Unit 1 Concept of Algorithm, Flowchart and Languages

Algorithm

The term algorithm refers to the logic of a program. It is a step-by-step description of how to arrive at a solution to a given problem. It is defined as a sequence of instructions that when executed in the specified sequence, the desired results are obtained. In order to qualify as an algorithm, a sequence of instructions must possess the following characteristics:

1. Each instruction should be precise and unambiguous.
2. Each instruction should be executed in a finite time.
3. One or more instructions should not be repeated infinitely. This ensured that the algorithm will ultimately terminate.
4. After executing the instructions, the desired results are obtained.

Advantages of an Algorithm

1. It is easy to understand and easy to write.
2. It is an effective method of solving certain sets of problems
3. No need to knowledge of a specific programming language
4. It is easy to detect and solved the mistake from the algorithm.

Disadvantages of an algorithm

1. It is time consuming process.
2. There is only a manual method to check whether an algorithm is correct or not.

Flow Chart

A flowchart is a pictorial representation of an algorithm. Programmers often use it as a program-planning tool for visually organizing a sequence of steps necessary to solve a problem using a computer. It uses boxes of different shapes to denote different types of instructions. The actual instructions are written within these boxes using clear and concise statements.

Flow Chart Symbol

Only a few symbols are needed to indicate the necessary operations in a flowchart. The ANSI (American National Standards Institute) has standardized the basic flowchart symbols.

Terminal

The terminal symbol indicates the beginning (start), end (stop) and pauses (halt) in a program's logic flow. It is the first and the last symbol in a flowchart.

Input/Output

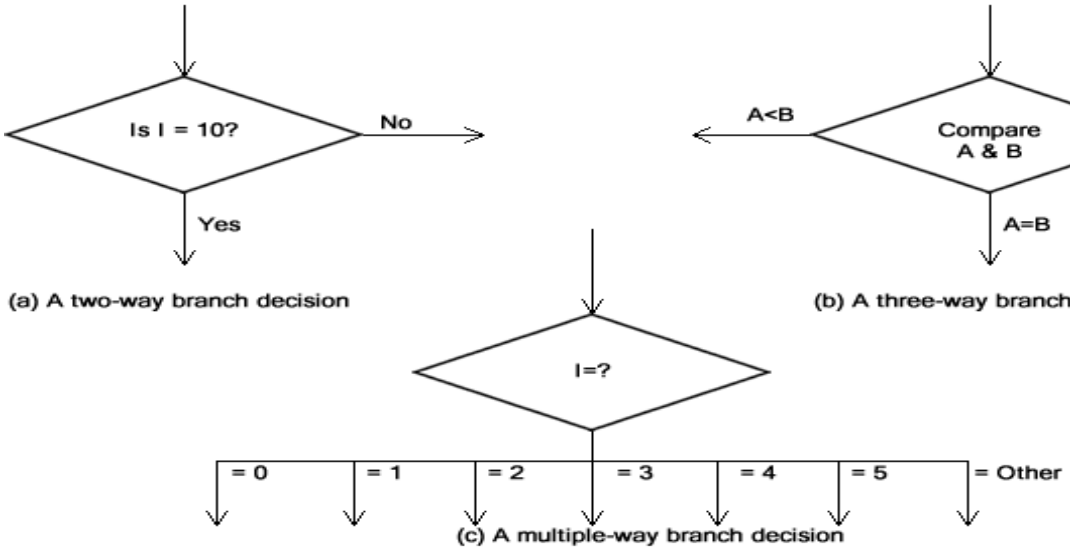
The input/output symbol denotes any function of an input/output nature in a program. Hence, all program instructions to input/output data from any type of input/output device (such as keyboard, mouse, scanner, monitor, printer etc.) are indicated with input/output symbols in a flowchart. Even instructions to input/output data from/to a storage device (such as disk, tape etc.) are indicated with input/output symbols.

Processing

A processing symbol represents arithmetic and data movement instructions. Hence, all arithmetic processes of adding, subtracting, multiplying, and dividing are indicated by a processing symbol in flowchart. The logical processes of moving data from one location of the main memory to another (assignment statement) are also denoted by this symbol. When more than one arithmetic and data movement instructions are executed consecutively, they are normally placed in the same processing box, and they are assumed to be executed in the order of their appearance.

Decision

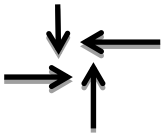
The decision symbol indicates a decision point, that is, a point at which a branch to one of two or more alternative points is possible. Following figure shows three different ways of using a decision symbol.



Note that the criterion for making a decision is clearly indicated within the corresponding decision box. Moreover, the condition upon which each of the possible exit paths is executed should be identified, and all the possible paths should be accounted for. During execution, the appropriate path is followed depending upon the result of the decision.

Flow lines

Flow lines with arrowhead indicate the flow of operation, that is, the exact sequence in which the instructions are executed. The normal flow of a flowchart is from top to bottom and left to right. Arrowheads are required only when the normal flow is not followed. However, as a good practice, and to avoid ambiguity, flow lines are usually drawn with an overhead at the point of entry to all flowchart symbols.



Connectors

Whenever a flowchart becomes so complex that the number and direction of flow lines are confusing, or it spreads over more than one page, it is useful to utilize connector symbols as a substitute for flow lines. This symbol represents an entry from, or an exit to another part of the flowchart. A connector symbol is a circle with letter or digit places within it to indicate the link. A pair of identically labeled connector symbols is used to indicate continues flow when the use of a line is confusing. Hence, two connectors with identical labels serve the same function as a long flow line.

Rules for Flowchart

ANSI had recommended a number of general rules and guidelines to help standardize the flowcharting process. Various computer manufacturers and data processing departments usually have similar flowcharting standards. The basic flowcharting rules and guidelines are as follows:

1. First chart the main line of logic, and then incorporate details.
2. Maintain a consistent level of details for a given flowchart.
3. Do not chart every detail; otherwise, the flowchart will only be a graphic representation of all the steps of the corresponding program. A reader interested in details can refer to the program itself.
4. Use common statements that are easy to understand for words within flowchart symbols.
5. Be consistent in using names and variables in the flowcharts.
6. Go from left to right and top to bottom in constructing flowcharts.
7. Keep the flowchart as simple as possible. Crossing of flow lines should be avoided.
8. If a new flowcharting page is needed, break the flowchart at an input or output point. Moreover, use properly labeled connectors to link the portions of the flowchart on different pages.

Advantages of Flowcharts

1. **Better Communication:** a flowchart is a pictorial representation of a program. Therefore, it is easier for a programmer to explain the logic of a program to some other programmer, or to his/her boss through a flowchart rather than the program itself.

2. **Proper Program Documentation:** Program documentation involves collecting, organizing, storing, and otherwise maintaining a complete historical record of programs, and other documents associated with a system. Good documentation is needed for the following reasons:
    a. Documented knowledge belongs to an organization, and does not disappear with the departure of a programmer.
    b. If a project is postpones, documented work helps in restarting the project later from the stage at which it was stopped.
    c. If a programmer has to modify a program, documented work provides him/her a more understandable record of what was originally done.

3. **Efficient coding:** Once a flowchart is ready, programmers find it very easy to write the corresponding program because the flowchart acts as a road map for them. It guides them to go from the starting point of the program to the final point, ensuring that no steps are omitted. The ultimate result is an error free program developed at a faster rate.

4. **Systematic Debugging:** A flowchart is very helpful in detecting, locating, and removing mistakes in a program in a systematic manner because programmer find it easier to follow the logic of the program in flowchart form. The process of removing errors in a program is known as debugging.

5. **Systematic Testing:** Testing is the process of confirming whether a program will successfully do all its intended jobs under the specified constraints. For testing a program, the program is executed with different sets of data as input to test the different paths in the program logic.

Disadvantages of Flowcharts

1. Flowcharts are very time consuming and laborious to draw with proper symbols and spacing, especially for large complex programs.

2. Owing to the symbol-string nature of flowcharting, any change or modification in the program logic will usually require a completely new flowchart. Redrawing a flowchart being a tedious task, many programmers do not redraw or modify the corresponding flowcharts when they modify their programs. This leaves a program and its flowchart in an inconsistent state. That is, the logic used in the program and that shown in its flowchart do not match. This defeats the purpose of use of flowcharts as documentation support for programs.

3. There are no standards determining the amount of detail that should be included in a flowchart.

Computer Languages

A computer can only do what a programmer asks it to do. To perform a particular task programmer writes a sequence, called the program. The command or instruction can be given to the computer to perform a certain specified operation on the given data. Now as we know only human languages and computer knows only machine language, we need some media through which we can communicate with the computer. So we can complete our desired task. That media is Language. So, languages are tools human can use to communicate with the hardware of a computer system.

Each language has a systematic method of using symbols of that language. In English, this method is given by the rules of grammar. Similarly, the symbols of particular one computer language must also be used as per set of rules which are known as the "Syntax" of that language, the language which you are using.

Language Generation

- The first generation languages or **1GL** are **machine language** or **low level language**.
- The second generation languages or **2GL** are **assembly languages**.
- The third generation languages or **3GL** are **high-level languages** such as C.
- The fourth generation languages or **4GL** are languages that consist of statements similar to statements in a **human language**. Fourth generation languages are commonly used in database programming and scripts.
- The fifth generation languages or **5GL** are programming languages that contain **visual tools** to help develop a program. A good example of a fifth generation language is Visual Basic.

Machine Language or low level language (1GL)

Machine language is made up of only two symbols "0" and "1" with all its different combinations, So all the instruction are coded in 0s and 1s. There is a specific binary code for each instruction. The binary code for certain operation differs from computer to computer. Each microprocessor has its own instruction and set and corresponding machine codes.

An instruction prepared in any machine language has two-part format, as shown as under:

Opcode (Operation Code)	Operand (Address /

e.g.

0010 (Store)	0000001110 (Location)

Operation Code is the function that must be performed and **Operand** are the variables involved in this functions.

Machine language is directly understood by the computer because it is made up of only two symbols "0" and "1". So no translation is required.

Advantages

- Programs can be executed immediately upon completion because it doesn't require any translation.
- No extra storage space is needed.
- Programmer has complete control over the performance of the hardware.

Disadvantages

- Tedious to program
- Time consuming to code
- Error prone
- Operation codes have to be memorized
- Assignment of memory is done by programmer
- Time consuming for development
- Programs development is machine dependent
- Preparation of programs was slow and costly

Assembly Language (2GL)

Machine language was tedious to code and error was expected to arise in bulk. To ease the programmer's burden, mnemonic codes and symbolic addresses were developed. Letter symbols were substituted for basic machine language commands codes. That is sub or s for subtract, mvc for move character. Symbolic addresses are used in place of actual machine address or location.

Format of assembly language is similar to machine language:

Mnemonic Code	Symbolic

Examples of commands are MOV, ADD, SUB, INC, etc. Examples of Variable names are SUM, MARKS, AVERAGE, etc. Examples of Register Names are AX, DX, CX, etc.

Example of Assembly language instruction:

ADD	AX	NUM1

This instruction adds value of NUM1 to the **AX (Accumulator Register)**.

The symbolic language made program writing so much easier for the programmers but it must be translated into machine code before being used for operation. The translation is actually done by a special translating program.

Advantages

- Easier to code and understand programs as compared to machine language programs.
- Task of memory management and allocation not done by program (taken care of by the assembler)
- Can use Macros (Macro is a bunch of instruction referred as a single name)

Disadvantages

- Programs have to be translated before execution.
- Translation of programs takes up time.
- Programs are machine dependent. (Restricted for use on that machine)
- Additional storage area needed for the source programs and object codes.

Example of Assembly Language

- Microsoft Assembly Language (MASM), Turbo Assembler

High Level Language (3GL)

To write a program in any languages, a programmer has to remember all the operation codes of the computer and know in detail what each code does and how it affects the various registers of the computer. However, we have also seen that in order to write good computerized programs the programmer should mainly concentrate on the logic of the problem rather than the details of internal structure of the computer.

This is a term which describes programming languages that do not control the functions of the computer's components so closely. The instruction here is even more English like, and as a result, the software industry began to explode with varying applications and systems software. High level languages have the following features:

Self-Documenting

Extensive use of English words and English-like statement structures make the codes easy to read and remember.

Requires Translation

Since, instructions cannot be directly understood by the computer, they have to be converted into machine codes. Translations like compilers and interpreters are used to convert each instruction into many instructions.

Syntax

Syntax is the rule of the language to be followed. Statements have to be coded in a particular manner and certain keywords cannot be used for purposes other than what they were meant for.

Library Subroutines

It contains pre-defined procedures and functions to perform tasks that are commonly needed by programmers.

Machine Independent

Programs written for 1 machine can be run on another with little or no modifications.

Procedure Oriented

Computations are identified in sequences of instructions. Instructions identify not only the kind of ('what') tasks to be performed but indicates the way ('how') each task can be achieved.

In high level language source program is compiled and object program is generated. After object program has been generated linker (used to incorporate any library subroutines) links library subroutines and at last it generates an Executable Program. It uses compiler or interpreter are the translators used for translation of program into machine language.

Advantages

- Wide range of 3GL available that support the development of software with varying nature (e.g. system software, business applications, etc.)
- Allows programmers to have full control over the way tasks are performed.
- Highly portable compared to Assembly Language.

Disadvantages

- Programmers must be adequately trained.
- Databases / File oriented applications are tedious to code.
- Programmers must identify how tasks are to be achieved.

Example of 3GL

BASIC (Beginners All Purpose Symbolic Instruction Code), **COBOL** (Common Business Oriented Language), **PASCAL**, **FORTRAN** (Formula Translation), **C**, **RPG** (Report Program Generator), PL/1 (Programming Language 1), etc.

Translators

To convert your higher-level language program into machine level language program, there is use of special software or program that is known as Translator. Translator can be divided into two categories: Compilers and Interpreters. One more kind of translator used in assembly language to assemble the program is called assembler.

Compiler

Complier is a program, which checks the syntax error in your program and converts your program into machine language from higher-level language. If there is any error in any line of program compiler give error message and doesn't convert your program into machine level language. But it is faster than the interpreter. Compiler is used in C, C++ program.

Interpreter

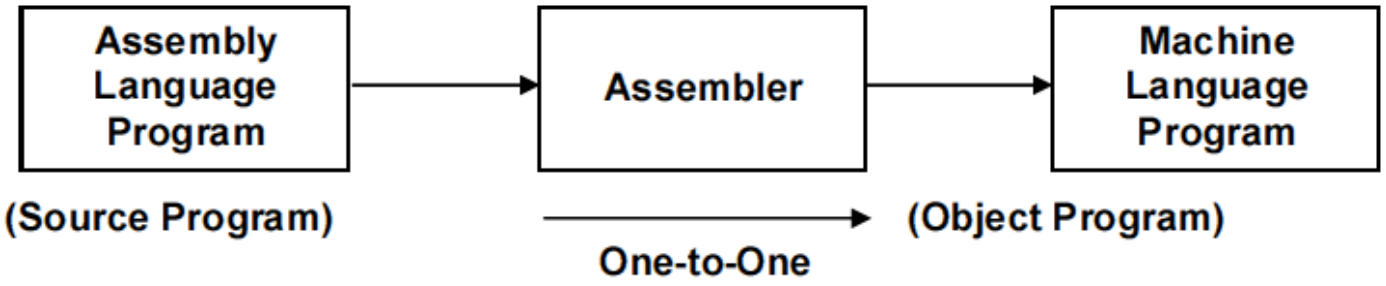
Interpreter is a program which checks your program line by line and after checking first line it converts into machine language and execute it so it is useful for the line by line tracing of your program. It is slower than compiler. It is used in Java program.

Assembler

Assembler is a special program (translator) which translates symbolic operation codes into machine codes, and symbolic address is addressed into an actual machine address.

It allocates areas of main storage. It is able to produce a printed listing of the Object Programs together with comments. The symbolic instruction is translated into one machine code, which is one feature which distinguishes a low-level from a high level language.

A symbolic program written by a programmer in assembly language is called a **source program**. After the source program has been converted into machine language by as assembler, it is referred to as an **object program**.



Editor

Editor is a system program allows adding, deleting and manipulating the text. It provides several commands to add delete and manipulate the text. It provides also the facility of save, save as, print as well as facility to copy, cut, paste, select, find, replace, delete the text.

In market several editors are available such as DOS editor, notepad, WordPad, etc. Many programming complier also provides the facility of editor such as C editor, VB editor, etc. which includes the facility to compile, execute, trace program.

In C we are generally using Turbo C++ compiler, Borland C++ compiler, Microsoft C compiler, or many others. In Turbo C++ compiler there are several menus having File, Edit, Search, Run, Compile, and Help. File menu provides the facility to open, save, close file. Edit menu allows us to copy, cut, paste, find, replace, and select, etc. operation on text. Search menu provides the facility to search, replace the text. Compile menu provides the facility to compile the program. Run menu allows us to run (execute) our program. Help menu provides the help on different topics.

The Turbo C++ Editor

The turbo C++ offers everything you need to write, edit, compile, link, run, manage and debug your programs. You require TC.EXE file to activate the Turbo C++ i.e. TC editor. The menu bar at the top of the screen is the gateway of the menus.

To go to the menu bar, there are three different ways:

1. Press F10 key
2. Press Alt+ch, where ch is the first character of the menu options
3. Click anywhere on it.

The once you open this editor, it has following menu options:

-File -Edit -Search -Run -Compile
-Debug -Project -Options -Window -Help

File Menu : The file menu provides commands for creating new files, opening Existing files, saving files, changing directories, printing files, shelling to DOS & quitting Turbo C++.

Edit Menu : The edit menu provides commands cut, copy and paste text in edit Windows. You can also undo changes and reverse the changes you have just undone.

Search Menu : The search menu provides commands to search for text. Function declarations and error locations in your files.

Run Menu : The Run menu provides commands to run your program and to start and end debugging sessions.

Compile Menu : The Compile menu provides commands to compile the program in the active edit window or to make or build your project.

Debug Menu : The debug menu provides commands to control all the features of the integrated debugger.

Project Menu : The Project menu contains all the project management commands to do the following:
- Create or open a project
- Add or delete files from your project
- Set operations for a file in the project
- View included files for a specific file in the project

Options Menu : The Option menu contains for viewing and changing various default setting in Turbo C++.

Window Menu : The Window menu contains window management commands.

Help Menu : This Help menu provides access to the online help system.

Editor Commands

The Turbo C++ offers variety of commands to do several tasks. Depending upon their functions, commands are classified into following categories:

(1) Cursor Movement Commands
(2) Insert & Delete Commands
(3) Block Commands
(4) Miscellaneous Commands

Cursor Movement Commands

Command	Function
→	To move cursor one character left
←	To move cursor one character right
↑	To move cursor one line up
↓	To move cursor one line down
PgUp	To move cursor one page i.e. screen up
PgDn	To move cursor one page i.e. screen down
Ctrl+W	To scroll up one line
Ctrl+Z	To scroll down one line
Ctrl + A or Ctrl + ←	To move cursor one word left
Ctrl + F or Ctrl + →	To move cursor one word right
Home	To move cursor at beginning of line
End	To move cursor at end of line
Ctrl + Home	To move cursor at the top of window
Ctrl + End	To move cursor at the end of window
Ctrl + PgUp	To move cursor at the top of file
Ctrl + PgDn	To move cursor at the bottom of file

Insert and Delete Commands

Command	Function
Delete	To delete the character
BackSpace or Shift+Tab	To delete character to left
Ctrl + Y	To delete the line
Ctrl + T	To delete the Word
Ctrl + Q Y	To delete line from current cursor position to end
Ctrl + N	To insert the line
Insert	To make insert mode on/off

Block Commands

Command	Function
Ctrl + K B	To set beginning of the block
Ctrl + K K	To set end of the block
Ctrl + K C	To copy the block
Ctrl + K V	To move the block
Ctrl + K Y	To delete the block
Ctrl + K H	To hide the marked block i.e. Unhide the block
Ctrl + K W	To write a block to the disk
Ctrl + K R	To read a block from the disk

Miscellaneous Commands

Command	Function
Ctrl + Q A	Search & Replace
Ctrl + Q F	Search
Ctrl + L	Search again
Ctrl + [or Ctrl+]	Pair matching

Hot Keys

Turbo C++ provides hot keys, or shortcut for your convenience. These hot keys can be classified into following category:

1. General hot keys
2. Menu hot keys
3. Editing hot keys
4. Online Help hot keys
5. Window management hot keys
6. Debugging/Running hot keys

General hot keys

Command	Function
F1	Displays a help screen
F2	Saves the file that in the active edit window
F3	Brings up a dialog box so you can open file
F4	Runs your program to the line where the cursor in positioned

Menu hot keys

Command	Function
Alt + Spacebar	Takes you to the system menu
Alt + C	Takes you to compile menu
Alt + D	Takes you to Debug menu
Alt + E	Takes you to Edit menu
Alt + F	Takes you to File menu
Alt + H	Takes you to Help menu
Alt + O	Takes you to Options menu
Alt + P	Takes you to Project menu
Alt + R	Takes you to Run menu
Alt + S	Takes you to Search menu
Alt + W	Takes you to Window menu
Alt + X	Exits Turbo C++

Editing hot keys

Command	Function
Alt + Backspace	Undo
Shift+Alt+Backspace	Redo
Shift+Delete	Places selected text in Clipboard, deleted selection
Shift+Insert	Pastes text from Clipboard into the active window
Ctrl+Delete	Remove selected text from window
Ctrl+Insert	Copies selected text to Clipboard

Online help hot keys

Command	Function
F1	Opens a context-sensitive help screen
F1 F1 (Two time)	Brings up Help on Help
Shift + F1	Brings up Help Index
Alt + F1	Displays previous Help Screen
Ctrl + F1	Calls up language-specific help in the active edit window

Window Management hot keys

Command	Function
Alt + #	Displays a window, where # is the number of window you want to view.
Alt + 0	Displays a list of open windows
Alt + F3	Closes the active window
Alt + W T	Tiles all open windows
Alt + F5	Displays user screen
Ctrl + F5	Changes size or position of active window

Debugging/Running hot keys

Command	Function
Alt + F4	Opens an inspector window
Alt + F7	Takes you to previous error
Alt + F8	Takes you to next error
Alt + F9	Compiles to .OBJ
Ctrl + F2	Resets running program
Ctrl + F3	Brings up call stack
Ctrl + F4	Evaluates an expression
Ctrl + F7	Adds a watch expression
Ctrl + F8	Sets or clears conditional breakpoints
Ctrl + F9	Runs program

Question Bank

Multiple Choice Questions

1. _____is a step by step approach to solve any problem.
   (a) Process (b) Programming Language (c) Algorithm (d) Compiler
2. _____is a pictorial representation of an algorithm.
   (a) Data Diagram (b) Flow Chat (c) Pie Chart (d) Program
3. The process of walking through a program's logic on paper before you actually write the program is called _____.
   (a) Desk checking (b) flowcharting (c) pseudo coding (d) testing
4. What symbol is used to represent output in a flowchart?
   (a) Square (b) circle (c) parallelogram (d) triangle
5. What is the standard decision symbol for a flowchart?
   (a) circle (b) lozenge (c) diamond (d) square
6. Mnemonic a memory trick is used in which of the following language?
   (a) Machine Language (b) Assembly Language (c) High Level Language (d) None of above
7. The translator program used in assembly language is called _____.
   (a) Compiler (b) Interpreter (c) Assembler (d) Translator
8. _____ is easily re-locatable language.
   (a) Machine Language (b) Assembly Language (c) High Level (d) Medium Language
9. Which of the following is called low level languages?
   (a) Machine Language (b) Assembly Language (c) Both of the above (d) None of above
10. Which of the following is problem oriented language?
    (a) High level language (b) Machine language (c) Assembly language (d) Low level language
11. A compiler is a translating program which
    (a) Translates instruction of a high level language into machine language
    (b) Translates entire source program into machine language program|
    (c) It is not involved in program's execution
    (d) All of above
12. Which of the following is machine independence program?
    (a) High level language (b) Machine language (c) Assembly language (d) Low level language
13. Which is the limitation of high level language?
    (a) Lower efficiency (b) Machine dependence (c) Machine level coding (d) None of above
14. High level language is also called _____
    (a) Problem Oriented Language (b) Business Oriented Language
    (c) Mathematically Oriented Language (d) All of above
15. C language is _____
    (a) High level language (b) Machine language (c) Assembly language (d) Low level language

Questions

1. What is an algorithm? List Characteristics of an algorithm.
2. Write advantages and disadvantages of an algorithm.
3. Write an algorithm/flowchart for following.
   a. To find simple interest. Hint: SI = (P * R * N)/100
   b. To find maximum of given three numbers.
   c. To find out N! (Factorial of N).
   d. To find weather given number is odd or even. To find sum of odd value and even value digits of a given number.
   e. To find out minimum from N numbers.
   f. To check whether inputted number is prime number or not.
   g. To check whether inputted number is palindrome number or not.
   h. To check whether inputted number is Armstrong number or not.
   i. To print N terms of Fibonacci series.
   j. Sum=$1^2 + 2^2 + 3^2 + 4^2 + 5^2 + 6^2 + 7^2 + \ldots$ and so on.
4. What is flowchart? List Symbols used in flowchart.
5. Write advantages and disadvantages of flow chart.
6. List symbols used to draw flow chart. Explain any one.
7. List Rules to draw flow chart.
8. What is an Editor? Give 3 examples of Well Know Editors.
9. List Languages for all generation.
10. Write advantages and disadvantages for following.
    a. Machine Level or Low Level Language (1GL)
    b. Assembly Language (2GL)
    c. High Level Language (3GL)
11. What is Translator? List all translators.
12. Explain any one translator in detail.