

Introduction to programming: -**Concept of algorithm: -**

- It is list of instructions specifying a particular description of a step by step process that terminates after a finite number of steps for solving an algorithm problem producing the correct answer in the end.
- It is used to solving the problems.
- A finite set of an instruction that specifies a sequence of operation to be carried out in order to solve a specific problem.
- An unambiguous procedure specifying a finite number of steps to be taken.
- Ordered set: Each instruction is well-defined.
- Produce a result: Effectiveness

Methods of specifying algorithm: -

- **Pseudocode:** Specifies the steps of algorithm using essentially natural language of superimposed control structure.
- **Flowchart:** A traditional graphical tool with standardized symbols. Show the sequence of steps in an algorithm.

Properties of algorithm: -

- **Finiteness:** There is an exact number of steps to be taken and has an end.
- **Absence of ambiguity:** Means that every instruction is precisely described and clearly specified.
- **Sequence of Execution:** Instructions are performed from top to bottom.

Input: READ, OBTAIN, GET

Output: PRINT, WRITE

Compute: LET, COMPUTE, CALCULATE, DETERMINE

Initialize: SET, INITIALIZE

Add one: INCREMENT

Subtract one: DECREMENT

- **Input and output:** Defined the unknowns of the problem is specified and with the expected outcome.
- **Effectiveness:** The solution prescribed is guaranteed to give a correct answer and that the specified process is carried out.
- **Scope definition:** Applies to a specific problem or class of problem.

Ex. 1. Write the algorithm to find the sum of two given numbers.

Step 1: Read A , B
 Step 2: Let/Compute Sum= A+B
 Step 3: Print Sum
 Step 5: Stop.

Ex. 2. Write the algorithm to find the sub of two given numbers.

Step 1: Read A , B
 Step 2: Let/Compute Sub= A-B
 Step 3: Print Sub
 Step 5: Stop.

Ex. 3. Write the algorithm to find the Mul of two given numbers.

Step 1: Read A , B
 Step 2: Let/Compute Mul= A*B
 Step 3: Print Mul
 Step 5: Stop.

Ex. 4. Write the algorithm to find the Div of two given numbers.

Step 1: Read A , B
 Step 2: Let/Compute Div= A/B
 Step 3: Print Div
 Step 5: Stop.

Ex. 5. Write the algorithm to find the sum, sub,mul and Div of two given numbers.

Step 1: Read A , B
 Step 2: Let/Compute Sum= A+B
 Step 3: Let/Compute Sub= A-B
 Step 4: Let/Compute Mul= A*B
 Step 5: Let/Compute Div= A/B
 Step 6: Print Sum
 Step 7: Print Sub
 Step 8: Print Mul
 Step 9: Print Div
 Step 10: Stop.

Ex 6: Develop an algorithm to interchange the values assigned to two variables A and B. (For example, if A=2 and B=3, after interchange, it should be A=3 and B=2).

Algorithm: To interchange the values.

Step 1. Read A = 2 B = 3
 Step 2. TEMP = A
 A = B
 B = TEMP
 Step 3. Print A, B
 Step 4. Stop

CONDITIONAL [if-else] :

->In a conditional statement you make a test.

->The result of the test is a Boolean - either True or False.

->If the result of the test is True you take a certain course of action and if the result of the test is False you take another course of action.

Syntax:

```
IF condition THEN
    sequence 1
ELSE
    sequence 2
```

->If the condition is True sequence 1 is executed, otherwise sequence 2 is executed.

->The ELSE sequence is optional.

Example: Write an algorithm to print the larger of two numbers.

Step 1: Start

Step 2: Read a, b . /* a, b two numbers */

```

Step 3: If a>b then      /*Checking */
        Display "a is the largest number".
    Else
        Display "b is the largest number".

```

Step 4: Stop.

Example: Write an algorithm to check the given number is Odd Or Even.

Step 1: Start

Step 2: Read no .

```

Step 3: rem=no %2      /*To Find Reminder given no*/

```

```

Step 3: If (rem==0) then /*Checking */
        Display "No is Even".
    Else
        Display "No is Odd".

```

Step 4: Stop.

LOOPS [WHILE/FOR]:

-> A loop is a sequence that gets executed several times.

->A complete execution of a sequence is called an iteration of the loop.

->There are two main loop constructs - WHILE and FOR.

while loop in programming

The while loop is used to execute the loop body until a specific condition is **false**. We mainly apply this idea when we don't know how many times the loop will execute.

The while loop consists of a loop condition, a block of code as a loop body, and a loop update expression if required. First, the loop condition is evaluated, and if it is true, code within the loop body will be executed. This process repeats until the loop condition becomes false. For better intuition, while loop can be thought of as a repeating if statement.

Syntax:

```

WHILE condition
sequence
ENDWHILE

```

The sequence is executed as long as the condition is True. The loop terminates when the condition is False.

Example 1: Write an algorithm to print 1 to 5 no's using while.

Step 1: Start

Step 2: Read no=5 .

Step 2: Let/Compute no=1

Step 3: while (no<=5) then /*Checking */
 print no

Step 4: no=no+1 /*To increment by 1 in every no */
 Goto Step 3




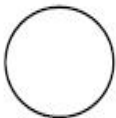


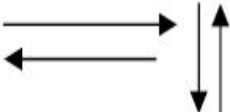
Step 4: Stop.

Concept of flow chart: -

- Flowcharting is a tool developed in the computer industry, for showing the steps involved in a process.
- A flowchart is a diagram made up of boxes, diamonds and other shapes, connected by arrows - each shape represents a step in the process, and the arrows show the order in which they occur.
- Flowcharting combines symbols and flow-lines, to show figuratively the operation of an algorithm.

Flowcharting Symbols: -

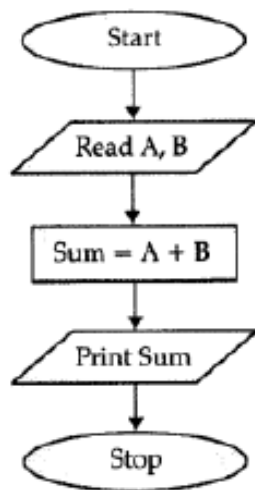
- There are 6 basic symbols commonly used in flowchart.
- Terminal, Process, input/output, Decision, Connector and Predefined Process.
- This is not a complete list of all the possible flowcharting symbols; it is the ones used most often in the structure of Assembly language programming.

Symbol	Name	Function
	Process	Indicates any type of internal operation inside the Processor or Memory
	input/output	Used for any Input / Output (I/O) operation. Indicates that the computer is to obtain data or output results
	Decision	Used to ask a question that can be answered in a binary format (Yes/No, True/False)
	Connector	Allows the flowchart to be drawn without intersecting lines or without a reverse flow.
	Predefined Process	Used to invoke a subroutine or an interrupt program.
	Terminal	Indicates the starting or ending of the program, process, or interrupt program.
	Flow Lines	Shows direction of flow.

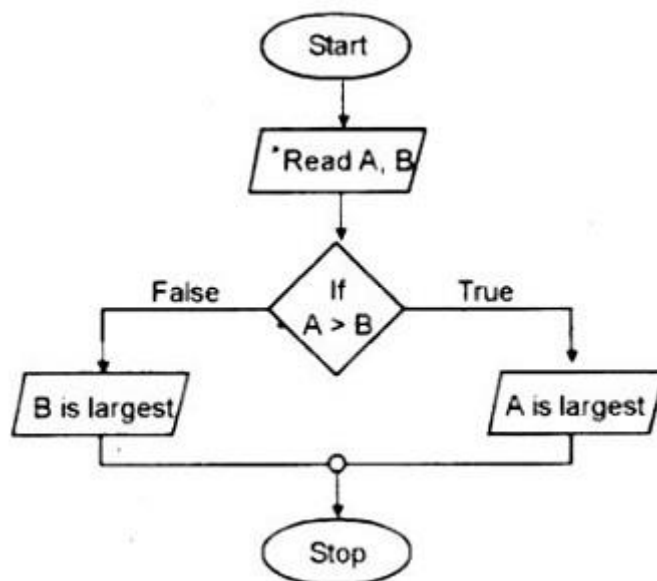
General Rules for flowcharting: -

- All boxes of the flowchart are connected with Arrows. (Not lines)
- 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.
- The Decision symbol has two exit points; these can be on the sides or the bottom and one side.
- 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.
- Connectors are used to connect breaks in the flowchart. Examples are:
 - From one page to another page.
 - From the bottom of the page to the top of the same page.
 - An upward flow of more then 3 symbols
- Subroutines and Interrupt programs have their own and independent flowcharts.
- All flow charts start with a Terminal or Predefined Process (for interrupt programs or subroutines) symbol.
- All flowcharts end with a terminal or a contentious loop.

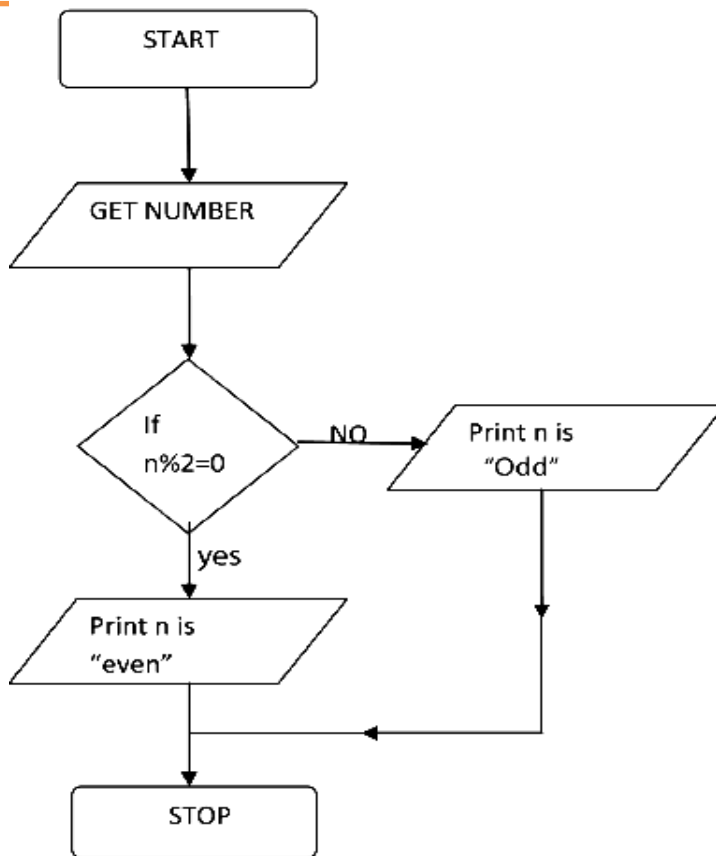
Example-1: Draw flowchart to find sum of two nos.



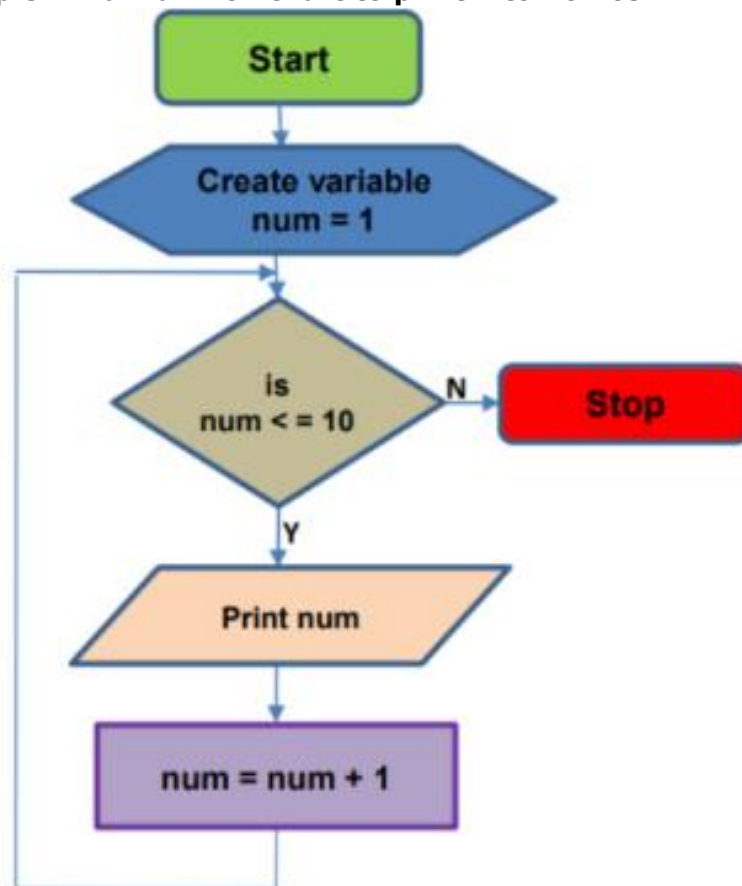
Example 2: Draw flow chart to print the larger of two numbers



Example 3: Draw a flow chart to check the given number is Odd Or Even



Example : Draw an flowchart to print 1 to 10 nos.



History of C: -

- The root of all modern language is ALGOL. ALGOL is introduced by international group in 1960. ALGOL was the first computer language to use a block structure.
- In 1967 Martin Richard developed a language called BCPL.
- In 1970 Ken Thompson created a language using BCPL and called it simply B.
- C was developed from ALGOL, BCPL and B by Dennis Ritchie at Bell Laboratory in 1972.
- C uses many concepts from these languages.
- An also use the many concept from these language and added the concept of data type and other powerful futures.
- Then after in 1978 Kernighan and Ritchie developed a new language K&R C.
- In 1989 the ANSI-C was recognize by ANSI committee.
- The in the year 1990 the ISO committee standardise the language is known as ANSI/ISO C.

Year	Language Name	Developed By
1960	ALGOL	International C
1967	BCPL	Martin Richard
1970	B	Ken Thompson
1972	Traditional C	Dennis Ritchie
1978	K & R C	Kernighan and Ritchie
1989	ANSI C	ANSI Committee
1990	ANSI/ISO C	ISO Committee

Important of C: -

- Programs are written in C are efficient and fast.
- C is efficient because C has much different type of data type and powerful operators.
- C is many times faster then other language.
- There are only 32 keyword and its strength lies in its built in functions.
- C is highly portable. This means that C programs written for one computer can be run on another with little or no modification.
- C language is block structure programming language. So user can create a block or function module.
- Another important feature of C is its ability to extend itself.
- We can add our own functions to C library.

Basic Structure of C Programs: -

- Documentation Section: - The documentation section consists of a set of comment line. In this section we can provide the name of program, date of program, author and other general information regarding to program.
- Link Section: - The link section provides instruction to the compiler to link function from the header file.
- Definition Section: - The definition section defines the symbolic constant.
- Global Declaration Section: - There are some variables that are used in more then one function. That kind of variable are called global variables. The Global variables are declared in global declaration section. That is outside of all the function. This section is also known as **user-defined** section.
- Main Function: - Every C program must have one main() function. This section is consisting of two parts. Declaration part and Executable part. In declaration part we must declared all variable which are used in a program. There is at least one executable statement in executable part. These two parts must write between the opening and closing brace.
- Subprogram Section: - The subprogram section contains all the user-defined function. The user define function are generally placed immediately after the main function.

Documentation Section
Link Section
Definition Section
Global Declaration Section
main() function section
{
Declaration Part
Executable Part
}
Subprogram Section (User Define Function)

Just Remember: -

- Every C program must have a main() function.
- Use of more than one main() is illegal.
- Execution of the program is start from main().
- The execution of function starts from opening brace and ends at closing brace.
- C programs are written in lowercase. Uppercase may be used for identifier, variable name or output string.
- Every C program statement in a C language must end with a semicolon (;).
- All variables must be declared their types before they are used in program.
- We must include header files using #include directive.

C Tokens: -

- In a C program the smallest individual units are known as a C Tokens.
- C has six types of tokens as shows as under.
 1. Keywords. Ex.- float, int, for, while, if, else, switch, struct, union, break, etc..
 2. Identifiers. Ex.- main, amount, etc..
 3. Constants. Ex.- -15.98, 30, 100, etc..
 4. String. Ex.- "ABC", "pqr", "Year", "Amount", etc..
 5. Special Symbols. Ex.- [], { }, (), etc..
 6. Operators. Ex.- +, *, -, /, <, >, =, etc..

Keywords: -

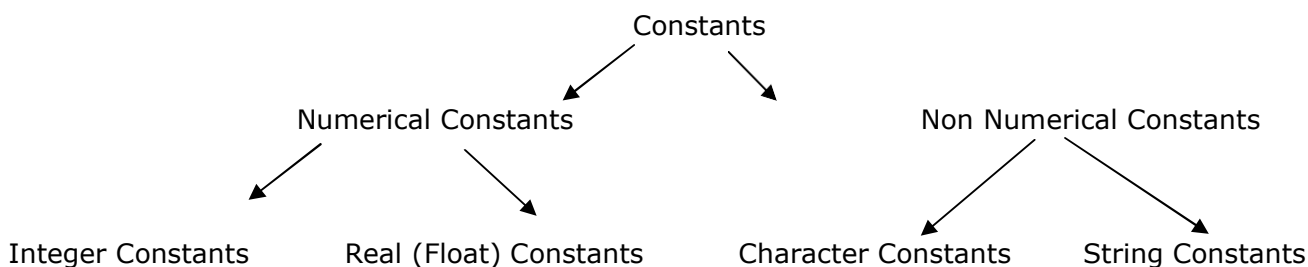
- Every C word is classified as a keyword.
- All keyword have fixed meanings.
- These meaning cannot be changed.
- All keywords must be written in lowercase.
- Ex.- auto, break, case, char, const, do, while, for, if, else, goto, int, float, double, void etc..

Identifiers: -

- Identifier refers to the mane of variables, functions and array name, structure name.
- These names are defined by user, so they are known as user-define names.
- Identifier consists of letters and digits.
- Both uppercase and lowercase are permitted.
- **Rules of Identifiers: -**
 1. First character must be an alphabet or underscore.
 2. Must consist of only letters, digits and underscore.
 3. Only 31 characters are significant.
 4. We can not use keyword as identifiers.
 5. White space are not allowed in identifiers.

Constants: -

- Constants refer as fixed value in C that does not change during the execution of a program.
- C supports several types of constants.



- Integer Constants refer to a sequence of digit like 1056.
- Real (Float) constants refer a fractional (point) part like 17.76. Such numbers are called real or floating point constants.
- A single character constant contains only a single character. They are enclosed within single quote mark. Ex. - 'g', 'K', 'l' etc...
- A string constant is sequence of character enclosed within double quote mark. Ex. - "Well Done", "Hello Word", etc...

Variables: -

- A variable is a data name that may be used to store a data value.
- Variables may take different values at different time during execution.
- A variable name can be chosen by the programmer.
- Ex.- Average, height, Total, Counter_1, class_strength etc....
- **Rules For Variable: -**
 1. They must begin with letter.
 2. Variable of 31 characters are valid.
 3. Uppercase and lowercase are significant.
 4. It should not be a keyword.
 5. White space is not allowed.
- Some valid example of variable.
John, Value, T_raise, Delhi, x1, ph_value, mark, sum2, sum_5 etc....

Data Types: -

- C language is very rich in data types.
- The varieties of data types are available in ANSI - C.
- C supports three type of data types,
 1. Primary (Fundamental or Basic) data type.
 2. Derived data type.
 3. User-define data type.
- All C compiler support five type of fundamental data types.
- Integer (int), character (char), floating point (float), double floating point (double) (long double) and null (void).
- Integers are whole numbers with the range of values and define by keyword int.
- Floating point numbers are defining in C by the keyword float, double and long double.
- A single character can be defined as a char data type.
- The void type has no values. This is usually used to specify the type of function.
- The range of data types is given below.

Data Type	Size (in Byte)	Range
char	1 byte	-127 to 127
unsigned char	1 byte	0 to 255
int	2 byte	-32,768 to 32,767
unsigned int	2 byte	0 to 65535
float	4 byte	3.4E -38 to 3.4E +38
double	8 byte	1.7E -308 to 1.7E +308
long double	10 byte	3.4E -4932 to 1.1E +4932
void	Null	Null

