جامعة المستقبل
AL MUSTAQBAL UNIVERSITY

كلية العلوم
قسم الانظمة الطبية الذكية

Lecture: (2)

INTRODUCTION TO FLOWCHARTS

Subject: Computer Programming (I)

Level: First

Lecturer: Dr. Maytham N. Meqdad

Introduction to flowcharts

Before you start coding a program it is necessary to plan the step by step solution to the task your program will carry out. Such a plan can be symbolically developed using a diagram. This diagram is then called a flowchart. Hence a flowchart is a symbolic representation of a solution to a given task. A flowchart can be developed for practically any job. Flowcharting is a tool that can help us to develop and represent graphically program logic sequence. It also enables us to trace and detect any logical or other errors before the programs are written.

TYPES OF FLOWCHARTS

Computer professionals use two types of flowcharts:
- Program Flowcharts.
- System Flowcharts

1. Program Flowcharts:

These are used by programmers. A program flowchart shows the program structure, logic flow and operations performed. It also forms an important part of the documentation of the system. It broadly includes the following:
- Program Structure.
- Program Logic.
- Data Inputs at various stages.
- Data Processing
- Computations and Calculations.
- Conditions on which decisions are based.
- Branching & Looping Sequences.
- Results.
- Various Outputs.

The emphasis in a program flowchart is on the logic.

2. System Flowcharts:

System flowcharts are used by system analyst to show various Processes, sub systems, outputs and operations on data in a system.

FLOWCHART SYMBOLS

Normally, an algorithm is expressed as a flowchart and then the flowchart is converted into a program with the programming language. Flowcharts are independent of the programming language being used. Hence one can fully concentrate on the logic of the problem solving at this stage. A large number of programmers use flowcharts to assist them in the development of computer programs. Once the flowchart is fully ready، the programmers then write. It in the programming language. At this stage he need not concentrate on the logic but can give more attention to coding each instruction in the box of the flowchart in terms of the statements of the programming language selected.

A flowchart can thus be described as the picture of the logic to be included in the computer program. It is always recommended for a beginner, to draw flowcharts prior to writing programs in the selected language. Flowcharts are very helpful during the testing of the program as well as incorporating further modifications. Flowcharting has many standard symbols. Flowcharts use boxes of different shapes to denote different types of instructions.

The actual instruction is written in the box. These boxes are connected with solid lines which have arrowheads to indicate the direction of flow of the flowchart. The boxes which are used in flowcharts are standardized to have specific meanings. These flowchart symbols have been standardised by the American National Standards Institute. (ANSI.(While using the flowchart symbols following points have to be kept in mind:

-The shape of the symbol is important and must not be changed.
-The size can be changed as required.
-The symbol must be immediately recognizable.
-The details inside the symbol must be clearly legible.
-The flow lines, as far as possible, must not cross.

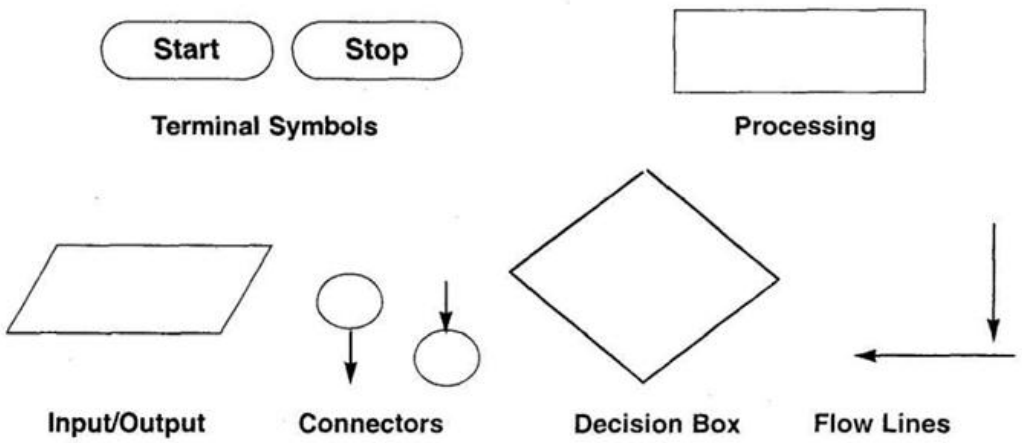


Fig. 1 : Flowchart Symbols

Terminal Symbol:
Every flowchart has a unique starting point and an ending point. The flowchart begins at the start terminator and ends at the stop terminator. The Starting Point is indicated with the word START inside the terminator symbol. The Ending Point is indicated with the word STOP inside the terminator symbol. There can be only one START and one STOP terminator in your entire flowchart. In case a program logic involves a pause, it is also indicated with the terminal symbol.

Input/Output Symbol:
This symbol is used to denote any input/output function in the program. Thus if there is any input to the program via an input device, like a keyboard, tape, card reader etc. it will be indicated in the flowchart with the help of the Input/Output symbol. Similarly, all output instructions, for output to devices like printers, plotters, magnetic tapes, disk, monitors etc. are indicated in the Input/Output symbol.

Process Symbol:
A process symbol is used to represent arithmetic and data movement instructions in the flowchart. All arithmetic processes of addition, subtraction, multiplication and division are indicated in the process symbol. The logical process of data movement form one memory location to another is also represented in the process box. If there are more than one process instructions to be executed sequentially, they can be placed in the same process box, one below the other in the sequence in which they are to be executed.

Decision Symbol:
The decision symbol is used in a flowchart to indicate the point where a decision is to be made and branching done upon the result of the decision to one or more alternative paths. The criteria for decision making is written in the decision box. All the possible paths should be accounted for. During execution, the appropriate path will be followed depending upon the result of the decision.

Flowlines:
Flowlines are solid lines with arrowheads which indicate the flow of operation. They show the exact sequence in which the instructions are to be executed. The normal flow of the flowchart is depicted from top to bottom and left to right.

Connectors:
In situations, where the flowcharts becomes big, it may so happen that the flowlines start crossing each other at many places causing confusion. This will also result in making the flowchart difficult to understand. Also, the flowchart may not fit in a single page for big programs. Thus whenever the flowchart becomes complex and spreads over a number of pages connectors are used. The connector represents entry from or exit to another part of the flowchart. A connector symbol is indicated by a circle and a letter or a digit is placed in the circle. This letter or digit indicates a link. A pair of such identically labeled connectors are used to indicate a continued flow in situations where flowcharts are complex or spread over more than one page. Thus a connector indicates an exit from some section in the flowchart and an entry into another

section of the flowchart. If an arrow enters a flowchart but does not leave it, it means that it is an exit point in the flowchart and program control is transferred to an identically labeled connector which has an outlet. This connector will be connected to the further program flow from the point where it has exited. Connectors do not represent any operation in the flowchart. Their use is only for the purpose of increased convenience and clarity.

ADVANTAGES OF FLOWCHARTS

There are a number of advantages when using flowcharts in problem solving.They provide a very powerful tool to programmers to first represent their program logic graphically and independent of the programming language.

-Developing the program logic and sequence. A macro flowchart can first be designed to depict the main line of logic of the software. This model can then be broken down into smaller detailed parts for further study and analysis.

-A flowchart being a pictorial representation of a program, makes it easier for the programmer to explain the logic of the program to others rather than a program

-It shows the execution of logical steps without the syntax and language complexities of a program.

-In real life programming situations a number of programmers are associated with the development of a system and each programmer is assigned a specific task of the entire system. Hence, each programmer can develop his own flowchart and later on all the flowcharts can be combined for depicting the overall system. Any problems related to linking of different modules can be detected at this stage itself and suitable modifications carried out. Flowcharts can thus be used as working models in design of new software systems.

-Flowcharts provide a strong documentation in the overall documentation of the software system.

-Once the flowchart is complete, it becomes very easy for programmers to write the program from the starting point to the ending point. Since the flowchart is a detailed representation of the program logic no step is missed during the actual program writing resulting in error free programs. Such programs can also be developed faster.

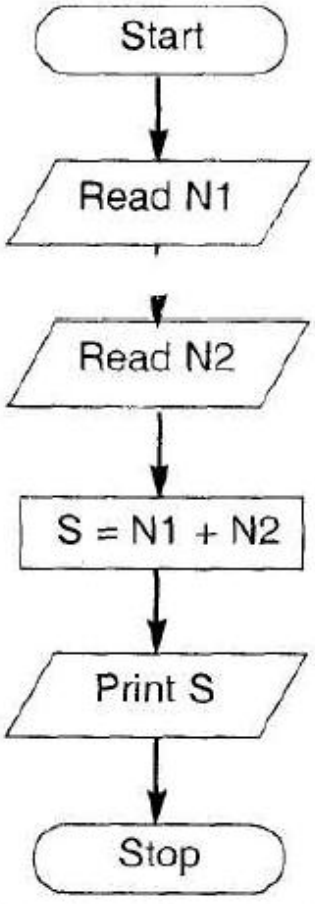
-A flowchart is very helpful in the process of debugging a program. The bugs can be detected and corrected with the help of a flowchart in a systematic manner.

- A flowchart proves to be a very effective tool for testing. Different sets of data are fed as input to program for the purpose.

DEVELOPING FLOWCHARTS

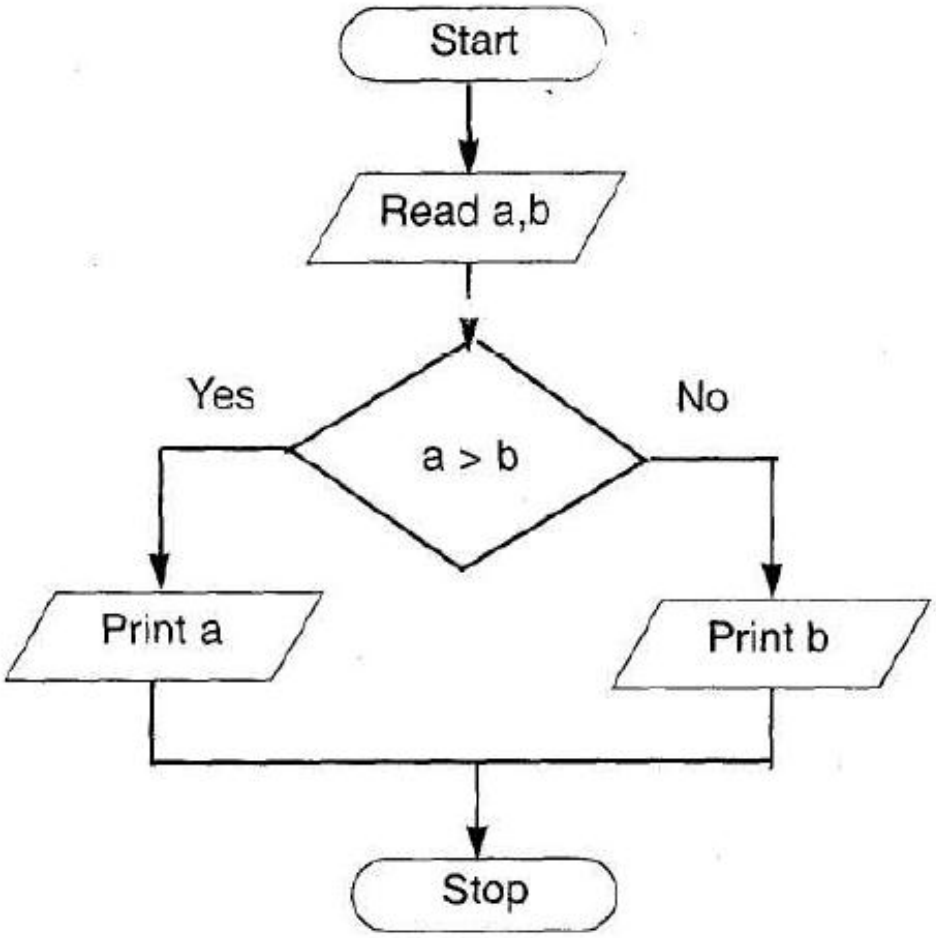
In developing the flowcharts following points have to be considered:
-Defining the problem.
-Identify the various steps required to form a solution.
-Determine the required input and output parameters.
-Get expected input data values and output result.
-Determine the various computations and decisions involved.
With this background of flowcharts and flowchart symbols let us now draw some sample flowcharts. First we shall write the steps to prepare the flowchart for a particular task and then draw the flowchart.

Example: To prepare a flowchart to add two numbers.
The steps are :
1. Start.
2. Get two numbers N1 and N2.
3. Add them.
4. Print the result.
5. Stop.



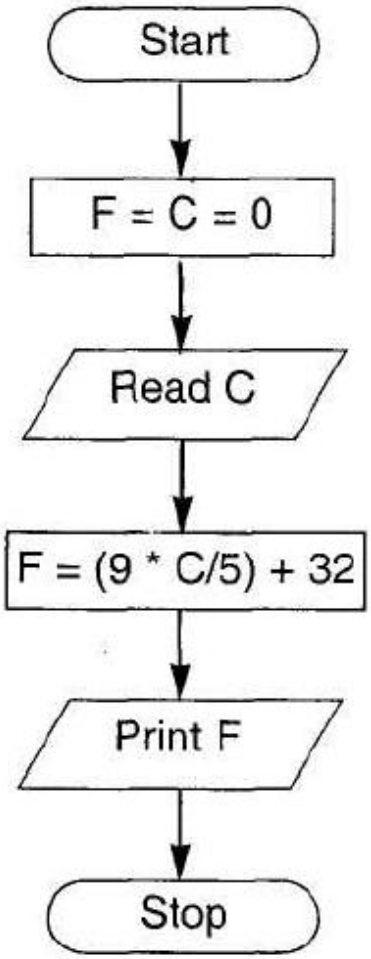
Example: To prepare a flowchart to determine the greatest of two numbers. Here we use the decision symbol. We also combine the two reads for numbers A and
B in one box.

The steps are :
1. Start
2. Get two number A and B.
3. If A > B then print A else print B.
4. Stop.



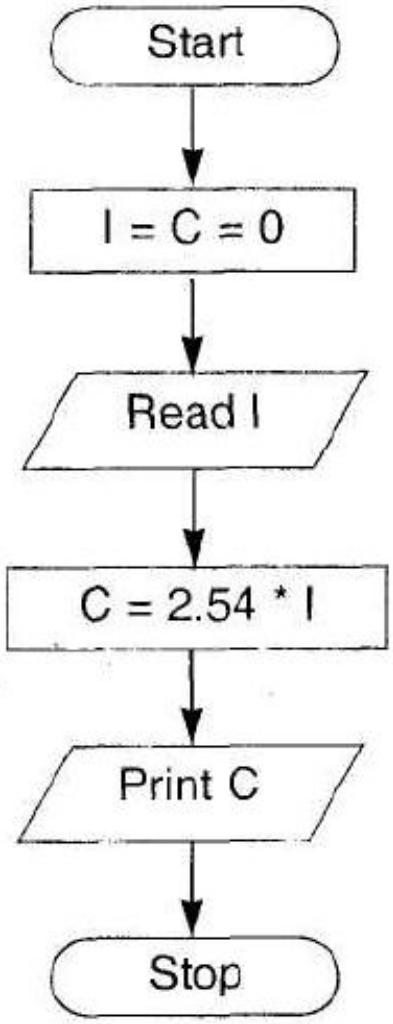
Example: Flowchart for a program that converts temperature in degrees Celsius to degrees Fahrenheit. First let us write the steps involved in this computation technique.

1. Start.
2. Create F and C (for temperature in Fahrenheit and Celsius).
2. Read degrees Celsius into C.
3. Compute the degrees Fahrenheit into F.
4. Print result (F).
5. Stop.



Example: Flowchart for a program that converts inches to centimeters First let us write the steps involved in this computation technique.

1. Start.
2. Create C and I (for Centimeters and Inches respectively).
2. Read value of Inches into I
3. Compute the Centimeters into C.
4. Print result (C).
5. Stop.



Flowcharts for decision making :

Computers are used extensively for performing various types of analysis. The decision symbol is used in flowcharts to indicate it.
The general format of steps for flowcharting is as follows:
- Perform the test of the condition.
- If condition evaluates true branch to Yes steps.
- If condition evaluates false branch to No steps.

Programming Considerations :

Most programming languages have commands for performing test and branching.
The exact commands and syntax depends on the language used. Some of the conditional constructs available in programming languages for implementing decision making in programs are as follows:
- If
- If - else - endif
- If - elseif - endif

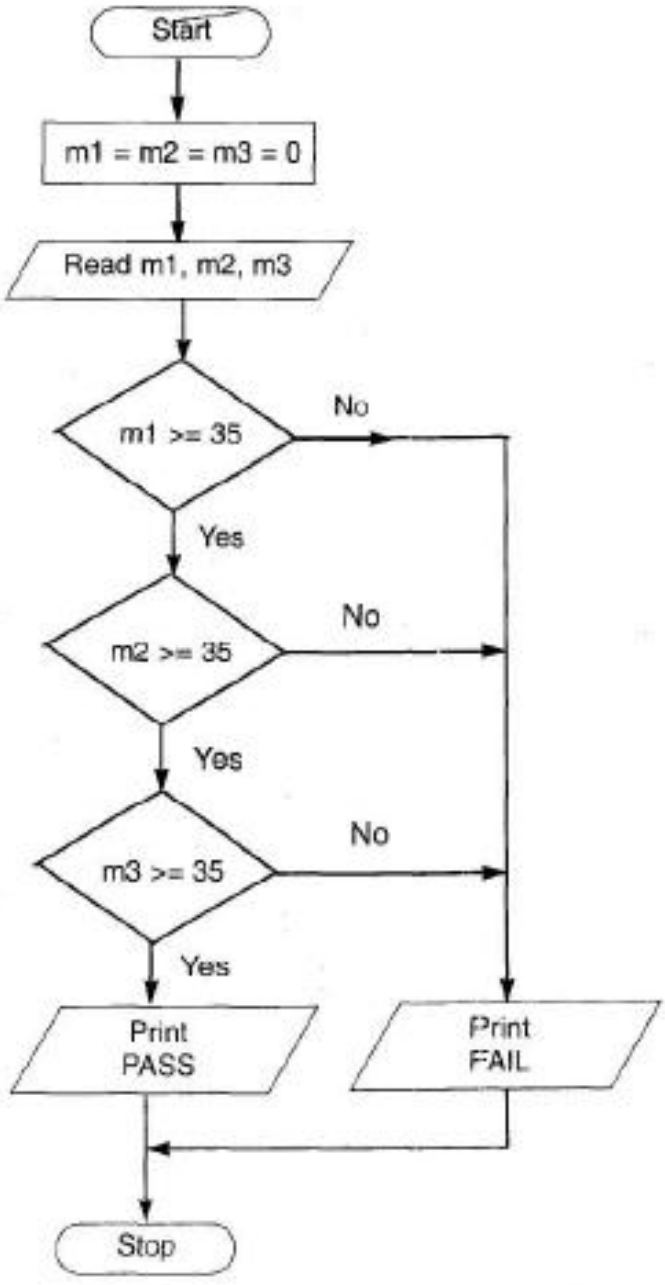
Introduction to Flowcharting / 9

- Do case - endcase.
- Switch.

All languages do not support all of the above constructs.
The operators available for implementing the decision test are as follows:
- Relational Operators (which determine equality or inequality)
- Logical Operators, (useful for combining expressions)

The branching to another set of commands can be implemented by using functions, procedures etc.

Example: Flowchart to get marks for 3 subjects and declare the result. If the marks >= 35 in all the subjects the student passes else fails.
The steps involved in this process are :
1. Start.
2. Create m1, m2, m3.
3. Read marks of three subjects m1, m2, m3.
4. If m1 >= 35 goto step 5 else goto step 7
5. If m2 >= 35 goto step 6 else goto step 7
6. If m3 >= 35 print Pass. Goto step 8
7. Print fail
8. Stop


Start
m1 = m2 = m3 = 0
Read m1, m2, m3
m1 >= 35
No
Yes
m2 >= 35
No
Yes
m3 >= 35
No
Yes
Print
PASS
Print
FAIL
Stop

Flowcharts for loops

Looping refers to the repeated use of one or more steps. i.e. the statement or block of statements within the loop are executed repeatedly. There are two types of loops. One is known as the fixed loop where the operations are repeated a fixed number of times. In this case, the values of the variables within the loop have no effect on the number of times the loop is to be executed. In the other type which is known as the variable loop, the operations are repeated until a specific condition is met. Here, the number of times the loop is repeated can vary.

The loop process in general includes :

- Setting and initialising a counter
- execution of operations
- testing the completion of operations
- incrementing the counter

The test could either be to determine whether the loop has executed the specified number of times, or whether a specified condition has been met.

Programming considerations :

Most of the programming languages have a number of loop constructs for efficiently handling repetitive statements in a program. These include :

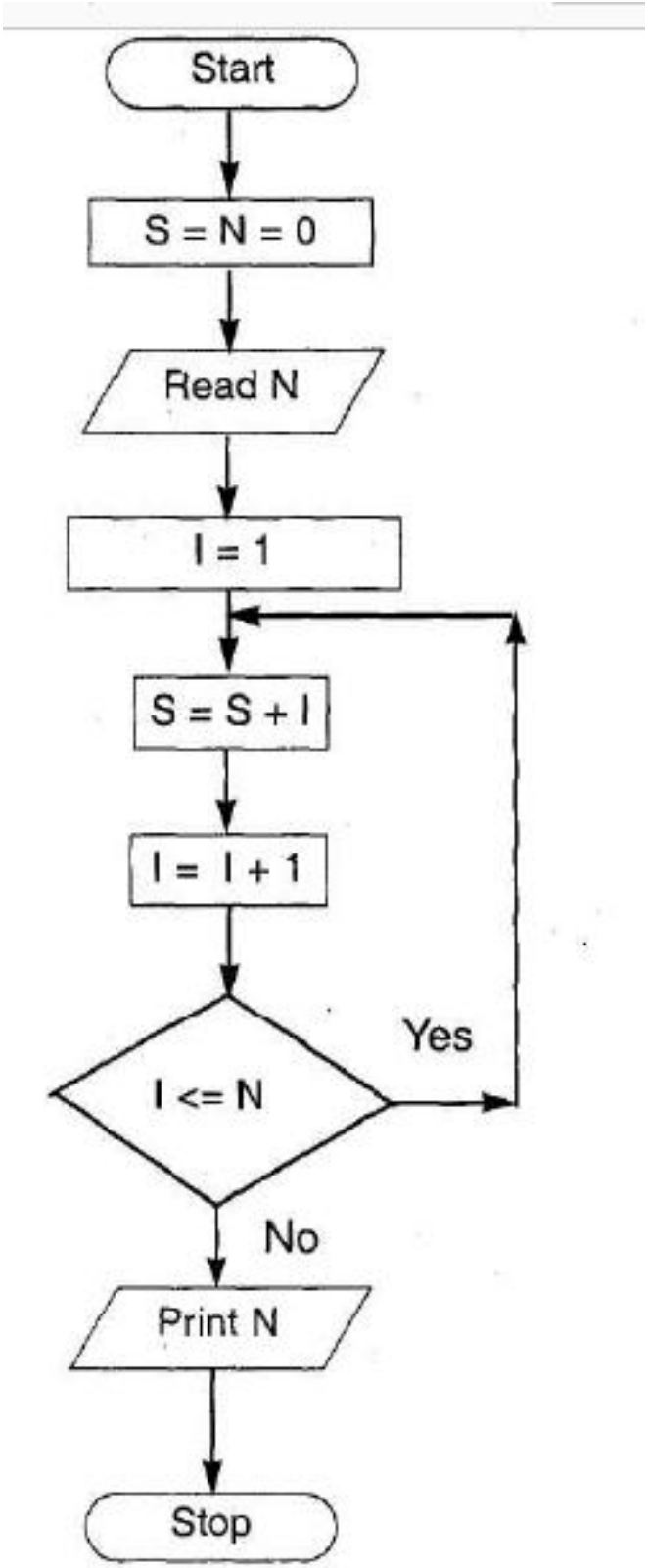
- do-while loop
- while loop
- for loop
- for-next loop

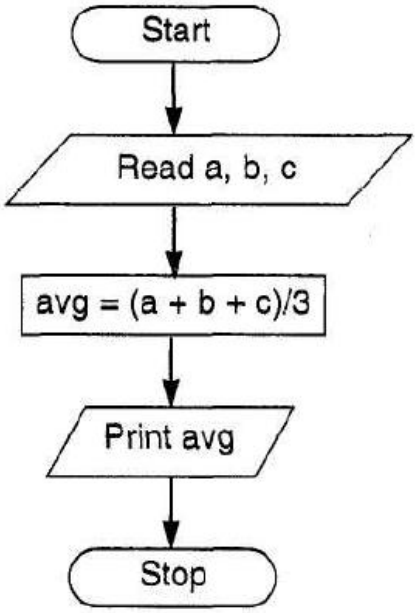
In most of the looping situations, we make use of counters. In situations where the loop is to be repeated on the basis of conditions, relational operators are used to check the conditions.

Example: To find the sum of first N numbers. This example illustrates the use of a loop for a specific number of times.

The steps are:

1. Start
2. Create memvars S , N, I
3. Read N
4. Set S (sum) to 0
5. Set counter (I) to 1.
6. S = S + I
7. Increment I by 1.
8. Check if I is less than or equal to N. If no, go to step 6.

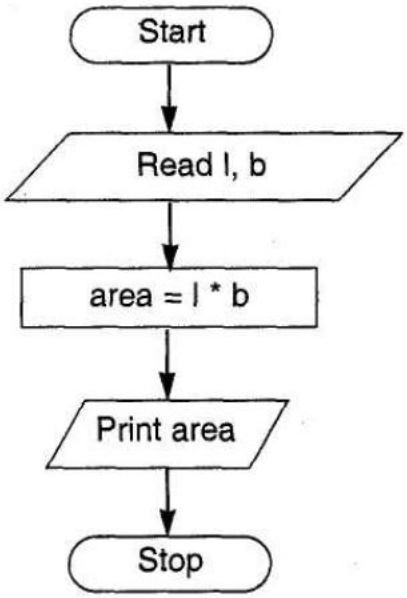

Start
S = N = 0
Read N
I = 1
S = S + I
I = I + 1
I <= N
Yes
No
Print N
Stop



The steps are :

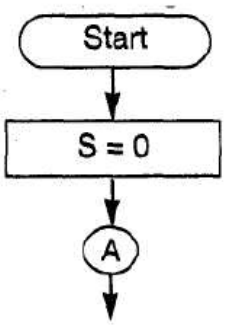
1. Start
2. Read numbers a, b, c
3. Compute the average as (a + b + c)/3
4. Print average
5. Stop.

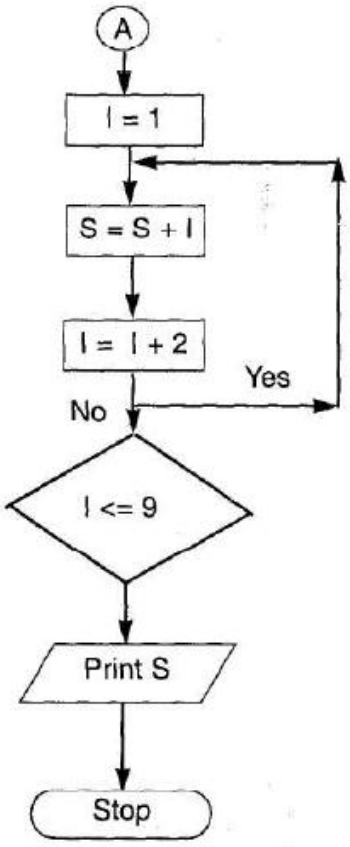
==========

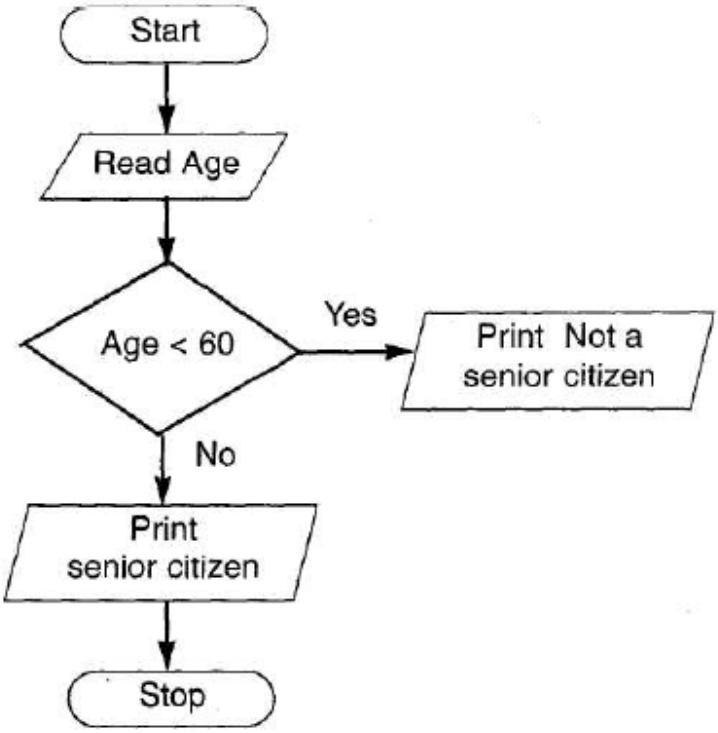


The steps are :

1. Start
2. Read length l and breadth b
3. Compute the area as l * b
4. Print area
5. Stop.


Start
S = 0
A


A
I = 1
S = S + I
I = I + 2
Yes
No
I <= 9
Print S
Stop


Start
Read Age
Age < 60
Yes
Print Not a senior citizen
No
Print senior citizen
Stop