Introduction to Algorithm and programming Language.

The word Algorithm derives from the name of a mathmeticiran. Mohamad ibn-musa.

Algorithm is a well defined instruction for performing some task.

The Algorith gives the logic of programme. i.e. step-by-step description of how to solve the problem.
once the Algorithm is develop then we can implement it in any High level language. like C, C++ etc.

* Qualities of Good algorithm:-

(i) Input and output should be defined preciesely. pricisely means acorately and exactly.

(ii) Each step in algorith must be clear. and without doubtful

(iii) Algorith should be must effective among many different ways to solve a problem.

(iv) An algorith should not have a computer code. Instead the algorithm should be written in such a way that it can be used in similar programming languages.

* Key features of Algorithm:-

(i) **Sequence:-** Sequence means each steps of the Algorithm is executed in the specified order.

Algorithm to add two Numbers:-

Step-1 - Input the first Number as A.
Step-2 - Input the second Number as B.
Step-3 - Set sum = A+B.
Step-4 - print sum.
Step-5 - End.

(ii) **Decision:-** Decision statements are used when the outcome of the process depend on some condition. Example if x=y then printf Equal.

```
if condition
        then process1
else    process2
```

Algorithm to check Equality of two numbers.

→ **Step-1:-** Input the first Number As A

Step-2:- Input the second number as B

Step-3:- if A = B
        then print "Equal"
else
        print "Not equal"

Step 4:- End

(iii) **Repetation**:- It involves executing one or more steps for a number of times. It can be used with construct for, while, do-while loops.

Algorith to print first 10 Natural numbers.

Step-1- [initialize.] set I=0, N=10

Step-2- Repeat step while I<=N

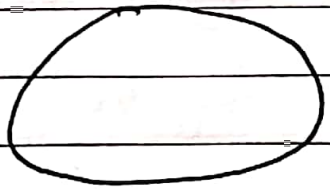
Step-3- print I

Step-4- End

* **Flow chart**:- A flow chart is a Graphical Representation of an Algorithm.

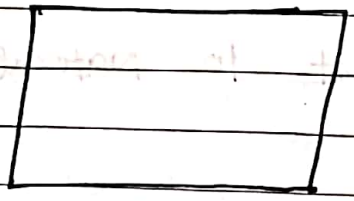
Basic symbols of flow chart:-

(i) **Terminal**:- The oval symbol indicates start, stop and Halt in a programme's logic flow.

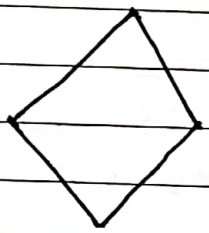


(ii) **Input/output**:- A parallelogramme denotes any fn of input/output type.

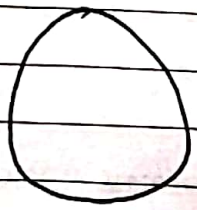
(iii) Processing:- A box Represents Arithmetic instructions. All Arithmetic process such as Adding, Subtracting, multiplication and division are indicated by action or process symbol.



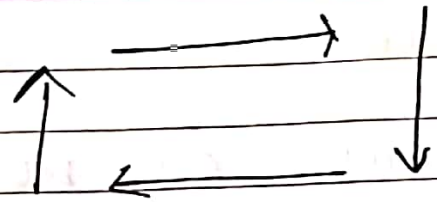
(iv) Decision:- Diamond symbol represents decision point. Decision based operation such as yes/no question or true/false indicated by diamond in flow chart.



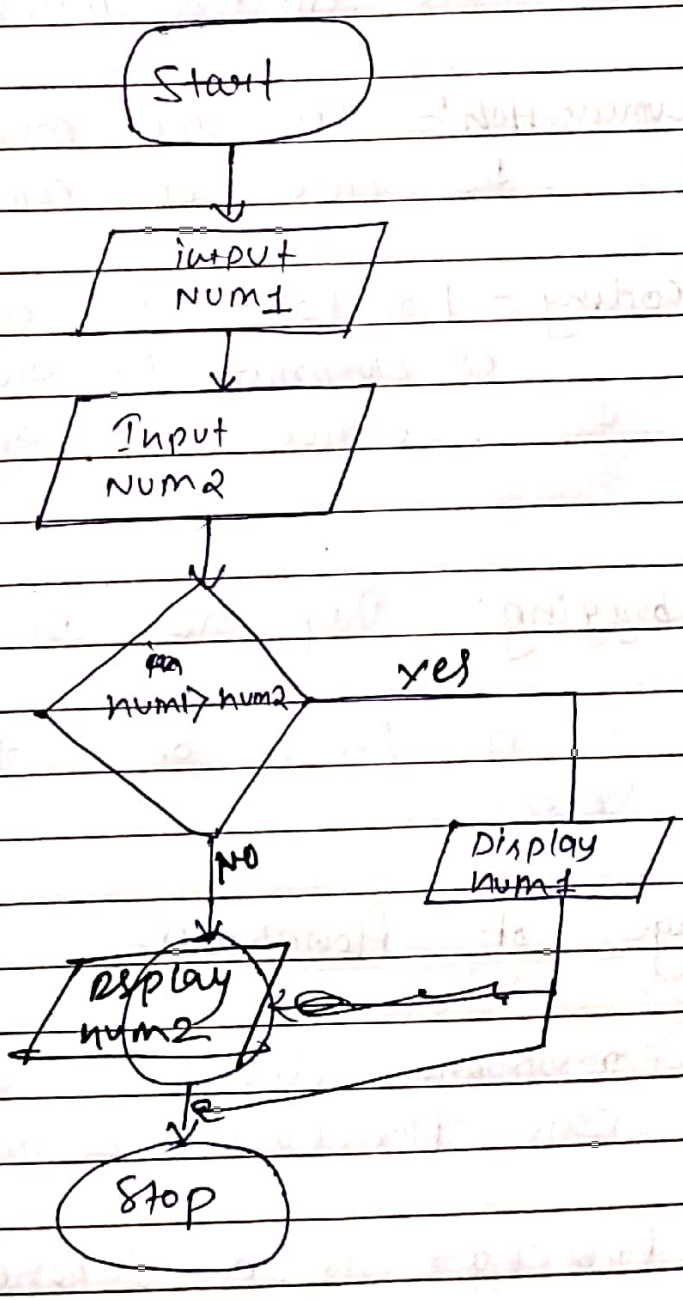
(v) Connectors:- Whenever the flow chart become complex or it spread over more than one pages, It is useful to use connectors to avoid any confusion. it is represented by a circle.



(vi) Flow line or Arrows:- Arrows indicated the exact sequence in which instructions are executed.



→ Flow chart to input two numbers from user and display the largest of two numbers.



* Advantage of Flowchart:

(i) Communication:- flowcharts are better way of communicating.

(ii) Effective Analysis:- They Help to analys the problem in a more effective manner.

(iii) proper Documentation:- They are more helpful in the case of complex programme.

(iv) Efficient Coding:- Flowchart act as a guide or blueprint for the programmers to code the solution in any programming language.

(v) proper Debugging:- They help in Debugging process.
debugging is a process of indentifying and removing errors.

* Disadvantage of Flowchart:-

(i) Sometimes programme logic is quite complicated. In that case flowchart become complex.

(ii) Drawing flowchart is a laborious and time - consuming

(iii) As the flowchart can't be typed, reproduction of flowchart become a problem.

Algorithm

* Analysis of ~~Flowchart~~:-

Analysis of Algorithm is the determination of the amount of time and space resource required to execute it.

Algorithm

* Analysis of Flowchart:-

Analysis of Algorithm is the determination of the amount of time and space resource required to execute it.

Programming languages

A programming language is a language specifically designed ~~for~~ to Express computation that can be performed using computer.

programming language are used to create programms that control the behaviour of system to Express Algorithm.

The term programming language refers to high level language such as BASIC, C, C++, COBOL, FORTRAN e.t.c.

Assembly level language:- Assembly level lang. are similar machine level lang, but they are easier to programme in ~~beee~~ compared to machine lang. because they allow programme to substitute names for numbers.

* Generation of programming language.

The concept of Generation of programming lang. is closely connected to the advance in technology.

There are 4 Generation of programming language. includes machine lang., Assembly lang., high-level lang., very-high level lang.

(i) First Generation of programming language:-
(Machine-level lang.)

The first Generation of programming lang. was introduced in 1940. It is also known as Binary language. This lang. is made up of 0 and 1. It is the only lang. a computer is able to understand without using translation programmes.
The programmes code in 0 and 1 and transfer it to computer by using punch card, or punch tapes.
This language is close to machines.

→ Advantage:-

(i) Machine lang. makes fast and efficient use of the computer.

(ii) It does not require, translator to translate codes.

Disadvantages:-

(i) All operation code have to be remembered, which are in the form 10001, 0011, 10010110 e.t.c.

(ii) All memory address have to be remembered.

(iii) It is hard to find errors in programme.

2. Second Generation programming lang (2GL)
(Assembly lang. or Low level lang.):-

Second Generation programming languages introduced in 1950.
Alphanumeric codes instead of 0 and 1 are used in this lang. for programming. e.g. ADD for addition, SUB for subtraction e.t.c.
This lang. is also close to machine.
programmes are translated into machine lang. using Assemblers.
Used to develop kernals, hardware drivers e.t.c.
This lang. is Not portable.

Advantage:-

(i) Assembly lang. are easier to understand and use as compared to the machine lang.

(ii) It is easy to locate errors and correct Errors.

(iii) It is easily modified.

Disadvantages:-

(i) Like machine lang. it is also machine dependent

(ii) Since, it is machine-dependent, the programmers also need to understand the hardware.

E.g. of Assembly lang. are:

RISC, CISC, X86

3. Third Generation language:- (3GL) (High-level lang)

Introduced in 1960s
High level lang. enables people to write programs easily. The main motive is to write programs in Native lang English.
These are Symbolic language that use english words and mathematical Symbols Rather than Alphanumeric codes.
These languages are close to Humen.
In this generation lang. the compiler was first introduced to convert the language into machine lang.)

The first High-level language developed for Mark 1 Computer.

e.g. FORTRAN first HLL developed by IBM., BASIC, COBOL, ALGOL

Advantages:-

(i) Easy to learn and understand the lang. Compared to Assembly lang.

(ii) Easy to find errors.

(iii) They are machin-independent i.e. runs on any computer Not for a single Computer

Disadvantage:-

(i) It take more time to compile a programme than Assembly and machine lang.

4. Fourth Gen prog. lang. :- (very High-level lang) (4GL) or Non-procedural lang.

Introduced in 1970s.
These lang. are closed to Natural lang. becoz almost coding done in English lang.
The code is Easier to maintain in this language
less coding to do as compared to HLL of any ~~to~~ programme.

Increas the speed of developing program.

e.g. SQL, MATLAB

5. Fifth Gen lang. Use AI to make computer smarter.

e.g. JARVIS

Compilar, Assembler, Interpreter

(i) Assembler :- The Assembler is used to translate the programme written in Assembly lang. into mechine code.

Source code → [Assembler] → object code

(ii) Compilar :- A compilar is a software that transforms source code in a programming lang. into ~~Another~~ mechine lang.
It creates object file (.OBJ) during compilation.
object file converts Binary codes.
Compilar Scan whole code and then ~~to~~ to Generate object file or Executable file.

Source file → [Compiler] → Object file.

(iii) Interpreter :- Interpreter translate source code line by line and send for immidiate execution. It execute line by line.
there is no object file Generation in Interpreter.
Execution time is slower in interpreter becoz it ~~execute~~ translate and execute line by line.

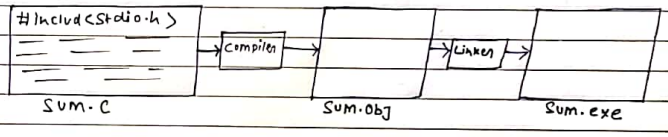
Source file → [Interpreter] → Operating system.

Difference b/w compiler and Interpreter

	Compiler	Interpreter
(i)	Compiler converts Entire source code of a programming lang. into executable file.	(i) Interpreter takes a source program and runs it line by line, translating each line as come to it.
(ii)	Compiler takes large Amount of time to Analyze the Entire Source code.	(ii) Interpreter takes less Amount of time to Analyze the source code.
(iii)	Execution time is faster.	(iii) Execution time is slower becoz it translate and Execute line by line.
(iv)	Compiler generates the Error message only after Scanning the whole program. So debugging is comparatively hard as the error can be present any where in the program.	(iv) Its Debbugging is Easier as it continues translating the programme until the error is met.
(v)	Generates intermediate object file.	(v) No Intermediate object file is generated.
	e.g. C, C++, Java	e.g. python, perl

* **Linker:-** Some buil-in Header files are Stored in in HLL. These libraries are predefined and ~~and~~ these contains Basic functions which are Essential to for execution of the programs these function are linked to the libraries by a programme called Linker.

The Linker Links the Required libraries to programme and the creates Executable file.



* **Loader:-** Loader is a program that loads program into system memory. Loader is a part of operating system that is responsible for loading the programs into memory and prepares them for execution.