Sulakshi Sewmini Samarakoon
29878

TUTORIAL: 01

1)

Programming languages are indispensable when it comes to crafting programs and developing software, as they serve as vital instruments for code composition and software creation.

2) Source code vs Machine code

• The proximity of source code to the user is closer compared to machine code, whereas machine code is in closer proximity to the machine.
• HLL languages are utilized in writing source code, whereas machine languages are employed for machine code.
• Unlike machine code, which doesn't require a language translator, source code necessitates a language translator.

High Level Language vs Low Level Language

• Binary code or mnemonics are employed in low-level languages, whereas high-level languages use English-like languages.
• Low-level languages are reliant on specific machines, while high-level languages are designed to be machine-independent.
• High-level languages are portable, whereas low-level languages typically lack portability.

Compiler vs Interpreter

The compiler performs a complete translation of the entire program, whereas the interpreter translates the program line by line in the order it was entered.

Structured language vs Object-oriented language

Structured language and object-oriented language represent distinct programming language paradigms, each offering its unique approach to code organization and manipulation.

C vs C++

C is classified as a procedural programming language, whereas C++ encompasses support for both procedural and object-oriented programming paradigms.

C++ vs Java

C++ supports both procedural and object- oriented programming paradigms, whereas Java is primarily an object-oriented programming language.

Syntax error vs Logical error

A syntax error arises from a violation of the language's grammar rules, whereas a logical error occurs when parentheses are incorrectly used or when a program produces inaccurate results during runtime.

TUTORIAL : 02

1)

• Comments in an AC program are written by utilizing "//" at the start of the comment.
• The purpose of comments in a program is to provide the programmer with an understanding of what the program signifies.

2)

The main function holds the utmost significance in a C program.

3)

Purpose of 'scanf' is to get user inputs.

4)

Yes. 'standard c' is a case sensitive language.

5)

• Valid :- record1, $tax, name, name_and_address

• Invalid :- 1record, file-3, return, name and address, name-andaddress, 123-45-6789 o 1record is invalid because c identifiers can't start with a number. o File-3 is invalid because in identifiers hyphens or minus signs are not allowed return is invalid because return is reserved keyword in c. o Name and address is invalid because c doesn't allow spaces in identifiers. o Name-and-address is because in c identifiers hyphens or signs are not allowed. o 123-45-6789 is invalid because in c identifiers hyphens or minus signs are not allowed.

6)

a. Function printf always begins printing at the beginning of a new line. – false

b. Comments cause the computer to print the text enclosed between /* and */ on the screen when the program is executed. – false

c. The escape sequence \n when used in a printf format control string causes the cursor to position to the beginning of the next line on the screen. – true

d. All variables must be defined before they're used. - true

e. All variables must be given a type when they're defined. – true

f. C considers the variables, number and NuMbEr to be identical. – false

g. A program that prints three lines of output must contain 3 printf statements. – false

7)

```
*
**
***
****
*****
```

8)

a. Scanf("d", value); - there is no % before d and & before value

b. printf("The product of %d and %d is %d"\n, x, y);- There are 3 %d symbols but only 2 variables and the \n is written outside the "" symbol

c. scanf("%d", anInteger);- there is no & before anInteger

d. printf("Remainder of %d divided by %d is \n", x, y, x %y);- There are 3 variables but there are only 2%d symbols.

e. Print(“The sum is %d\n,” x+y);- the comma is inside the “” & print is a wrong command correct command is “printf”

f. Printf(“The value you entered is : %d\n,&value);- there is no ending symbol of “” symbol and there is an extra & Infront of value

9)

g. printf(“%d”, x); - 2
h. printf(“%d”, x + x); - 4
i. printf(“x=); - x=
j. printf(“x=%d”, x); - x=2
k. printf(“%d = %d”, x + y, y + x); -5 = 5
l. z=x+y; - Nothing
m.scanf(“%d%d”, &x , &y);- Nothing
n. /*printf(“x + y = %d”, x + y); */- x+y=5
o. Printf(“\n”);- A line break will print

10)

n) C operators are evaluated from left to right. –**false**

➢ It depends on the precedence of the operators

o) The following are all valid variable names: _under_bar_, m928134, t5, j7,her_sales, his_account_total, a,b,c,z,z2. – **true**

p) The statement printf(“a=5”,); is a typical example of an assignment statement. – **false**

➢ Because a=5 is only the typical example of assignment statement not printf(“a=5”);

q) A valid arithmetic expression containing no parentheses is evaluated from left to right. – **false**

➢ It depends on the precedence of the operators.

r) The following are all invalid variable name: 3g, 87, 67h2, h22, 2h –**false**

➢ Because h22 is a valid variable name.