

Unit 1 - Basics of Algorithms and Flow charts

Objectives:

- ❖ Students will come to Know Introduction to Problem solving and algorithms
- ❖ Students can understand about the algorithms
- ❖ Students will cover the: Qualities of good algorithms
- ❖ Students will be able to know about the Advantages and Disadvantages of Algorithm,
- ❖ Properties and efficiency of an algorithm and Tracing of an algorithm
- ❖ Types of Control Structures and Nested Control Structures in Algorithms

Module 1 – Introduction to problem solving and algorithms

1.1 Introduction:

Intelligence is one of the key characteristics which differentiate a human being from other living creatures on the earth. Basic intelligence covers day to day problem solving and making strategies to handle different situations which keep arising in day to day life. One person goes Bank to withdraw money. After knowing the balance in his account, he/she decides to with draw the entire amount from his account but he/she has to leave minimum balance in his account. Here deciding about how much amount he/she may with draw from the account is one of the examples of the basic intelligence. During the process of solving any problem, one tries to find the necessary steps to be taken in a sequence. In this Unit you will develop your understanding about problem solving and approaches.

Problem Solving:

Can you think of a day in your life which goes without problem solving? Answer to this question is of course, No. In our life we are bound to solve problems. In our day to day activity such as purchasing something from a general store and making payments, depositing fee in school, or withdrawing money from bank account. All these activities involve some kind of problem solving. It can be said that whatever activity a human being or machine do for achieving a specified objective comes under problem solving. To make it clearer, let us see some other examples.

Example 1: If you are watching a news channel on your TV and you want to change it to a sports channel, you need to do something i.e. move to that channel by pressing that channel number on your remote. This is a kind of problem solving.

Example 2: One Monday morning, a student is ready to go to school but yet he/she has not picked up those books and copies which are required as per timetable. So here picking up books and copies as per timetable is a kind of problem solving.

Example 3: If someone asks to you, what is time now? So seeing time in your watch and telling him is also a kind of problem solving.

Example 4: Some students in a class plan to go on picnic and decide to share the expenses among them. So calculating total expenses and the amount an individual have to give for picnic is also a kind of problem solving.

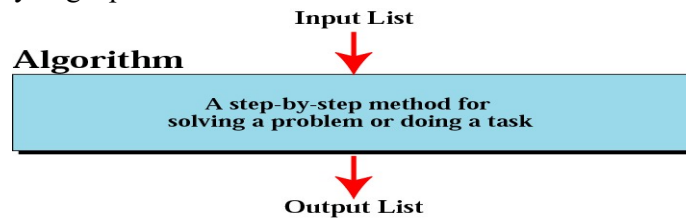
- Now, we can say that problem is a kind of barrier to achieve something and problem solving is a process to get that barrier removed by performing some sequence of activities.
- Here it is necessary to mention that all the problems in the world cannot be solved. There are some problems which have no solution and these problems are called Open Problems.
- If you can solve a given problem then you can also write an algorithm for it. In next section we will learn what is an algorithm?

1.2 Algorithm Introduction

Algorithm is a step-by-step process of solving a well-defined computational problem. In practice, in order to solve any complex real-life problems, first we have to define the problem and then, design algorithm to solve it. Writing and executing a simple program may be easy; however, for executing a bigger one, each part of the program must be well organized. In short, algorithms are used to simplify the program implementation. The study of algorithms is one of the key foundations of computer science. Algorithms are essential to the way computers process information, because a computer program is essentially an algorithm that tells the computer what specific steps to perform (in what specific order) in order to carry out a specified task, such as calculating employees' paychecks or printing students' report cards. Thus, an algorithm can be considered to be any sequence of operations that can be performed by a computing device such as a computer.

1.2.1 What is an Algorithm?

- Algorithm is a step by step procedure to solve a particular problem or “A sequence of activities to be processed for getting desired output from a given input.
- It’s a easy way of analyzing a problem.



1.2.2 Structure of an algorithm:

START

- **STEP 1**
- **SETP 2**
- **SETP 3**
- **SETP 4**
-so on

STOP

Example 1: A simple algorithm to print „Good Morning“.

Step 1: Start

Step 2: Print „Good Morning“

Step 3: Stop

Example 2: Let us take one simple day-to-day example by writing algorithm for making “Maggi Noodles” as a food.

Inputs to the algorithm: Maggi Noodles packet, Pan, Water, Gas

Expected output: Maggi Noodles

Algorithm:

Step 1: Start

Step 2: Take pan with water

Step 3: Put pan on the burner

Step 4: Switch on the gas/burner

Step 5: Put maggi and masala

Step 6: Give two minutes to boil

Step 7: Take off the pan

Step 8: Take out the maggi with the help of fork/spoon

Step 9: Put the maggi on the plate and serve it

Step 10: Stop

Example 3: Find the area of a Circle of radius r.

Inputs to the algorithm: Radius r of the Circle.

Expected output: Area of the Circle

Algorithm:

Step1: Start

Step2: Read\input Radios

Step3: Set 3.14 to PI

Step4: Set Area \leftarrow PI* Radios * Radios // calculation of area

Step5: Print Area

Step6: stop

Example 4: Write an algorithm to read two numbers and find their sum.

Inputs to the algorithm: FirstNumber, SecondNumber

Expected output: Sum of the two numbers.

Algorithm:

Step1: Start
Step2: Read\input num1.
Step3: Read\input num2.
Step4: Set Sum \leftarrow num1+num2 // calculation of sum
Step5: Print Sum
Step6: End

1.3 Expressing Algorithms:

An algorithm is a set of steps designed to solve a problem or accomplish a task. Algorithms are usually written in pseudocode, or a combination of your speaking language and one or more programming languages, in advance of writing a program. An algorithm may be expressed in a number of ways, including:

- **Natural language:** usually verbose and ambiguous.
- **Flow charts:** avoid most (if not all) issues of ambiguity; difficult to modify w/o specialized tools; largely standardized.
- **Programming language:** Tend to require expressing low – level details that are not necessary for a high – level understanding.

1.4 Qualities of a good algorithm

1. Inputs and outputs should be defined precisely.
2. A good algorithm should produce correct and accurate results for any set of legal or correct inputs.
3. Each step in algorithm should be clear and unambiguous.
4. Algorithm should be most effective among many different ways to solve a problem.
5. An algorithm shouldn't have computer code. Instead, the algorithm should be written in such a way that it can be used in similar programming languages Whenever we write an algorithm, we should make sure that all of these parameters given for a good algorithm are incorporated.

1.5 Advantage and disadvantages of algorithm

1.5.1 Advantages

1. It is a step-wise representation of a solution to a given problem, which makes it easy to understand.
2. An algorithm uses a definite procedure.
3. It is not dependent on any programming language, so it is easy to understand for anyone even without programming knowledge.
4. Every step in an algorithm has its own logical sequence so it is easy to debug.
5. By using algorithm, the problem is broken down into smaller pieces or steps hence, it is easier for programmer to convert it into an actual program

1.5.2 Disadvantages

1. Writing algorithm takes a long time.
2. An Algorithm is not a computer program, it is rather a concept of how a program should be
3. Big tasks are difficult to put in Algorithms.
4. Difficult to show Branching and Looping in Algorithms.

1.6 Properties of algorithm:

1. **Finiteness:** An algorithm must always terminate after a finite number of steps. It means after every step one reach closer to solution of the problem and after a finite number of steps algorithm reaches to an end point.
2. **Definiteness:** Each step of an algorithm must be precisely defined. It is done by well thought actions to be performed at each step of the algorithm. Also the actions are defined unambiguously for each activity in the algorithm.
3. **Input:** An algorithm has zero or more inputs, taken from a specified set of objects.
4. **Output:** An algorithm has one or more outputs, which have a specified relation to the inputs.
5. **Effectiveness:** All operations to be performed must be sufficiently basic that they can be done exactly and in finite length

1.7 Algorithm Efficiency

Algorithmic efficiency is a property of an algorithm which relates to the number of computational resources used by the algorithm. An algorithm must be analyzed to determine its resource usage, and the efficiency of an algorithm can be measured based on usage of different resources.

- ❖ Speed - Method that takes the least time
- ❖ Space - Method that uses the least memory
- ❖ Code — Method that is shortest to describe

Speed is now the most important factor Example

A real world example: Travelling Vempalli to Hyderabad in Vehicle

Case 1:

1. Take vehicle
2. Check vehicle condition if not get it repair
3. Drive from vempalli to Proddutur
4. Then Proddutur to Hyderabad
5. Travel is successfully completed.

Case 2:

1. Take vehicle
2. Check vehicle condition if not get it repair
3. Drive from Vempalli to kadapa
4. Then Kadapa to kurnool
5. Then Kurnool to Hyderabad
6. Travel is successfully completed

While comparing both cases Algorithm executed successfully with same output but in case 1 travelling time is lesser than the case 2 travelling time.

1.8 Tracing an Algorithm

Tracing of an Algorithm means showing how the value of a variable changes during the execution of an algorithm.

Steps in tracing:

1. Identify the variables in the algorithm that need to be traced.
2. Examine the value of the variables at each step in the execution of the algorithm.
3. Determine if the algorithm is giving the correct outputs for a set of legitimate/legal input. Values.
4. Analyze and determine what the purpose of the algorithm

In this module we will see examples of computational algorithms and the techniques used for tracing algorithms. A computational algorithm is a set of precise instructions expected to be expressed as a computer program that can execute on a computer.

Let us now get started with an example of adding two numbers.

Algorithm: Add two numbers

- Step1: Start
- Step2: Input x
- Step3: Input y
- Step4: Set $z \leftarrow x + y$
- Step5: Output z
- Step6: Stop

If you notice the algorithm it takes two inputs, performs an addition operation and gives the output. Let us see how the algorithm executes step by step.

Input x /* Takes input from the user by using input devices like keyboard. After entering input from the key board, for ex: 5 the value is stored in a memory location. The name of this memory location is x in this particular case. Any time this variable x can store only one value. For ex: if you set x value 6 you will find the new value is referred by x is 6 but not 5 that means the old value which was stored in x values is lost. This is a very good example that shows a variable can store only one value at a

time. Input value :5 */

Input y /* Similar to previous statement where new memory location is created for variable y and the user inputs to y is stored. Input value :2 */

Set z to x+ y /* in this statement actual operation that is performed on the input values which is to add the numbers referred to by the variables x and y.

In this example you will notice after the addition operation is executed and the values 5 and 2 are added and set to new variable z. */

Output z //the result value i.e. z is displayed to an output device like monitor?

You will notice that algorithm also can be visualized as functions. A function performs a specific operation like addition and multiplication or it could be more complex operations like finding square roots or sorting list of numbers.

Variable:

- ❖ Variables are used to handle the data by the algorithm.
- ❖ Variables refer to a memory location where the actual value is stored.
- ❖ Variable can store one value at a time.
- ❖ Old values of the variable are lost when a new value is assigned.

1.9 Control structure:

Control Structures are just a way to specify flow of control in programs. Any algorithm or program can be clearer and more understood if they use self-contained modules called as logic or control structures. It basically analyzes and chooses in which direction a program flows based on certain parameters or conditions.

19.1 Types of control structures:

There are three types of control structures

1. Sequence
2. Selection or Branching
3. Loop or Repetition

1.9.1 Sequence Control Structure: This refers to the line – by – line execution, in which statements are executed sequentially, in the same order in which they appear in the script. They might, for example, carry out a series of read or write operations, arithmetic operations, or assignments to variables.

Example problem:

Write algorithm to find the greater number between two numbers

Step1: Start

Step2: Read\input the Number1

Step3: Read\input the Number2.

Step4: Set Average \leftarrow (Number1+ Number2)/2

Step5: Print Average

Step6: End

1.9.2 Selection or Decision Control Structure: The branch refers to a binary decision based on some condition. Depending on whether a condition is true or false. If the condition is true, one of the two branches is explored; if the condition is false, the other alternative is taken. This is usually represented by the ‘if-then’ construct in pseudo-codes and programs. This structure is also known as the selection structure

Example problem:

A person whose age is more than or equals to 18 years is eligible to vote. Now write an algorithm to check whether he is eligible to vote?

Step 1: Start

Step 2: Input/Read age //Taking age value from the user

Step 3: Check if age \geq 18

a) Print “Eligible to vote” //If Condition is true

Step 4: Otherwise/else

a) Print “Not eligible to vote” //if Condition is false

Step 5: Stop

1.9.3 Repetition or Loop Control Structure: This is a control structure that allows the execution of a block of statements multiple times until a specified condition is met. The loop allows a statement or a sequence of statements to

be repeatedly executed based on some loop condition. It is represented by the 'while' and 'for' constructs in most programming languages, for unbounded loops and bounded loops respectively. (Unbounded loops refer to those whose number of iterations depends on the eventuality that the termination condition is satisfied; bounded loops refer to those whose number of iterations is known before-hand.) In the flowcharts, a back arrow hints the presence of a loop. A trip around the loop is known as iteration. You must ensure that the condition for the termination of the looping must be satisfied after some finite number of iterations, otherwise it ends up as an infinite loop, a common mistake made by inexperienced programmers. The loop is also known as the repetition structure.

Example problem:

Algorithm to print all-natural numbers upto "n" (Here you need to accept a number from the user, and to print all the natural numbers till 'n' i.e. 1 to n)

Step 1: Start

Step 2: Read/Input n

Step 3: Store 1 to "i" //initialization

Step 4: Do the following statements until $i \leq n$ //condition

a) print i

b) add 1 to i //increment

Step 5: Stop

Nested control structures: Combining the use of these control structures, for example, a loop within a loop (nested loops), a branch within another branch (nested if), a branch within a loop, a loop within a branch, and so forth, is not uncommon. Complex algorithms may have more complicated logic structure and deep level of nesting, in which case it is best to demarcate parts of the algorithm as separate smaller modules. Beginners must train themselves to be proficient in using and combining control structures appropriately, and go through the trouble of tracing through the algorithm before they convert it into code

Module 2- Introduction to Flow chart

Objectives:

- ❖ Students will come to Know Introduction to flowchart
- ❖ Students can understand about the symbols and its use
- ❖ Students will be able to know about the Advantages and Disadvantages of Flowchart
- ❖ Types of Control Structures and Nested Control Structures in Flowcharts
- ❖ Difference between algorithm and flowchart & Pseudo code.

1.10 Flowchart

The diagrammatic representation of an algorithm is called “Flowchart”. 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.

For example suppose you are going for a picnic with your friends then you plan for the activities you will do there. If you have a plan of activities then you know clearly when you will do what activity. Similarly when you have a problem to solve using computer or in other word you need to write a computer program for a problem then it will be good to draw a flowchart prior to writing a computer program. Flowchart is drawn according to defined rules.

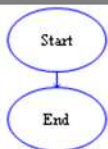
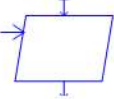
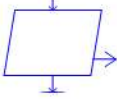
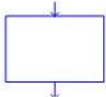
Features of flowchart:

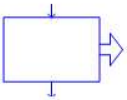
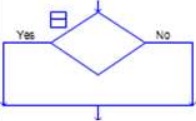
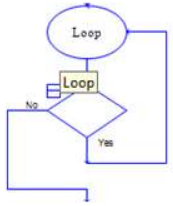
The features of a flowchart are:

1. It is an easy method of communication.
2. It is independent of a programming language.
3. It is the key to correct programming.
4. It clearly indicates the task to be performed at each level.

1.11 Flowchart Symbols

There are 6 basic symbols commonly used in flowcharting of assembly language Programs: Terminal, Process, input/output, Decision, Connector and Predefined Process. This is not a complete list of all the possible flowcharting symbols, it is the ones used most often in the structure of Assembly language programming.

Purpose	Symbol	Name	Description
Start/Stop		Start & Stop	It is Oval symbol
INPUT		Input statement	Allow the user to enter data. Each data value is stored in a variable .
OUTPUT		Output statement	Display (or save to a file) the value of a variable .
PROCESSING		Assignment statement	An assignment statement is used to modify or replace the data stored at the memory location associated with a variable. .

Purpose	Symbol	Name	Description
PROCESSING (Function)		Procedure call	Execute a group of instructions defined in the named procedure. In some cases some of the procedure arguments (i.e., variables) will be changed by the procedure's instructions.
Selection		Decision making statement	Allows you to make 'decision' in boolean type
Loop		Iteration statements	Which allows you to execute repeatedly until certain condition satisfied. (An iteration control statement controls how many times a block of code is executed)

1. 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 from one memory location to another is also represented in the process box. If there are more than one process

2. 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.

3. 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 are 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.

4. **Loops:** Looping is a problem-solving technique through which a group of statements is executed repeatedly, until certain specified condition is satisfied. Looping is also called a repetitive or an iterative control statement

5. Connectors:

This symbol shows continuation of the flow chart from one page to another. When you reach the bottom of the page or need to jump to another page, draw flow chart connector symbol and connect it to the last item on the chart. Label the inside of the symbol with a letter, typically beginning with an "A".

6. Predefined process (Function):

This symbol indicates a sequence of actions that perform a specific task embedded within a larger process.

7. 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.

1.12 General Rules for flowcharting

1. All boxes of the flowchart are connected with Arrows. (Not lines)
2. Flowchart symbols have an entry point on the top of the symbol with no other entry points. The exit point for all flowchart symbols is on the bottom except for the Decision symbol.
3. The Decision symbol has two exit points; these can be on the sides or the bottom and one side.
4. Generally, a flowchart will flow from top to bottom. However, an upward flow can be shown as long as it does not exceed 3 symbols.
5. Connectors are used to connect breaks in the flowchart. Examples are:
 - i. From one page to another page.
 - ii. From the bottom of the page to the top of the same page.
6. Subroutines (Functions) and Interrupt programs have their own and independent flowcharts.
7. All flow charts start with a Terminal or Predefined Process (for interrupt programs or subroutines) symbol.
8. All flowcharts end with a terminal or a contentious loop.

Flowcharting uses symbols that have been in use for a number of years to represent the type of operations and/or processes being performed. The standardized format provides a common method for people to visualize problems together in the same manner. The use of standardized symbols makes the flow charts easier to interpret, however, standardizing symbols is not as important as the sequence of activities that make up the process.

1.13 Advantages and Disadvantages of using flowchart

1.13.1 Advantages: As we discussed flow chart is used for representing algorithm in pictorial form. This pictorial representation of a solution/system is having many advantages. These advantages are as follows:

- **Communication:** A Flowchart can be used as a better way of communication of the logic of a system and steps involve in the solution, to all concerned particularly to the client of system.
- **Effective analysis:** A flowchart of a problem can be used for effective analysis of the problem.
- **Documentation of Program:** Program flowcharts are a vital part of good program documentation. Program document is used for various purposes like knowing the components in the program, complexity of the program etc.
- **Efficient Program Maintenance:** Once a program is developed and becomes operational it needs time to time maintenance. With help of flowchart maintenance become easier.
- **Coding of the Program:** Any design of solution of a problem is finally converted into computer program. Writing code referring the flowchart of the solution become easy.

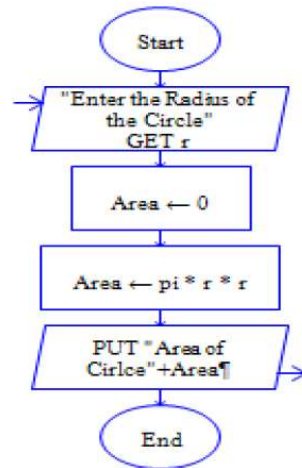
1.13.2 Disadvantages:

- Drawing flowchart is a time-consuming task.
- Manual tracing is needed to check correctness of flowchart drawn on paper.
- Simple modification in problem logic may lead to complete redraw of flowchart.
- Showing many branches and looping in **flowchart** is difficult.
- In case of complex program/algorithm, **flowchart** becomes **very complex** and **clumpy**.

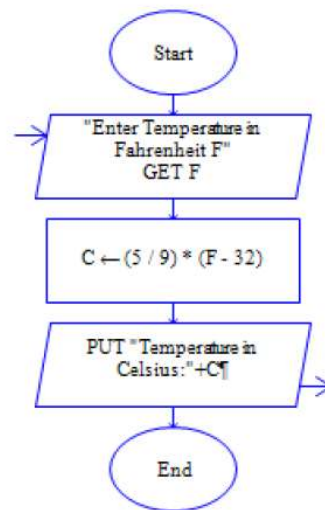
1.14 Some examples of Flowcharts

Now, we will discuss some examples on flowcharting. These examples will help in proper understanding of flowcharting technique. This will help you in program development process in next unit of this block.

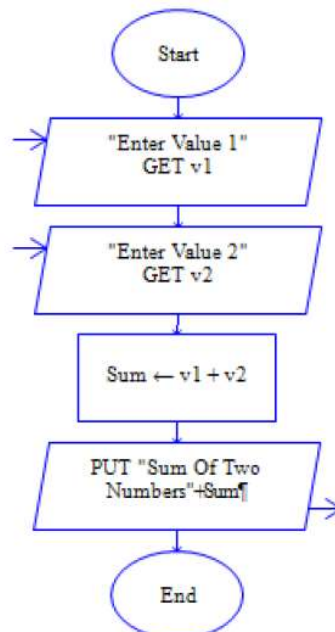
Problem1: Find the area of a circle of radius r



Problem 2: Convert temperature Fahrenheit to Celsius.



Problem3: Flowchart for an algorithm which gets two numbers and prints sum of their value.



1.15 Types of control structures:

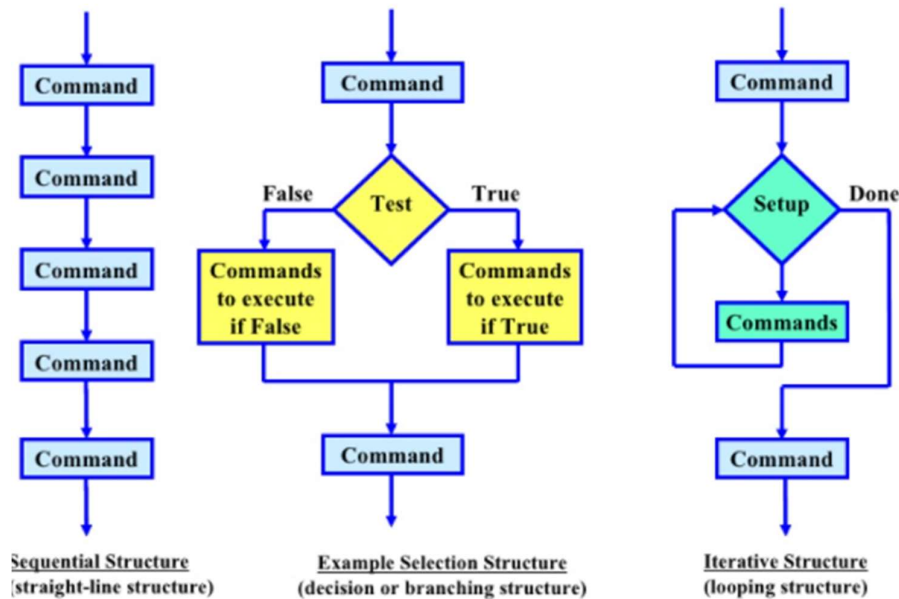
Definition

- The logic of a program may not always be a linear sequence of statements to be executed in that order.
- The logic of the program may require execution of a statement based on a decision.
- Control structures specify the statements to be executed and the order of execution of statements.

Where is the usage of control structure?

Flowchart and Pseudo Code use Control structures for representation

Flowcharts for sequential, selection, and iterative control structures



There are three kind of Control Structure:

1. Sequential
2. Selection(Branch or conditional)
3. Iterative(loop)

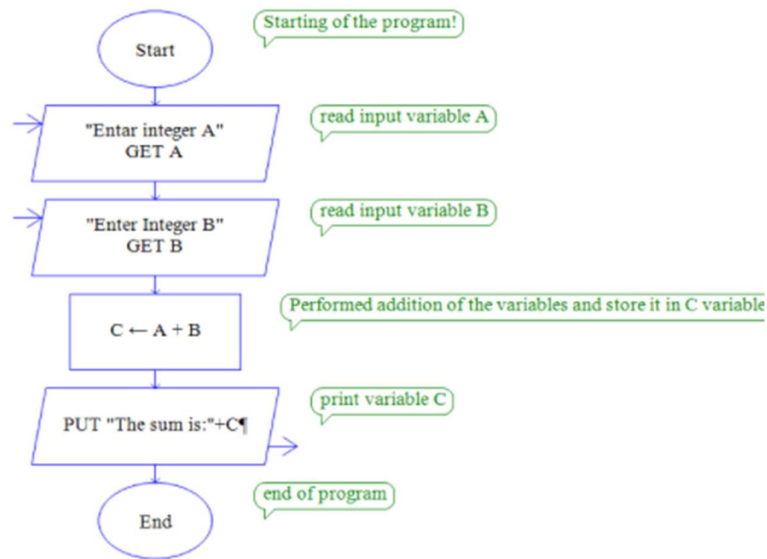
1.15.1 Sequential:

- Instruction is executed in linear order.
- Statements are executed in a specified order. No statement is skipped and no statement is executed more than once.

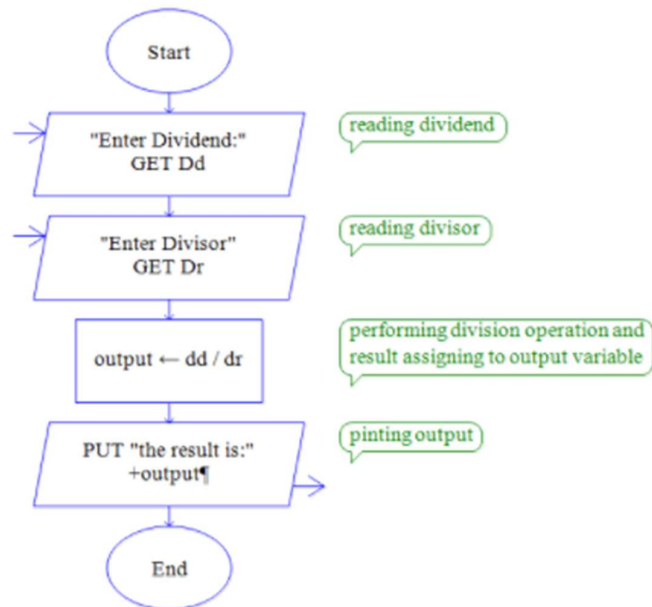


Example problem:

1. Take two numbers from the user print their sum?



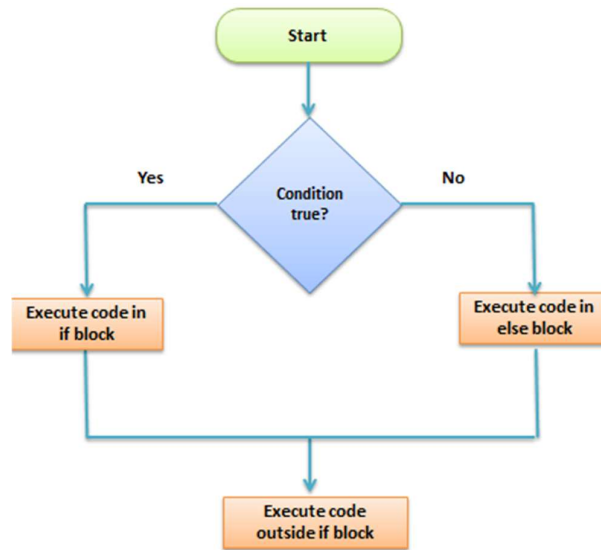
2. Take input of two numbers from the user and do the division of two numbers?



1.15.2 Selection (Branch or conditional):

- It's ask a true/false question THEN select the next instruction based on the answer.
- It selects a statement to execute on the basis of **condition**. Statement is executed when the condition is true and ignored when it is false

Example: if, if else, switch structures.



Example problem:

3. Draw a flowchart to determine greatest among two numbers?

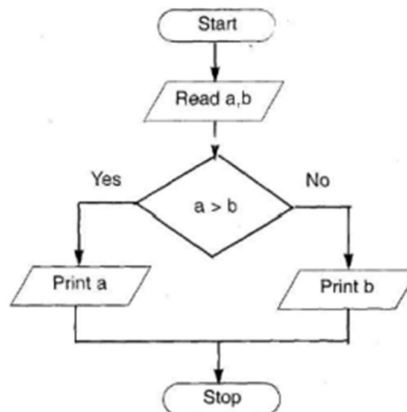


Fig. 2b) Flowchart to determine the greater of two numbers a and b

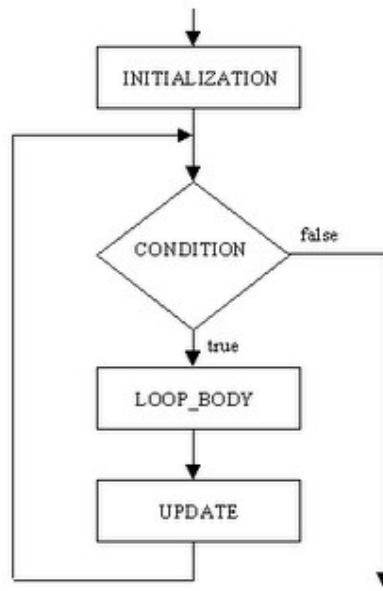
1.15.3 Iterative (loop):

- It's repeat the execution of a block of instruction.
- In this structure the statements are executed more than one time. It is also known as iteration or loop
- A loop structure is used to execute a certain set of actions for a predefined number of times or until a particular condition is satisfied

Example: while loop, for loop do-while loops etc.



Iterative Structure
(looping structure)

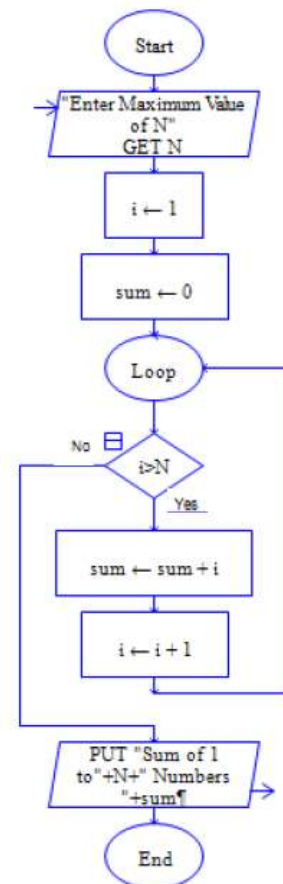


Iterative Structure

Example problem:

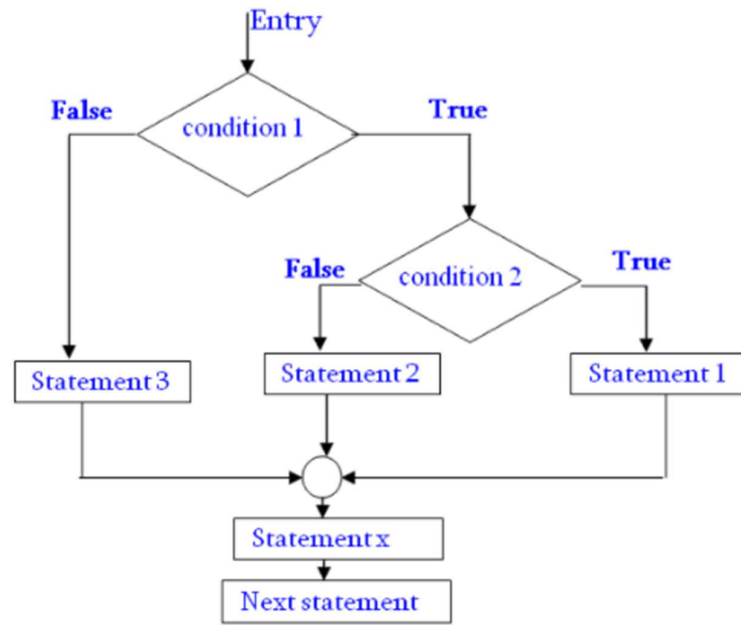
Write an algorithm and flowchart to find Total Sum from 1 to 100 natural numbers

- Step 1: Start
- Step 2: Read value of n
- Step 3: Set $i \leftarrow 1$
- Step 4: Set $Total_Sum \leftarrow 0$
- Step 5: Repeat the steps until i greater than n
 - a) Set $Total_Sum \leftarrow Total_Sum + i$
 - b) increment i // (i.e. $i = i + 1$)
- Step 6: Display "1 to 100 Natural Numbers Sum:" $Total_Sum$
- Step 7: End



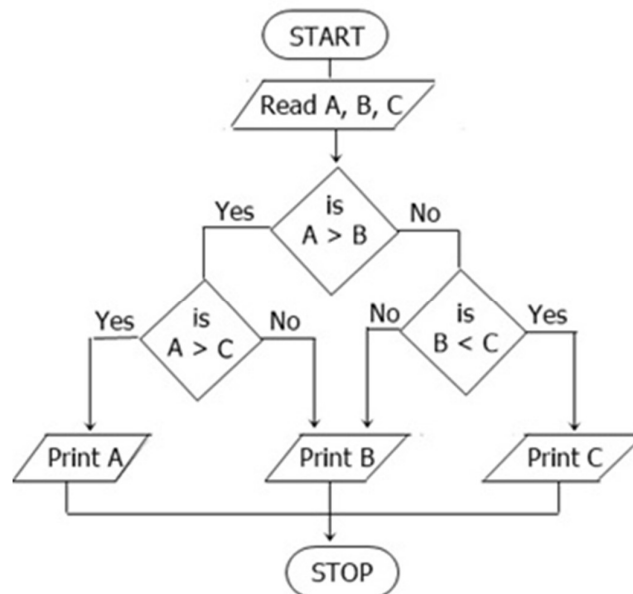
Nested Control Structures:

A nested control statement is a control statement that is contained within another control statement. You can do this to many levels. You can place control statements inside other control statements, for example an If...Then...Else block within a For...Next loop. A control statement placed inside another control statement is said to be *nested*.



Example problem:

1. Draw a flowchart to determine largest among three numbers?



1.16 Pseudocode

A pseudocode is an informal way of writing a program. However, it is not a computer program. It only represents the algorithm of the program in natural language and mathematical notations. Besides, there is no particular programming language to write a pseudocode. Unlike in regular programming languages, there is no syntax to follow when writing a pseudocode. Furthermore, it is possible to use pseudo codes using simple English language statements.

- Pseudocode is a kind of structured English for describing algorithm.
- It has no syntax like any of the programming language and thus can't be compiled or interpreted by the computer.
- No syntax for pseudocode

- Not executable program

1.17 Advantages of Pseudocode

- Improves the readability of any approach. It's one of the best approaches to start implementation of an algorithm.
- Acts as a bridge between the program and the algorithm or flowchart. Also works as a rough documentation, so the program of one developer can be understood easily when a pseudo code is written out. In industries, the approach of documentation is essential. And that's where a pseudo-code proves vital.
- The main goal of a pseudo code is to explain what exactly each line of a program should do, hence making the code construction phase easier for the programmer

1.18 How to write a Pseudo-code?

- To begin the comment double forward slash are used “//”.
 - Matching braces “{ and }” are used to present blocks where a compound statement (set of simple statements) can be illustrated as a block and terminated by a semicolon”;“. The body of a procedure constructs a block as well.
 - All the identifiers start with a letter and the datatype of the variables are not declared explicitly.
 - An assignment statement is used for the assigning values to the variables.
 - To produce the boolean values (i.e., true and false) the logical operators and, or and not and the relational operators <, ≤, =, ≥ and > are provided.
- Input and output are presented by read and write instructions.

Example:

1. A pseudocode to find the total of two numbers is as follows.

```
Sum Of Two Numbers()
begin
Set sum =0;
Read: number 1, number 2;
Set sum = number1 + number 2;
Print sum;
End
```

2. A pseudocode to find the area of a triangle is as follows.

```
AreaofTrinagle()
Begin
Read: base,height;
Set area =0.5*base*height;
Print area;
End
```

1.19 Differences between algorithm and Flowchart:

Algorithm	Flow chart
Step by step instruction representing the process of any solution.	Block by block information diagram representing the data flow.
It is step wise analysis of the work to be done	It is a pictorial representation of a process
Solution is shown in non computer language like English.	Solution is shown in graphical format.
It is something difficult to understand.	Easy to understand as compared to algorithm.
Difficult to show branching and looping.	Easy to show branching and looping.
Algorithm can be written for any problem.	Flow chart for big problem is impractical.
Easy to debug errors.	Difficult to debug errors.

It is difficult to write algorithm as compared to flowchart.

It is easy to make flowchart.

1.20 Differences between algorithm and Pseudocode:

Basis for comparison	Algorithm	Pseudocode
Comprehensibility	Quite hard to understand	Easy to interpret
Uses	Complicated programming language	Combination of programming language and natural language.
Debugging	moderate	simpler
Ease of construction	Complex	Easier

1.21 Difference between flow chart and pseudo code:

Flow chart	Pseudo code
A diagrammatic representation that illustrates a solution model to a given problem.	An informal high – level description of the operating principle of an algorithm.
Written using various symbols.	Written in natural language and mathematical notations help to write pseudo code.

Solved problems:

1. Write an algorithm to go for class picnic.

Algorithm:

- Step 1: Start
- Step 2: Decide the picnic venue, date and time
- Step 3: Decide the picnic activities
- Step 4: Hire a vehicle to reach to the venue and comeback
- Step 5: Goto to the picnic venue on the decided date
- Step 6: Do the activities planned for the picnic
- Step 7: Come back to school in the hired vehicle
- Step 8: Stop

2. To celebrate Teachers' Day

Algorithm:

- Step 1: Start
- Step 2: Decide the activities for teachers' day like dance performances, plays, etc.
- Step 3: Form groups of students and assign the decided activities from step 2 to each group.
- Step 4: Decide the practice timings for each group.
- Step 5: Each group to practice as per the timings decided in step 4.
- Step 6: Invite the teachers to Teachers' Day celebrations.
- Step 7: Perform the activities planned in step 2 on Teachers' Day
- Step 8: Stop

3. To celebrate New Year

Algorithm:

- Step 1: Start
- Step 2: Prepare a guest list for New Year party
- Step 3: Decide the venue, food menu, games and fun activities for the party
- Step 4: Invite the guests for the party
- Step 5: On New Year eve, get ready and enjoy the party
- Step 6: Stop

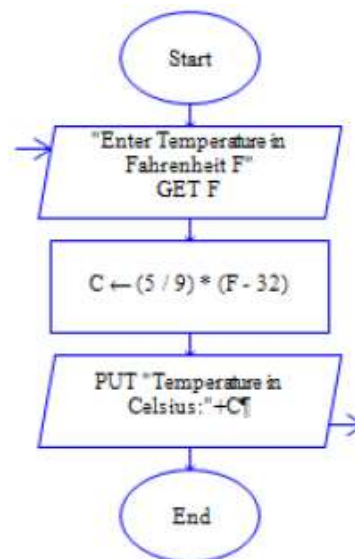
4. Convert temperature Fahrenheit to Celsius with flowchart

Inputs to the algorithm: Temperature in Fahrenheit

Expected output: Temperature in Celsius

Algorithm:

- Step1: Start
- Step2: Read Temperature in Fahrenheit F
- Step3: Set $C \leftarrow 5/9*(F-32)$
- Step4: Print "Temperature in Celsius: "+C
- Step5: End



5. Write an algorithm to read two numbers and find their product?

Inputs to the algorithm: First num1. Second num2.

Expected output: Sum of the two numbers.

Algorithm:

Step1: Start

Step2: Read\input num1.

Step3: Read\input num2.

Step4: set product \leftarrow num1 * num2 // calculation of product

Step5: Print product.

Step6: End

6. An algorithm to display even numbers between 0 and 99 with flowchart

Step 1: Start

Step2: input/read value of n

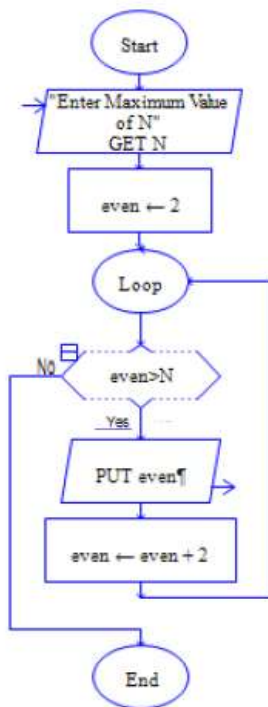
Step3: Set even \leftarrow 0

Step4: Do the following until even < 100

a) Display even

b) Set even \leftarrow even+2

Step5: End



7. Design an algorithm which generates even numbers between 1000 and 2000 and then prints them in the standard output. It should also print total sum:

Step 1: Start

Step2: input/read value of n

Step3: Set even \square 1000

Step4: Do the following until even < 2000

a) Display even

b) Set even $\square \square$ even+2

Step5: End

8. Define the following symbol and use?



Answer: Process Box: A process box is used to represent all types of mathematical tasks like addition, subtraction, multiplication, division, etc.

9. Algorithm to find area of a rectangle?

Algorithm:

Step 1: Start

Step 2: Take length and breadth and store them as L and B? // Input

Step 4: Multiply L and B and store it in area //Area of Rectangle = L*B

Step 5: Print area //Output

Step 6: Stop.

10. Write an algorithm to find the largest among three different numbers entered by user with flowchart

Step 1: Start

Step 2: Read variables a, b and c.

Step 3: If a greater than b

a) If a greater than c

i) Display a is the largest number.

b) else

i) Display c is the largest number.

Step 4: else

a) If b greater than c

i) Display b is the largest number.

b) else

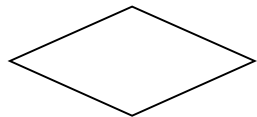
i) Display c is the greatest number.

Step 5: Stop

Multiple Choice Questions:

- 1) An Algorithm is expressed by
 - a) flowchart
 - b) common language
 - c) pseudo code
 - d) all
- 2) A good algorithm having finite steps?
 - a) True
 - b) False
- 3) Algorithm efficiency is measured by ?
 - a) speed
 - b) space
 - c) code
 - d) all of the above
- 4) How many control structures are there in algorithm?
 - a) 3
 - b) 2

- c) 1
- 5) Which control structure used weather a condition is true or false?
- a) **selection**
 - b) iteration
 - c) sequence
 - d) none
- 6) Which control structure used to check a condition more than one time?
- a) iteration
 - b) loop
 - c) **a & b**
 - d) None of the above
- 7) In computer science, algorithm refers to a pictorial representation of a flowchart.
- a) True
 - b) **False**
- 8) The process of drawing a flowchart for an algorithm is called _____
- a) Performance
 - b) Evaluation
 - c) Algorithmic Representation
 - d) **Flowcharting**
- 9) Actual instructions in flowcharting are represented in _____
- a) Circles
 - b) **Boxes**
 - c) Arrows
 - d) Lines
- 10) A box that can represent two different conditions.
- a) Rectangle
 - b) **Diamond**
 - c) Circle
 - d) Parallelogram
- 11) The following box denotes?



- a) **Decision**
 - b) Initiation
 - c) Initialization
 - d) I/O
- 12) There should be certain set standards on the amount of details that should be provided in a flowchart
- a) True
 - b) **False**
- Answer: b
- Explanation: The statement is false. There should be no set standards on the amount of details that should be provided in a flowchart.
- 13) Which of the following is not an advantage of a flowchart?
- a) Better communication
 - b) Efficient coding
 - c) Systematic testing
 - d) **Improper documentation**
- Answer: d
- Explanation: Flowcharts provide a proper documentation. It also provides systematic debugging.
- 14) The symbol denotes _____



- a) I/O
- b) Flow
- c) **Terminal**
- d) Decision

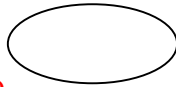
15) Pseudocode is an informal high-level description of an **algorithm**.

- a) **True**
- b) false

16) In a flowchart a calculation(process) is represented by

- a) **A rectangle**
- b) A rhombus
- c) A parallelogram
- d) A circle

17) The symbol denotes _____



- a) **I/O**
- b) Flow
- c) Loop
- d) Decision

18) To repeat a task a number times

- a) **Loop statement**
- b) Input statement
- c) Conditional statement
- d) Output statement

19) In a flow chart how are the symbols connected?

- a) Symbols do not get connected together in a flowchart
- b) **With lines and arrow to show the direction of flow**
- c) With dashed lines and numbers
- d) With solid lines to link events

20) A flow chart does need to have a start?

- a) **True**
- b) False

Unsolved Problems:

- 1) Write an algorithm and flow chart to add four numbers?
- 2) Write an algorithm to make tea/coffee
- 3) Write 6 No's Algorithms for performing day to day activities.
- 4) Write an algorithm and flow chart to find large number among given two numbers.
- 5) Draw the flow chart to convert distance entered in Km to Meters.
- 6) Accept the age of a person and check whether he/she is eligible to vote or not. A person is eligible to vote only when he/she is 18 years or more.
- 7) Draw the flow chart to find the area of a rectangle whose length and breadth are given
- 8) Draw the flow chart to find the average of three numbers a, b and c.
- 9) Write an algorithm and flow chart Print 1 to 100?
- 10) Write an algorithm to find average of three numbers?