

Title : <b>Basics of C programming language</b>	Duration (in hours) :15 Hours
<b>Objectives</b> <ul style="list-style-type: none"> <li>• Solve the problem using step by step approach with an algorithm.</li> <li>• Understand the significance of C language.</li> <li>• Understand keywords, identifiers, variables, and data types.</li> </ul>	
<b>Takeaways :</b> <ul style="list-style-type: none"> <li>• Student shall be able to know the usage of variables, operators and data types.</li> <li>• Student shall be able to write sequential algorithms and C programs.</li> </ul>	
<b>Significance of C language:</b> C language is the basic programming language. It is simple, reliable and easy-to-use. To be a programmer, it is advisable to begin with C language. Languages like C++, C# etc are built on the C language. <b>Applications of C</b> JVM, Python interpreter, Google search engine, Windows operating system, embedded softwares, Android OS and many system softwares are indeed written in C/C++, making C/C++ the mother of many languages today.	
<h2>ALGORITHMS</h2> <p><b>Definition:</b> Algorithm is defined as an effective step by step procedure with finite number of well-defined steps for problem solving.</p> <p><b>Characteristics of an Algorithm</b></p> <ul style="list-style-type: none"> <li>• Input</li> <li>• Definiteness</li> <li>• Effectiveness</li> <li>• Finiteness</li> <li>• Output</li> </ul> <ul style="list-style-type: none"> <li>• <b>Input:</b> It may accept zero or more inputs. These data elements are processed in the subsequent steps or instructions in the algorithm</li> <li>• <b>Definiteness:</b> Each instruction must be clear, well-defined and precise. There should not be any ambiguity. Means each and every instruction must specify clearly that, what should be done.</li> <li>• <b>Effectiveness:</b> Each instruction must be simple and be carried out in a finite amount of time. Means each operational step can at least in principle be carried out by a person using a paper and pencil in a minimum amount of time.</li> <li>• <b>Finiteness:</b> Algorithm should have finite sequence of instructions. That is, it should end after a fixed time. It should not enter into an infinite loop (It should come to an end).</li> <li>• <b>Output:</b> It should produce at least one output.</li> </ul> <p><b>Notations of an Algorithm</b></p>	

- Name of the algorithm: Specifies the problem to be solved.
- Step number - Identification tag of an instruction and it is an unsigned positive number.
- Explanatory comment within the square brackets
- Termination

### Advantages of Algorithm

- Effective communication: Since algorithm is written in English like language, logic of the problem can be explained easily to others
- Easy and efficient coding : It acts as a blue print during a program development
- Program debugging : Helps in debugging so that we can identify the logical errors in a program
- Program maintenance : maintaining the software becomes much easier

### Different patterns/types of Algorithms

1) Sequential : This pattern indicates the sequential flow of program logic (a step by step continuous flow or execution)

2) Conditional : In this pattern, different steps in an algorithm are carried out based on a condition (whether true/false) . In programming languages, a conditional pattern is implemented using decision making statements.

3) Iterative: In this pattern a task (one or more steps ) is repeated n number of times until a condition is satisfied. In programming languages, an iterative pattern is implemented using looping constructs. An iterative construct is also known as “repetitive” construct.

#### 1. Write an Algorithm to find Area of circle

Step 1: start

Step 2: Read r [ Input the value for radius]

Step 3: [Compute Area using formulas]

$\text{Area} \leftarrow 3.142 * r * r$

Step 4: Print Area [ Output the value of computed area]

Step 5: Stop [End of the Algorithm]

#### 2. Write an algorithm to find sum and average of 3 numbers

Step 1: start

Step 2: read a,b,c [ input 3 numbers]

Step 3:  $\text{sum} \leftarrow a + b + c$  [ find sum]

Step 4:  $\text{avg} \leftarrow \text{sum} / 3$  [find average]

Step 5: print sum, avg [output the result]

Step 6: stop






Disadvantages of an algorithm:

- Developing algorithm for large and complex problems would be time consuming and difficult to understand
- Understanding complex logic through algorithms would be difficult.

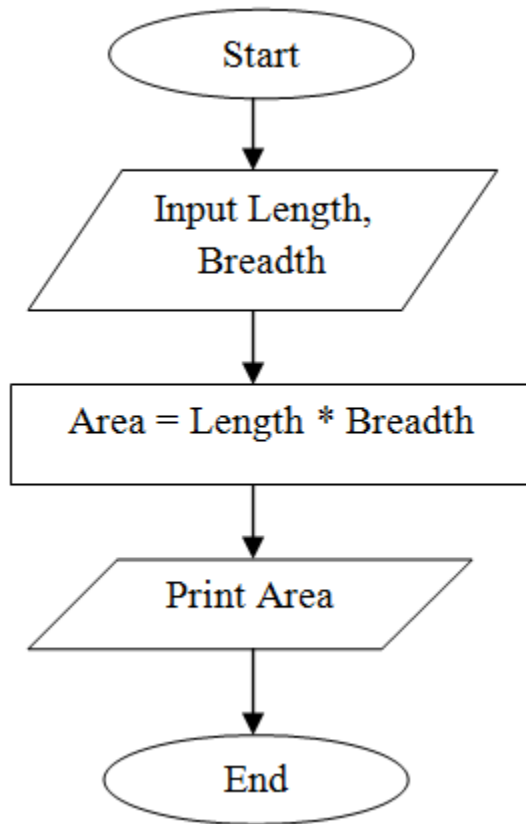
## FLOWCHART:

Definition: “A flowchart is a diagrammatic representation of a solution to a given problem.”  
In simple terms, it gives the flow of execution of different instructions in the program.

Basic symbols used to draw a flowchart :

Symbol	Name	Function
	Start/end	An oval represents a start or end point
	Arrows	A line is a connector that shows relationships between the representative shapes
	Input/Output	A parallelogram represents input or output
	Process	A rectangle represents a process
	Decision	A diamond indicates a decision

**Example : To compute area of rectangle**



#### Advantages of flowchart

- Effective Communication: Flowchart acts as a good visual aid to represent the logic for the problem solution.
- Easy and efficient coding: Writing program is easy if the flow of program is known.
- Effective Analysis: With the help of a flowchart, problem can be analyzed effectively.

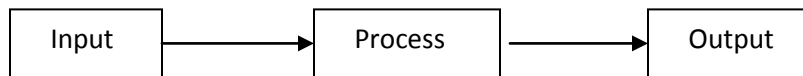
#### Disadvantages of flowchart

- Complex logic: For complicated logic, the flowchart becomes complex and clumsy.
- Alterations and modifications: If there is change in the logic, the flowchart has to be completely rewritten and requires lot of time.

## Introduction to Computer and C language

Now a days computers are used in almost all fields of our day-to-day life. It reduces the human workload. It has got huge storage capacity and high speed.

**Definition :** “Computer is an electronics device that receives data from the user, stores it in memory, processes the data and gives the desired output ”



Computer is a combination of hardware and software.

**Hardware:** “Hardware is that part of the computer system which user can touch and feel ”.

Eg: Monitor, Keyboard, Mouse, UPS, Printer, Scanner, CPU etc.

**Software:** Software is a set of programs to perform certain tasks.

Eg: Operating system, Banking application, Translators, Device drivers, Student Information Management system etc.

There are two types of software,

1.**System software:** Set of programs written by manufacturers that will control the computer system

. Eg: Operating system, Translators(Assembler, Compiler, Interpreter), Device Drivers etc

2.**Application software:** Set of programs written to perform some specific task in day-to-day life

. Eg: SIMS, Ticket Reservation application.

**Program:** Program is set of instructions to carry out a particular task.

### Logical organization of computer

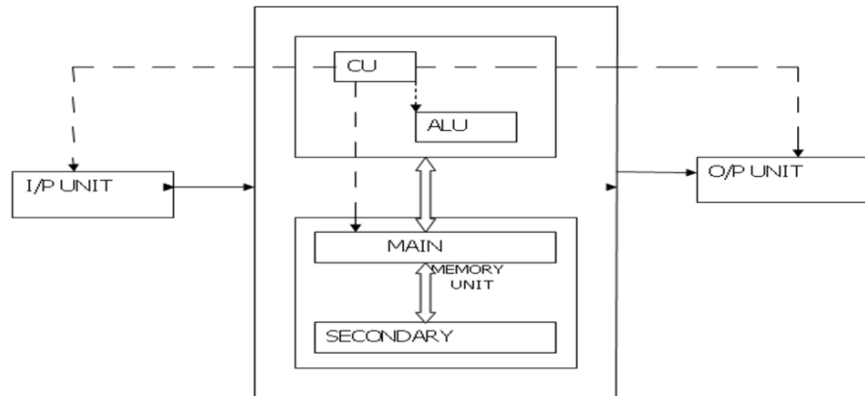
The four tasks that are carried out by computer system are:

- Accept the data.
- Process the data.
- Store the data
- Output the data. (results).

The four functional units of computer system are:

- Input Unit.
- Central Processing Unit.
- Memory Unit.
- Output Unit.

..... Dotted lines indicate the control signals issued by Control unit  
—————> represent data or instructions.



### Input Unit:

- It is an external device that is connected to the computer system
- The input unit is used to enter data or program (instructions) to the computer. The various input devices include keyboard, mouse, scanner, joystick etc.

### Output Unit:

- The output unit display the processed data (result). The output displayed on the monitor is called the softcopy output. And the output printed on the paper is called the hardcopy output.
- The various output devices include monitor, printer, speakers, plotters, LCDs, Plasma panel displays etc.
- Some devices act as both input and output devices such as Touch panel and modem(modulator and demodulator)

### Central Processing Unit:

- The CPU acts as a Brain of the computer. It organizes and processes the data and instructions received from the input source such as keyboard or file. The CPU refers to the microprocessor of the computer. People always refer their computers based on the type of CPU they contain.

CPU consists of two units namely

1. Control Unit
2. Arithmetic and Logic Unit

#### 1. Control Unit

- The control unit controls and coordinates the activities of all the units by issuing proper commands.

- Fetching data and instructions from memory.
- Interpreting the instructions.
- Controlling the transfer of data and instructions to and from memory.
- Controlling the input and output devices.
- The overall supervision of computer system

**2. Arithmetic and logic unit-** In this unit, actual operations are carried out under the supervision of control unit. The control unit issues signals to ALU to perform the following activities:

Arithmetic unit consists of an electronic circuitry which performs basic arithmetic operations +, -, \*, / and logical unit consists of logic circuitry which performs logical operations AND, OR, NOT and relational operations. The results are sent to memory unit and then to output unit.

### **Memory Unit:**

- The memory unit store data and instructions. This is also called as storage device
- The memory unit does the following.
- Stores the instructions or data.
- Stores the intermediate results.
- Memory of a computer is measured in terms of bits, bytes and words

Bits = 0 or 1

1 nibble = 4 bits

1 Byte = 8 bits

1 Kilo Byte = 1024 bytes

1 Mega Byte =  $10^6$  bytes

1 Word = Sequence of 16bits/32bits.

Memory is classified in to 2 types namely

1. Main memory
2. Secondary Memory

**Main memory-** Temporary memory, because data and instructions are lost when the computer is switched off. It is also referred to as primary memory.

- It is Internal to the computer system.
- CPU can access directly.
- Volatile memory.
- Semiconductor memory.
- Expensive.
- Less storage capacity

e.g.: ROM(non volatile), RAM (Volatile), etc.

### **RAM (Random access memory):**

It is read write memory. It is just like a page of a notebook, where you can write something to or read something from. All the programs are brought into RAM just before execution.

### **ROM(Read Only Memory) :**

It is Non volatile . It stores mainly the monitor programs and BIOS (Basic Input Output System)

Programs. The information stored in it can only be read but cannot be modified. The contents of ROM can be programmed under special conditions. It is manufacturer programmed memory.

### **Secondary Memory(Auxiliary memory)**

It is Permanent memory. It stores a large amount of information for a long time. It is also called as backup memory. Made up of magnetic material. Magnetic Tape, magnetic disk and drum are secondary devices.

- External to the computer system.
- CPU can't access directly.
- Non-volatile memory.
- Magnetic memory
- not expensive
- Large storage capacity
- e.g.: magnetic storage devices like floppy disk, hard disk, pen drive. Optical storage devices like CD, DVD etc

Differences between primary and secondary memory:

<b>Primary Memory</b>	<b>Secondary Memory</b>
Also called as Main memory.	Also called as Auxiliary memory.
Accessing the data is faster.	Accessing the data is slow.
CPU can access directly	CPU cannot access directly
Semiconductor memory.	Magnetic memory.
Expensive	Not Expensive
It is Internal memory.	It is External memory.
Data storage capacity is less	Data storage capacity is more or huge
Examples : RAM, ROM	Examples: hard disk, floppy disk, magnetic tape etc

Differences between RAM and ROM memory



<b>RAM</b>	<b>ROM</b>
Random Access Memory.	Read Only Memory.
<b>Volatile memory.</b> The contents of the RAM are lost when power is turned off	<b>Non-volatile memory.</b> The contents of the ROM are not lost when power is turned off.
<b>Temporary storage medium</b>	<b>Permanent storage medium</b>
<b>The data can be read and written.</b>	<b>The data can only be read, but the data cannot be written.</b>
<b>The programs are brought into RAM just before execution.</b>	<b>BIOS and monitor programs are stored.</b>

Differences between Hardware and software:

<b>Hardware</b>	<b>Software</b>
The physical components making up the system are termed as Hardware.	Software is a set of programs used to perform certain tasks (logical component )
The components include keyboard, floppy drive, hard disk, monitor, CPU, printer, wires, transistors circuits etc	Softwares include compilers, loaders, Banking s/w, library s/w, payroll s/w etc.
Hardware works based on instructions	Software tell the hardware what to do

Differences between System software and Application software:

<b>System software</b>	<b>Application software</b>
Collection of programs written by expert programmers /manufacturers.	Collection of programs written by users (programmers )
System software can be used to control the computer system.	Application software is written to perform particular task.
System software helps in executing other programs.	Application software are not used for executing other programs.
Examples include compilers, loaders .Operating System etc.	Examples include Banking s/w, library s/w, payroll s/w etc

**Programming Languages**

- Program: Set of instructions to solve a problem.
- Programming Language: Set of keywords and grammar for instructing a computer to perform specific task. Computer programs are written in a programming language.

**Different types of Languages**

- M/c Language: Consists of only zeros and ones, which computer can understand. It is m/c dependent.
- Assembly Language: Consists of symbols / mnemonics which constitute instructions, It is m/c dependent.
- High level Languages: it is simple and similar to natural language with some grammatical rules. It m/c independent C, C++, COBOL, Pascal, BASIC.

**Differences between Machine and Assembly Languages:**

Machine Language	Assembly language
Instructions are in the form of 0's & 1's	Instructions are in the form of Mnemonics
Writing program, debugging and understanding is difficult	Writing program, debugging and understanding is easy
Translator is not required	Translator(Assembler) is required to convert assembly language program to m/c language
Execution is faster	Execution is slower

**Difference between high level and low level**

High Level Language	Low level Language
Machine independent	Machine dependent
Uses English like instructions	Uses mnemonics

Writing program, debugging, understanding is easy	Writing program, debugging, understanding is difficult
Compiler/ Interpreter is used to convert high level language to m/c language	Assembler is used to convert Assembly language to m/c language
Execution is slower	Execution is faster

English language:

Alphabet → Word → Phrases → Sentence/Instruction

C programming language:

Character Set → Tokens → Instructions → Program

### **Character set in C:**

Character set in C includes:

Alphabets or Letters → A-Z and a-z

Digits → 0-9

Special characters → \$, ? , { , } , \ etc.

White spaces → \t, \n etc.

### **C Tokens:**

Token is the smallest individual unit in a C program.

There are 6 types of C tokens:

- Keywords.
- Identifiers.
- Constants.
- Operators.
- Strings.
- Special Symbols.

### **Keywords:**

- Keywords are basic building blocks.
- All keywords have fixed meaning and cannot be changed.
- Keywords must be written in lowercase.
- There are 32 keywords in C language.
- Keywords cannot be used as identifiers.

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

**Identifiers:**

- Identifiers refer to the names of variables, functions and arrays.
- Identifiers are user-defined names.
- It consists of a sequence of letters and digits. Both uppercase and lowercase letters are allowed.

Rules to frame identifiers:

- First character must be an alphabet or underscore.
- Must consist of only letters, digits or underscore.
- Only first 31 characters are significant.
- Keywords cannot be used as identifiers.
- Should not contain white spaces.

**Constants:**

- Constant is fixed values that do not change during program execution.
- Types of Constants:

**1. Numeric constants.****a. Integer constants.**

- An integer constant is a sequence of digits.
- Types of Integer constants:
  1. Decimal integer constants.  
Ex: 1234, -7766, +4646, 324567
  2. Octal integer constants.  
Ex: 034, 01234, 076, 0
  3. Hexadecimal constants.  
Ex: 0x34, 0X9ab, 0x6D, 0x1

**b. Floating point constant (Real constants)**

- Real constants are used to represent quantities that vary continuously like distance, height, temperature, price etc.
- Real constants are also called as Floating –point constants.
- Ex: 0.1527, -0.66, 555.89 etc.
- A real number can be expressed in exponential or scientific notation.

- General form is :

mantissa e/E exponent

where

mantissa – is either a real number or an integer.

exponent – is an integer number with optional plus or minus sign.

Ex: 215.78 is written as 2.1578e2

0.00034 is written as 3.4E-4

## 2. Character constants.

### a. Single character constants.

- A single character constant contains a single character enclosed within a pair of single quotation mark.
- Ex: '4', 'P', '!', ' ' etc.
- Character constants have integer values known as ASCII (American Standard Code for Information Interchange ) values.
- A-Z → 65-90  
a-z → 97-122  
0-9 → 48-57

### b. String character constants.

- A String Constant is a sequence of characters enclosed in double quotes.
- Characters may be letters, numbers, special characters and blank space.  
Ex: "BVB", "1899", "?...!", "H", "57-11" etc

### c. Backslash Character Constants

- These constants are combinations of two characters. Also called as Escape sequences.
- They are used in output functions.
- Ex: \n (newline), \a (audible alert), \t (horizontal tab), \0 (null) etc.

## **Operators:**

These are symbols used to perform mathematical and logical operations.

Eg: Arithmetic operators : +, -, \*, /

Relational operators : <, <=, >, >=, = etc.

**Strings** : Sequence of characters

**Special Symbols** : [ ], ( ), { }, ?, \$ etc.,

## **Variables :**

- A variable is an identifier. It is a name given to a memory location which stores a data value.
- Variable takes different values at different times during execution.  
Ex: Average, total, class\_sum, x etc.

**Declaration of Variables:****Syntax:**

data\_type v1, v2, v3, ..... vn ;

where data\_type → can be any primary data\_type.

v1, v2, v3 → names of variables.

Ex: int count ;

float num , x ;

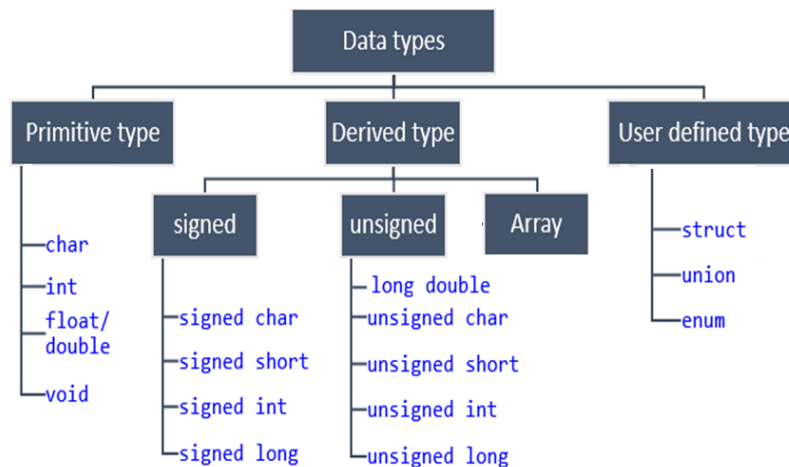
**Data Types:** C has a rich set of data types.

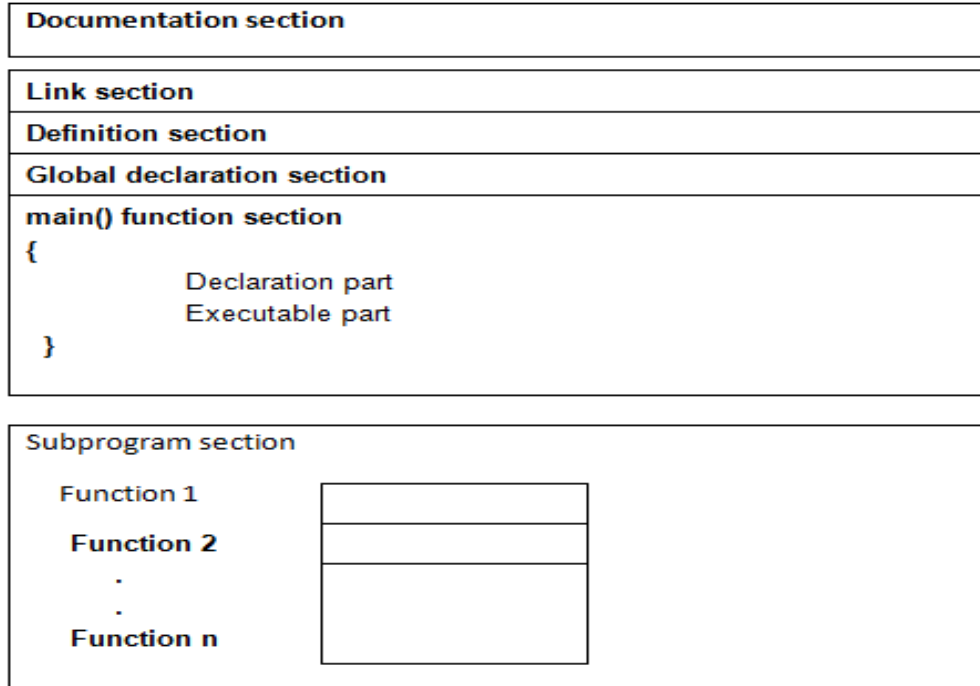
C supports three classes of data types:

- i. Primary or fundamental data type.(int, float, char, double)
- ii. Derived data type.(functions, arrays, pointers)
- iii. User-defined data type.(structure, union, enum)

Primary data types:

	Data type	Size (bytes)	format specifier
1.	char	1	%c
2.	int	2	%d
3.	float	4	%f
4.	double	8	%lf

**Structure of C-program:**



**Documentation section:** It contains the explanatory note about the program. This will increase the readability.

There are 2 types of documentation

1. External documentation
2. Internal documentation

External documentation is done outside the main() function. It contains the purpose of the program, author of the program and date of creation.

Internal documentation is done inside the main program or function. Its used to comment on the operation of particular instruction.

There are two ways of commenting.

1. Multi line comment
2. Single line comment

Syntax for multiline comment is: `/*-----  
-----*/`

For eg: `/*Program to find the area of circle`

`Author : xyz`

`Date : --/--/---- */`

Syntax for single line comment : `//-----` For eg: `sum=a+b //Adds two numbers`

**Link section:** It contains some preprocessor directives or header files. These header files contain

some predefined functions required for program execution.

For eg: `#include<stdio.h>` or `#include "stdio.h"`

Here include is a command and `stdio.h` is the header file which contains some standard input output functions. Without some header files our C program doesn't get executed.

Declaration section: This section contains declaration of user defined functions.

Global declaration: In this section we declare the variables whose visibility is through out the program. Such variables are called as global variables.

`main()` : This is the important function in the C program. It is the entry point of any C program. No other user defined function should have this name.

Body of this function has 2 parts

1. Local declaration: It contains the declaration for variables required for processing in the main function.
2. Executable statements: These are nothing but the instructions in the program.

User defined function definition: This section contains definition of user defined functions.

```
Ex: #include<stdio.h>
    main()
    {
        printf("welcome to c programming language \n");
    }
```

Example : (to demonstrate the usage )

- **Variable**

Ex: `int sum; float percentage ;char code ;`

- **Hierarchy of Operations**

Determine the hierarchy of operations and evaluate the following expression:

$i = 2 * 3 / 4 + 4 / 4 + 8 - 2 + 5 / 8$

Stepwise evaluation of this expression is shown below:

$i = 2 * 3 / 4 + 4 / 4 + 8 - 2 + 5 / 8$

$i = 6 / 4 + 4 / 4 + 8 - 2 + 5 / 8$

$i = 1 + 4 / 4 + 8 - 2 + 5 / 8$

$i = 1 + 1 + 8 - 2 + 5 / 8$

$i = 1 + 1 + 8 - 2 + 0$

$i = 2 + 8 - 2 + 0$

$i = 10 - 2 + 0$

$i = 8 + 0$

$i = 8$

operation: \*

operation: /

operation: /

operation: /

operation: +

operation: +

operation : -

operation: +

- 

Test your Understanding: (include true/false, fill in the blanks)

**[A]Which of the following are invalid variable names and why?**

BASICSALARY    \_basic    basic-hra



```
#MEAN      group .      422
FLOAT      HELLO      queue.
team'svictory  Plot # 3      2015_DDay
```

**[B] Point out the errors, if any, in the following C statements:**

- (a) `int = 314.562 * 150 ;`
- (b) `name = 'Ajay' ;`
- (c) `varchar = '3' ;`
- (d) `3.14 * r * r * h = vol_of_cyl ;`
- (e) `k = ( a * b ) ( c + ( 2.5a + b ) ( d + e ) ;`
- (f) `m_inst = rate of interest * amount in rs ;`

**[C] Pick up the correct alternative for each of the following questions:**

1. A C variable cannot start with

- 1) An alphabet                      3) A special symbol other than underscore
- 2) A number                         4) Both (2) & (3) above

2. Which of the following shows the correct hierarchy of arithmetic operators in C

- (1) `**`, `*` or `/`, `+` or `-`      (3) `**`, `/`, `*`, `+`, `-`
- (2) `**`, `*`, `/`, `+`, `-`              (4) `/` or `*`, `-` or `+`

3. The expression, `a = 30 * 1000 + 2768 ;` evaluates to

- (1) 32768                              (3) 113040
- (2) -32768                            (4) 0