application. However, a comprehensive description of the general purpose, stored program, digital, electronic computer will follow.

General purpose is used as opposed to special purpose. A special purpose machine may be applied to only one given purpose with the advantage that it may perform that given task swiftly, efficiently, and economically but with the disadvantage that it has no versatility. A gunnery computer, for example, could not be used effectively to keep an inventory. The general purpose computer, on the other hand, is versatile and can do both, but with the disadvantage that it may perform with less speed, efficiency, and economy than a device designed for one specific purpose.

Stored program is the most important computer characteristic. The term describes the concept of storing within the computer's memory a program: the detailed list of instructions which the computer is to carry out in the solution of a problem or in handling data. The instructions are coded in a form the computer is designed to accept. Each instruction calls upon the computer to perform one of the operations for which it is designed.

A program may contain instructions telling the computer to do operations relating to the data or relating to the instructions themselves. Thereby, a computer is capable of modifying a given instruction, or by selection, through the computer's logical ability, altering the sequence of performance.

The stored program concept provides not only adaptable and automatic processing of data, but unlimited versatility as well, since a computer can be directed to carry out different procedures by placing different programs in memory.

Digital and analog are roughly the two broad classes of computing devices. The two classes of machines have different uses and frequently supplement one another. Some recent designs include both types in a single system which is called a hybrid computer.

Digital is associated with operations of discrete steps and with discrete symbols of information. So, in a digital computer, computation is done with symbols representing digits and with manipulation of these symbols by rules of arithmetic.

In an analog computer, some physical quantity such as voltage or temperature is considered analogous to the magnitude of the numbers under consideration. The common example of an analog device is the slide rule.

The advantages of digital computers are the stored program capabilities and precision characteristics. The advantages of analog machines are lower cost and high speed in obtaining final answers.

Electronic is used to connote speed, an important aspect of computers. It is true that a completely electronic computer does not exist; there are still some push buttons and spinning tape reels or disks but the major operations are electronic. The earlier machines used mechanical relays as the switching elements; then came vacuum tubes, which permitted switching rates 10,000 times faster. The machines produced now are highly transistorized and have megacycle switching rates (one million cycles a second).

The phrase "general purpose, stored program, digital, electronic computers" still covers a broad class of computers. Some sell for $10,000 while others cost over $5,000,000; some are the size of a refrigerator while others require a large building; some do numerical problems and process payrolls while others play checkers and solve geometric theorems.

In a general sense, all computers are blood brothers in that, from a functional point of view, they may be divided into five areas of operation: input, output, memory, arithmetic-logic, and control. Sometimes, in a physical sense, these units are hard to identify. Machine designers are notorious for the use of a piece to do double duty, but functionally, these units do exist in all digital computers.

Functional Units

The schematic diagram of the main functional units of a computer serves as visual description of a typical computer and not any unique machine of a specific manufacture. (Figure I)

The input unit provides a means of entering information into the computer's memory. This information may be data or instructions which have been coded in some form acceptable to the machine.

The card reader is the input unit through which cards are "read," that is, the information recorded on the card is sensed and converted to electric pulses which are transmitted to the computer's memory.

A magnetic tape unit is used for both input and output. Information recorded on the surface of the tape is read as the tape is moved past a reading head, similar in theory to the reading head of a tape recorder.

The magnetic properties of flexible tape are carried onto another input-output device entitled random access magnetic disk. Information is recorded on rotating disks resembling several phonograph records stacked on a spindle. The random access feature permits locating information placed on the surfaces randomly as opposed to the tape exposing only one segment at a time.

Documents printed with magnetic ink are "read" in much the same way magnetic tape is read. Today, banks make wide use of magnetic characters, particularly on checks.

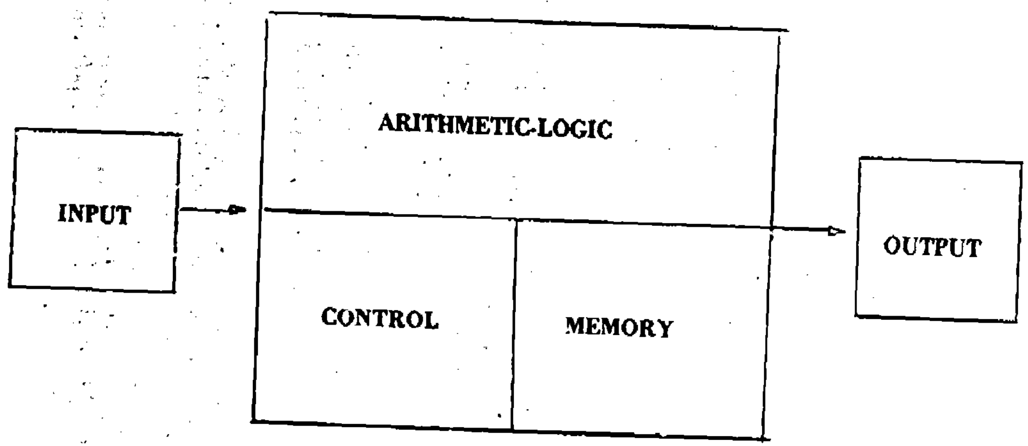


Figure 1

If all numbers and words, not just those printed in magnetic ink, could be read by computers, it would save much of the bother of translating information into special codes for computer input. The computer industry is working toward this kind of versatility. An optical manner has a set of electronic patterns in it and a photoelectric cell that scans material to be read, converting characters into electronic pulses.

The Central Processing Unit is divided into three sub-units (memory, control, arithmetic logic), each with a specific function. All information the computer receives from its input devices first goes into the memory portion of the central processing unit. The memory capacity is the feature that most distinguishes a computer from other calculating machines.

The computer's memory can be compared to a block of post office boxes, where each box holds one person's mail. In the computer's memory, each "box" holds one item of information, either data or an instruction.

Magnetic core memories were an important development. The individual magnetic cores are doughnut-shaped, the size of pinheads, and made of ferro-magnetic materials. The cores are strung like beads into frames which are called planes, that look something like tennis rackets. The frames are stacked up one on top of the other to make a basic memory unit.

Each core can be magnetized in either a clockwise or counterclockwise direction, signifying a "bit" or "no bit" condition. The cores are arranged in columns and each column is assigned an address. The address is used to obtain the data or instruction stored.

The cores under each address are connected separately in the computer's control section. Information (data or instructions) can instantly be "read out" of any address and used in working problems.

The control unit is considered "the brain of the brain." Control includes all of the circuitry necessary to perform all functions throughout the computer.

One instruction at a time is taken in sequence from the program stored in the memory to the control unit where it is interpreted and then executed. The meaning of an instruction and the procedure to be followed in carrying it out is implicit in the control unit's circuitry.

In receiving, interpreting, and executing instructions, the control unit causes the operation of the input, the output, the memory, and the arithmetic-logic units; it governs the transmission and storage of all information; and it coordinates the functioning of the various parts of the computer.

The arithmetic-logic unit functions as the desk calculator of the computer. It performs the arithmetic operations: addition, subtraction, multiplication, and division.

In addition to doing arithmetic, the arithmetic-logic unit can perform simple logical operations; that is, it can examine a number and distinguish whether it is zero or not and it can examine the sign of a number and distinguish whether it is positive or negative. Actually, all logic is based upon the computer's ability to discriminate between 0 and 1.

Output constitutes the finished results, and may be given in any of several forms, such as the already discussed punched card, magnetic tape, or direct access disks. There are forms that are exclusive to the output function, such as plotters, cathode ray tube display, and high-speed printer.

In summary, a computer can bring information in to memory through some input device, it can transmit information between the memory and internal processors which do simple logic and simple arithmetic operations, and it can transmit information from the memory to some output device. It can do all this with remarkable speed and with a high degree of reliability. A large-scale computing system can perform three million operations a second. This concept is staggering. Even a small inexpensive machine can do five hundred operations a second.

In the field of computation, the computer now permits calculations that were literally impossible before. Not only would such jobs require too many human operators, but the errors that would creep in would make the results doubtful. If you ever tried to add up one hundred numbers, you know it is not a simple task. By the time you are at the ninetieth number, you either transpose two digits or enter a number twice (or worse, think you have), and then you start over. A computer has no such difficulties.

In this unit, we have defined the computer and presented the general description of a typical computer. The following unit illustrates the procedures generally followed in solving a problem or handling data utilizing a computer.

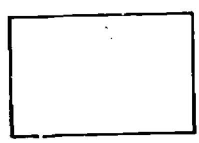
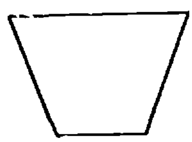
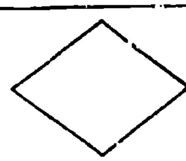
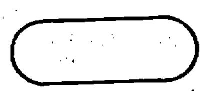
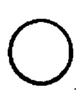
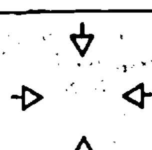
PROGRAMMING

In solving any kind of problem, a computer is helpless until it has been given a detailed set of instructions. The computer cannot figure out how to solve a problem. All it can do is slavishly follow directions step by step until the job is completed. This is true for any problem we try to solve with a computer, whether it's simply adding a column of numbers, or solving the complicated equations necessary to send a missile to the moon. These detailed instructions are called the computer program.

To write a program we must first understand the problem to be solved. We must analyze it and break it down into parts.

One way to do this is to construct a flowchart of the problem. This is a line diagram that shows each part of the total problem and its logical relationship to every other part. The flowchart is similar to the diagram of a pass play that a football coach draws on the blackboard, or to a pattern for making a dress. It shows component parts of the problem just as a play diagram indicates each blocking assignment. And it shows how all the parts fit together just as the pattern indicates where two pieces of cloth are to be sewn together.

The symbols shown are recommended for use in program flowcharts.

	PROCESSING: A group of program instructions which perform a processing function of the program.
	INPUT/OUTPUT: Any function of an input/output device.
	DECISION: The decision function used to document points in the program where a branch to alternate paths is possible based upon variable conditions.
	TERMINAL: The beginning, end, or a point of interruption in a program.
	CONNECTOR: An entry from, or an exist to, another part of the program flowchart.
	OFFPAGE CONNECTOR: A connector used instead of the connector symbol to designate entry to or exit from a page.
	FLOW DIRECTION: The direction of processing or data flow.

Flowcharts are used to illustrate many a relatively non-scientific example as found in Figure 2.

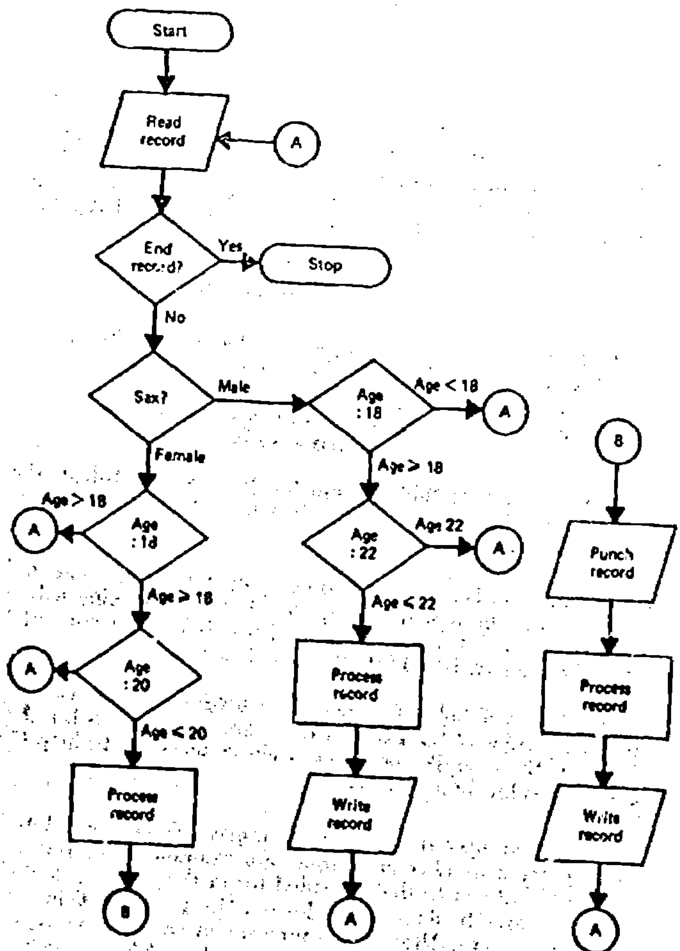
Once the problem has been mapped out in a flowchart, and broken down into individual simple operations, we must write the instructions telling the computer how to handle each operation. But here we run into difficulty; the computer does not understand English. So, we have to write our instructions in a language, or code, that the computer understands.

Computer languages currently exist at three broad levels--absolute machine language, symbolic language, and problem-oriented language. The first two levels are machine-oriented, and each computer designed has its own unique machine language.

If we were to write programs directly in absolute machine language, it would be necessary for us not only to express each instruction in the unique language of that machine, but also keep track of exactly where everything is stored in the memory portion of the computer. In the case of small programs, this is merely a pain in the neck. With larger programs, it becomes nearly impossible. Consequently, programming languages, machine-oriented and problem-oriented, have been devised to allow programmers to write in a form more closely related to the spoken vernacular. It is these two groups that form the assembler and compiler programming languages.

The symbolic machine languages are written for a special type of computer and as such supplies mnemonic counterparts for each machine language instruction. A special program, called an assembler, provides the go-between which produces machine language instructions from mnemonic commands. The instructions written in some programming language we call the source program. Through the actions of the assembler, an object program is produced. It is then the object program that is used for execution of the activity originally designed as the problem to be solved or data to be processed.

There are several advantages of an assembler system over machine language; the programmer is required to write less, the detail is reduced, and the language is easier to comprehend.



Start
Read record
A
End record?
Yes
Stop
No
Sex?
Male
Female
Age : 18
Age < 18
A
Age > 18
B
Age > 18
A
Age : 18
Age ≥ 18
Age : 22
Age 22
A
Punch record
Age ≤ 22
A
Age : 20
Age < 20
Process record
Process record
Process record
Write record
Write record
B
A
A

Figure 2

Problem-oriented languages are also referred to as higher-order languages. Rather than having the programmer adapt himself to the individual computer, these languages adapt the computer to the programmer. The coding rules are designed to assist the programmer in defining the computer processing to be performed. The translation of this coding into a machine-language program is performed by a set of routines called a compiler.

There are many different compiler languages. Each language consists of a set of allowable words and symbols, and a set of rules for using this vocabulary to define problems or problem-solving procedures. Some of the common languages with their acronyms are:

ACRONYMS	MEANING
COBOL	Common Business-Oriented Language
FORTRAN	Formula Translator
ALGOL	Algorithmic Language
BASIC	Beginner's All-Purpose Symbolic Instruction Code

Once the coding of the problem has been accomplished, the program, line by line, is keypunched to make the statements machine readable.

The punched cards represent a source program at this point, and are processed through the computer. If we are dealing with a problem-oriented language such as FORTRAN, the computer will make the translation into machine language.

The program will fail to translate if the programmer has violated any of the rules of the FORTRAN language. Since this is rather common, the compiler program produces messages to help the programmer locate his errors.

The program may successfully be translated, but may fail to execute for a number of reasons. For example, the input data may not conform to that provided for in the program; and the computer may be directed to do something it cannot do, such as divide by zero. These errors indicate an error in the logic of the solution and need to be reconciled. This process of elimination of errors is termed debugging.

Once a program is debugged, it is then ready for use. The intended use of course is to solve the defined problem.

In summary, the steps in the solution of a problem using a computer are: (1) clearly define the problem, (2) flowchart the logic of the solution, (3) code the solution using a programming language, (4) produce the coding in a machine-readable form (keypunch), (5) translate the coded program into machine-language (compiling), (6) debug, if necessary, and (7) process with data to obtain results.

A list of twenty-five practice problems follows. These problems were first used by the Illinois Institute of Technology in Saturday classes for secondary students of the Chicago area. The problems are well designed in degree of difficulty and will serve as an excellent source of applicable problems for beginning programmers to use.

PRACTICE PROBLEMS

All of the problems contained in this series can be solved through FORTRAN programming. Only the most elementary knowledge of the FORTRAN language is required.

Some of these problems require a knowledge of first year high school algebra, but most problems require no mathematical training.

An attempt has been made to arrange the series in order of difficulty; however, since the difficulty of any problem depends on the background and aptitude of the person attempting to solve it, our arrangement should be considered with reservations. Therefore, you should not assume that difficulty with a particular problem will imply greater difficulty with succeeding problems.

1. **Area of a square**

   Find the area of any square by the formula $A = S^2$.

2. Area of a rectangle

Find the area of any rectangle by the formula A = L.W

3. Volume of a cylinder

Find the volume of any cylinder when the radius of the base and the height of the cylinder are known.

The formula is $V = (3.14)(R^2)(H)$

4. Squares of integers 1 to N

Write a program which will generate a two column table showing, in the first column, the integers 1 to N, and in the second column the square of each integer. Do not use an integer greater than 25 for N.

5. Daily Pay

A man takes a job for 30 days. His pay for the first day is $.01, his pay for the second day is double or $.02, for the third day is $.04, for the fourth day $.08. Each day's pay is determined by the previous day's pay multiplied by 2. The program should generate a table which will show his pay for each of the thirty days.

6. Fahrenheit to Centigrade table

The formula for converting Fahrenheit values to Centigrade is:

$$C = \frac{5}{9}(F-32)$$

Write a program that will generate a two column table showing, in the first column, all the even Fahrenheit values from 2 to 50 inclusive and in the second column each corresponding Centigrade value. Assume that the Centigrade values are to be expressed to 4 decimal places.

7. Finding the mean of 10 numbers

Determine the sum of 10 numbers and divide by 10 to obtain the mean or average of the numbers.

8. Compound interest

The formula for compound interest is $A = P\left(1 + \frac{I}{C}\right)^{NC}$
where:
A represents the amount (principal + interest)
N represents the number of years
C represents the number of conversions per year
I represents the yearly rate of interest
P represents the principal

The program should determine the value of A when the other values are known.

9. Sum and last term of A. P.

Find the sum and last term of any arithmetic progression
Let A represent the first term
Let D represent the common difference
Let N represent the number of terms
Let L represent the last term
Let S represent the sum of the terms
Then the formula for the last term is $L = A + D (N - 1)$

The formula for the sum of the terms is $S = \frac{N (A + L)}{2}$

Example: The progression is: 1, 3, 5, 7, 9
Therefore $A = 1$
$D = 2$
$N = 5$

$L = A + D (N - 1)$ $S = \frac{N (A + L)}{2}$

$L = 1 + 2 (5 - 1)$ $S = \frac{5 (1 + 9)}{2}$

$L = 1 + 8$ $S = \frac{50}{2}$

$L = 9$ $S = 25$

10. Sum and last term of G. P.

Find the sum and last term of any geometric progression.

Let I represent the last term
Let A represent the first term
Let N represent the number of terms
Let R represent the common ratio
Let S represent the sum of the terms

Then the formula for the last term is $L = AR^{(N-1)}$

The formula for the sum of the terms is $S = \frac{A - RL}{1 - R}$

Example:
The progression is: 2, 6, 18, 54
Therefore $A = 2$
$R = 3$
$N = 4$

$L = AR^{(N-1)}$ $S = \frac{A - RL}{1 - R}$

$L = 2.3^{(4-1)}$ $S = \frac{2 - 3.54}{1 - 3}$

$L = 2.3^3$ $S = \frac{2 - 162}{-2}$

$L = 2.27$ $S = \frac{-160}{-2}$

$L = 54$ $S = 80$

11. Solution of $AX^2 + BX + C = 0$

Solve any second degree equation in one variable. The quadratic formula is

$$X = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A}$$

12. Solution of AX + BY = C and DX + EY = F

Write a program that will solve any pair of simultaneous equations in two variables:

$$AX + BY = C$$
$$DX + EY = F$$

13. Finding the divisors of an integer

Write a program which will list all the integers which are divisors of a chosen integer.

14. Finding primes 1 to 100

Write a program that will generate a list of all the prime numbers from 1 to 100.

15. Illegal cancellation

The equation $16/64 = \frac{1}{4}$ is a result which can be obtained by the cancellation of the 6 in the numerator and denominator. Find all the cases in which AB/BC = A/C for A, B, and C integers between 1 and 9 inclusive. Do not consider obvious special cases such as 22/22, 33/33, etc.

16. Four digit numbers

The four digit number 3025 has the following property: if the number formed by considering only the first two digits (30) is added to the number formed by considering only the last two digits (25), (the total will be 55), and if this number (55) is squared, the result will be the original number.

$$(55)^2 = 3025$$

Find all 4 digit numbers having this property. Do not check numbers beyond 9900 since 9901 would be arranged as

$$99 + 01 = 100 \text{ and } (100)^2 = 10000$$

which is a 5 digit number.

17. Loan repayment

A loan of $1,000 is to be repaid at the rate of $100 per month. The interest is charged at the rate of 1% on the unpaid balance each month. The program should generate a four column table showing the payment number, the balance, the interest for each month, and the amount paid on the principal each month. The new balance for each month is obtained by subtracting the amount paid on the principal from the old balance.

18. Mortgage problem

A $17,000 mortgage is to be repaid at the rate of $200 per month. The interest is charged at the rate of 6% each year, calculated each month. The program should be designed to generate a four column table which will show the payment number, the balance, the interest for each month, and the amount paid on the principal for each month. The new balance for each month is obtained by subtracting the amount paid on the principal from the old balance.

19. Addition table

Have the computer print out a table of addition facts. Let each of the two numbers to be added vary from one to ten. The output will appear as follows:

1 + 1 = 2
1 + 2 = 3
.
.
.
1 + 10 = 11
2 + 1 = 3
2 + 2 = 4
.
.
.
2 + 10 = 12
.
.
.
10 + 10 = 20 (Last Line of Output)

20. Payroll problem

Write a program to do a payroll where the input consists of the hourly rate, the number of hours worked, the pension rate, the income tax rate, the amount of bond deduction, and the name of the employee.

The output should include gross pay, pension deduction, bond deduction, income tax deduction, take home or net pay, and employee name.

Note: The name of the employee can be eliminated from the program.

21. Grade point average

In a certain school offering both "HONORS" courses and "REGULAR" courses, the letter grades A, B, C, D, and F are used to indicate achievement. To determine a student's grade point average the following weights are used.

LETTER GRADE	POINT VALUE "HONORS"	"REGULAR"
A	5	4
B	4	3
C	3	2
D	1	1
F	0	0

Write a program that will accept as input the number of "REGULAR" and "HONORS" letter grades received, and exhibit as output the student's grade point average. This program may be written with or without the name of the student.

22. Perfect numbers

In mathematics a number is considered to be perfect if the sum of all of its divisors (except itself) is equal to the original number.

Example: The divisors of 6 are 1, 2, and 3

$$1 + 2 + 3 = 6$$

Write a program that will determine all perfect numbers from 1 to 100, which will exhibit as output all the divisors of the number (except itself) and the perfect number.

23. Sorting problem

Write a program which will arrange in ascending order any number of 3 digit numbers (up to 999). You may assume that no two of the numbers are equal.

24. Farmer problem

A farmer with $100 goes to market to buy 100 head of stock. Prices are as follows: calves, $10 each; pigs, $3 each; chickens, $0.50 each. He gets 100 head for his $100. How many of each does he buy?

25. Monkey problem

There are 3 pirates and a monkey on a desert island who have gathered a pile of coconuts which are to be divided the next day. During the night one pirate arises, divides the pile into 3 equal parts and finds one coconut left over, which he gives to the monkey. He then hides his share away from the pile. Later during the same night, each of the other two pirates, in turn, arise and repeat the performance of the first pirate. In the morning all 3 pirates arise, divide the pile into 3 equal shares and find one left over which is given to the monkey. How many coconuts were in the original pile? Since the result is not unique, find all values from 1 to 1000 which satisfy the conditions.

BIBLIOGRAPHY

The following bibliography will provide a list of books that could be obtained as instructional materials for students in the FORTRAN programming language.

Allen and Bushnell, THE COMPUTER IN AMERICAN EDUCATION. New York: John Wiley & Sons, Inc.

Anderson, D., BASIC COMPUTER PROGRAMMING. New York: Appleton-Century-Crofts, Inc.

Anderson. D., COMPUTER PROGRAMMING IN FORTRAN IV. New York: Appleton-Century-Crofts, Inc.

Arnold, et al, INTRODUCTION TO DATA PROCESSING. New York: John Wiley & Sons, Inc.

Drowder, N., ARITHMETIC OF COMPUTERS. New York: Doubleday.

Darnowski, V., COMPUTERS - THEORY AND USE. Washington, D. C.: National Science Teachers Association.

Davis, G., INTRODUCTION TO ELECTRONIC COMPUTERS. New York: McGraw Hill.

Dorn & Greenberg, MATHEMATICS & COMPUTING - PART 1. New York: John Wiley & Sons, Inc.

Dorn & Greenberg, MATHEMATICS & COMPUTING - PART 2. New York: John Wiley & Sons, Inc.

Dorn & Greenberg, MATHEMATICS & COMPUTING WITH FORTRAN. New York: John Wiley & Sons, Inc.

Freiberger & Prager, APPLICATIONS OF DIGITAL COMPUTERS. New York: Ginn & Company, Inc.

Golde, H., FORTRAN II & IV FOR ENGINEERS & SCIENTISTS. New York: John Wiley & Sons, Inc.

Gruenberger & McCracken, INTRODUCTION TO ELECTRONIC COMPUTERS. New York: John Wiley & Sons, Inc.

Haag, J. N., COMPREHENSIVE FORTRAN PROGRAMMING. New York: Hayden Book Company, Inc.

Haga, E., UNDERSTANDING AUTOMATION. Elmhurst, Illinois: The Business Press.

Hanson & Gilmore, FUNDAMENTALS OF DIGITAL COMPUTERS. Elmhurst, Illinois: The Business Press.

Iverson, K., ELEMENTARY FUNCTIONS (& SOLUTION). Chicago, Illinois: SRA.

Kahn, G., BUSINESS DATA PROCESSING. New York: McGraw Hill & Company.

Korflage, R., LOGIC AND ALGORITHMS. New York: John Wiley & Sons. Inc.

Larsson, R. D., EQUALITIES & APPROXIMATIONS WITH FORTRAN. New York: John Wiley & Sons, Inc.

Laurie, E. J., COMPUTERS. Chicago, Illinois: Southwestern Publishing Company.

Lovis, F. B., CONTEMPORARY SCHOOL MATHEMATICS COMPUTER 1. New York: Houghton Mifflin.

Lovis, F. B., CONTEMPORARY SCHOOL MATHEMATICS COMPUTER 2. New York: Houghton Mifflin.

Marcovitz & Schweppe, ALGORITHMIC METHODS & MAD. New York: Macmillan & Company.

Martin, E. W., ELECTRONIC DATA PROCESSING - INTRODUCTION. New York: R. D. Irvin, Inc.

Mathematics Association of America, COMPUTERS & COMPUTING. American Mathematics Monthly, 2/65.

McCarthy, et al, INTEGRATED DATA PROCESSING. New York: John Wiley & Sons, Inc.

McCracken, D. D., DIGITAL COMPUTER PROGRAMMING. New York: John Wiley & Sons, Inc.

McCracken, D. D., GUIDE TO FORTRAN PROGRAMMING. New York: John Wiley & Sons, Inc.

Murphy, J. S., BASICS OF DIGITAL COMPUTERS. New York: Rider & Company.

NCTM, COMPUTER ORIENTED MATHEMATICS. Washington, D. C.: NCTM.

NCTM, REPORT ON COMPUTER ORIENTED MATHEMATICS. Washington, D. C.: NCTM.

Pennington, R. H., INTRODUCTION TO COMPUTER METHODS & NUMERICAL ANALYSIS. New York: Macmillan & Company.

Prager, William, INTRODUCTION TO FORTRAN & NUMERICAL METHODS. New York: Blaisdell Publishing Company.

Raucher, Schlumpf & Grossman, NEW YORK CITY CURRICULUM GUIDE - COMPUTER MATHEMATICS. New York: New York Board of Education.

Seligsohn, I. J., YOUR CAREER IN COMPUTER PROGRAMMING. New York: Messner (division of Simon & Schuster, Inc.).

S.M.S.G., ALGORITHMS, COMPUTATION & MATHEMATICS. San Francisco, California: Vroman Inc.

S.M.S.G., ALGORITHMS, COMPUTATION & MATHEMATICS - FORTRAN. San Francisco, California: Vroman Inc

S.M.S.G., ALGORITHMS, COMPUTATION & MATHEMATICS - TEACHER EDITION. San Francisco, California: Vroman Inc.

S.M.S.G., ALGORITHMS, COMPUTATION & MATHEMATICS - FORTRAN - TEACHER EDITION. San Francisco, California: Vroman Inc.

S.M.S.G., ALGORITHMS, COMPUTATION & MATHEMATICS - ALGOL - TEACHER EDITION, San Francisco, California: Vroman Inc.

Smith, E., COMPUTER MATHEMATICS. Birmingham, Michigan: Midwest Publications Inc.

Stark, P. A., DIGITAL COMPUTER PROGRAMMING. New York: Macmillan.

Thomas, S., COMPUTERS. New York: Holt, Rinehart & Winston.

Wenner, J. F., HIGH SCHOOL ORIENTATION COURSE IN D.P. Chicago, Illinois: Southwestern Publishing Company.

Williams, W. F., PRINCIPLES OF AUTOMATED INFORMATION RETRIEVAL. Elmhurst, Illinois: The Business Press.

PROPOSED CONSTITUTION

ARTICLE I - NAME

The name of this organization shall be the ____________ Computer Club.

ARTICLE II - PURPOSE

The purpose of this organization shall be fourfold: 1) To know by name and understand the function of the components of a typical computer system; 2) To learn an orderly process of preparing a problem for a computerized solution; 3) To become knowledgeable of the major rules of a programming language; 4) To become aware of the use of computers in society, and in the school's immediate community.

ARTICLE III - MEMBERSHIP

SECTION I

The membership of the club shall be composed of a group of students whose common interest is to seek answers to the following questions: 1) What is a computer? 2) How do humans make a computer do the things they want it to do? 3) Can I learn how to program a computer? 4) What effects does the computer presently have upon society? and how about the future? 5) Would I like to consider computer science as a vocation?

SECTION II

The sponsor or sponsors of this club shall be someone with an interest in computers, who will act as an activities coordinator.

ARTICLE IV - ACTIVITIES

The main activities of the club shall consist of meetings dedicated to computer-related exercises. Some possible activities are: 1) computer films, 2) visits to computer centers, 3) analyses of problems using a computer, 4) instruction in a programming language, 5) instruction in programming, 6) lectures, and 7) practice on a computer.