TUTORIAL 01

01. Programming languages are instructions for creating software applications for various tasks, including automating tasks, creating interactive websites, and developing sophisticated ones. They are essential for solving real-world problems, providing secure, reliable, and easy-to-maintain software with built-in features, error handling, debugging tools, and version control systems.

02.a)

Source Code	Machine Code
It is easy to read and understanding by human.	It is not easy to read and understating by human.
Source code needs to be. translated before executed (Byusing compiler and interpreter)	Machine code is executed. directly by computer's hardwarewithout any translation.
It is easy to write complex program.	It is difficult to write complexprogram.

b)

High-Level Language	Low-Level Language
It is easy to read and write forhumans. They have keyword, logical constructs, and meaningful variable name.	It is not easy to read and write for humans. They have symbols,binary code, and mnemonic instructions.

High level language provides many libraries, frameworks, andtools for developers.	Low level language is typically specific to a particular hardwarearchitecture.
A program written using high-level language will run on anydevice. It is not based on one device.	A program written using low- level language will not run onany device. It is based on one device.

c)

Compiler	**Interpreter**
First analyze the entire sourcecode. Then it converts into machine code.	It is executing the source code line by line.
Generally, faster, and more efficient.	Generally, slower, and less efficient.
Errors and issues are often detected during runtime.	Errors and issues are often detected during interpretation process.

d)

Structured Language	Object Orientated Language
It is using control structure likesequences, selection and iteration to execute program.	It is providing mechanisms like classes and objects to structureprograms.
It is using simple data types likeintegers, floats, and characters.	It is using complex data. structure like properties and functions.
It is promoting code reusabilitythrough procedural abstraction.	It is promoting code reusabilitythrough class inheritance and object composition.

e)

C	C++
C is primarily a procedural programming language. It is focuses on structured programming.	C++ supports multiple programming paradigms. It is focuses on Object Oriented Programming (OOP), procedural programming and generic programming.
C has a small standard librarythat provides essential functions.	C++ has an extensive standard library that includes the functionally of C's standard library.
C is widely used in system programming, embedded systems…	C++ widely used in applications, game developments, GUI applications…

f)

C++	Java
C++ supports multiple programming paradigms. It is focuses on Object Oriented Programming (OOP), procedural programming.	Java is primarily an Object-Oriented Programming language.
C++ allows manual memory management with features like pointers.	Java has automatic memory management trough garbage collection.
C++ code is compiled into machine-specific binaries, making it less platform-independent.	Java code is compiled into bytecode that runs on the Java Virtual Machine (JVM).

g)

Syntax Error	Logical Error
Syntax errors are detected by thecompiler or interpreter during the compilation or interpretation process.	Logical errors are do not triggerany error messages during compilation or runtime.
Syntax errors result from mistakes in the code's structuresuch as missing semicolon, brackets.	Logical error result from incorrect logic. For that case results will be incorrect.
It is providing information aboutthe specific line and location.	It is not providing informationabout the specific line and location.

TUTORIAL 02

1. In the C language, comments are written using "//" for single-line comments or "/* ... */" for multi-line comments. They provide explanations, documentation, disable code, and serve as reminders, enhancing code readability and understanding.

2. Main function

3. Input a value

4. Q4. Yes, C is a case sensitive language

5.

   (a) record1 :- valid identifiers

   (b) 1record :- invalid (a variable name should not start with a number.)

   (c) file-3 :- invalid (a hyphen should not be in the middle.)

   (d) return :- valid identifiers

   (e) $tax:- invalid (a variable name should not start with a symbol.)

   (f) name :- valid identifiers

   (g) name and address :- invalid (spaces should not in the middle.)

   (h) name-and-address :- invalid (hyphen should not be in the middle.)

   (i) name_and_address :- valid identifiers

   (j) 123 - 45 - 6789 :- invalid (a hyphen should not be in the middle and a variable name should not start with a number)

6. c. False, it is representing a new line

d. True

e. True

f. False, C is a case sensitive language. Therefor number and Number are different variable.

7.

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

8.

a) scanf("%d", value);
b) printf("The product of %d and %d is %d"\n", x, y);
c) Scanf("%d", &anInteger);
d) printf("Remainder of %d divided by %d is %d\n", x, y, x % y);

e) printf("The sum is %d\n," x + y);
f) Printf("The value you entered is: %d\n", &value);

9.

a. 2
b. 4
c. X=
d. 5=5
e. Nothing
f. Nothing
g. Nothing
h. Nothing
i. Nothing

10.

a. True
b. True
c. False
    - printf function is only print lines. It is not an assignment statement.
d. False
    - When executing a program, arithmetic expressions executing based on the operator precedence and associativity.
e. False
    - There's variable start with a letter.

TUTORIAL 03

1. X = x + 1;
   X += 1;
   X++;
   ++x;

2. A. z = (x++) + y;
   b. product *= 2;
   c. product = product * 2;
   d. if (count > 10) printf("Count is greater than 10.\n");
   e. total -= --x;
   f. total += x--;
   g. q = q % divisor;
      q %= divisor;
   h. printf("%.2f", 123.4567);
   i. printf("%.3f", 3.14159);

3. A. scanf("%d", &x);
   b. scanf("%d", &y);
   c. int i = 1;
   d. int power = 1;
   e. power *= x;
   f. i++;
   g. while (i <= y) {
      // Code block to execute while the condition is true
      // ...
      }
   h. printf("%d", power);

TUTORIAL 04

1.

- The logical OR operator in c is represented by ||, not | this one .
- Comparison operator: When comparing the value of numNeighbors with 4, you should use the equality operator ==, not the assignment operator =.
- The code block inside the if statement should be properly indented to indicate it .

2.

- The reason is that the first 'if 'statement checks whether 'number' is greater than 0.
- Since number has the value 4, which is indeed great than the 0.
- The inner 'if' statement is executed.
- The alpha has the value -1.0, which is not greater than 0.

3.

- All possible outcome
    - If 'doesSignificanWork' is true and 'makesBreakthrough' is true , 'nobelPrizecandidata' is set to 'true'.
    - If 'doesSignificanWork' is true and 'makesBreakthrough' is true , 'nobelPrizecandidata' is set to 'false'.
    - If 'doesSignificanWork' is false, and 'makesBreakthrough' is true , 'nobelPrizecandidata' is set to 'false'.

4.

1. If (taxcode == ‘T’) {
   Price += (taxrate /100) * price ;

   }

2. if (opCode == 1) {
   double X, Y;

   scanf("%lf %lf", &X, &Y);

   double sum = X + Y;

   printf("Sum: %lf\n", sum);

   }.

3. if (currentNumber % 2 != 0) {
   currentNumber = (3 * currentNumber) + 1;

   } else {

   currentNumber = currentNumber / 2;

   }

4. if (year % 4 == 0 && (year % 100 != 0 || year % 400 == 0)) {
   leapYear = true;

   } else {

   leapYear = false;

   }

5.
```
if (distance <= 100) {
    cost = 5.00;
} else if (distance > 100 && distance <= 500) {
    cost = 8.00;
} else if (distance > 500 && distance < 1000) {
    cost = 10.00;

} else {

    cost = 12.00;

}
```

TUTORIAL 05

Switch case

```
#include <stdio.h>

int main () {

   float num1,

   num2;int

   choice;

  printf("Enter two numbers:  ");
  scanf("%f %f", &num1, &num2);

   printf("1. +\n");

   printf("2. -\n");

   printf("3. *\n");

   printf("4. /\n");

   printf("Enter your

   choice: ");scanf("%d",

   &choice);
```

```
    switch (choice) {
        case 1:
            printf("%.2f + %.2f = %.2f", num1, num2, num1 + num2);
            break;

        case 2:
            printf("%.2f - %.2f = %.2f", num1, num2, num1 - num2);
            break;

        case 3:
            printf("%.2f * %.2f = %.2f", num1, num2, num1 * num2);
            break;

        case 4:
            if (num2 != 0) {
                printf("%.2f / %.2f = %.2f\n", num1, num2, num1 / num2);
            }
            else {
                printf("Can't divide by zero");
            }
            break;
    }

    return 0;
}
```

While loop

- #include <stdio.h>

```
int main () {

   int num, count = 0;
   int even_count = 0, odd_count = 0;

   printf("Enter 10 numbers\n");

   while (count < 10) {
      printf ("Enter %d number: ", count + 1);
      scanf ("%d", &num);

      if (num % 2 == 0) {
         even_count++;
      }
      else {
         odd_count++;
      }

      count++;
   }

   printf ("Even count: %d\n", even_count);
   printf ("Odd count: %d\n", odd_count);

   return 0;
}
```

- ```
  #include <stdio.h>

  int main () {

     int num;
     int even_count = 0, odd_count = 0;

     printf("Enter a series of numbers (Terminate with -99)\n");

     while (1) {
        printf ("Enter number: ");
        scanf ("%d", &num);

        if (num == -99) {
           break;
        }

        if (num % 2 == 0) {
           even_count++;
        }
        else {
           odd_count++;
        }

     }

     printf ("Even count: %d\n", even_count);
     printf ("Odd count: %d\n", odd_count);

     return 0;
  }
  ```
  ```

Do while loop

- #include <stdio.h>

```
int main () {

   int num, i = 0;
   int even_count = 0, odd_count = 0;

   do {
      printf ("Enter %d number: ", i + 1);
      scanf ("%d", &num);

      if (num % 2 == 0) {
         even_count++;
      }
      else {
         odd_count++;
      }

      i++;

   } while (i < 10);

   printf ("Even count: %d\n", even_count);
   printf ("Odd count: %d\n", odd_count);

   return 0;
}
```

- ```
  #include <stdio.h>

  int main () {

     int num;
     int even_count = 0, odd_count = 0;

     printf("Enter a series of numbers (Terminate with -99)\n");


     do {
        printf ("Enter numbers: ");
        scanf ("%d", &num);

        if (num != -99) {
           if (num % 2 == 0) {
              even_count++;
           }
           else {
              odd_count++;
           }
        }
        else {
           break;
        }

     } while (num != -99);

     printf ("Even count: %d\n", even_count);
     printf ("Odd count: %d\n", odd_count);

     return 0;
  }
  ```
  ```

For loop

- ```
  #include <stdio.h>

  int main () {

     int sum = 0;
     float avg;

     for (int i = 1; i <= 10; i++) {
        printf("Enter %d number: ", i);
        scanf("%d", &i);

        sum += i;
     }

     avg = sum / 10;

     printf("Average: %.2f", avg);

     return 0;
  }
  ```

- ```
  #include <stdio.h>

  int main () {

     int rows;

     printf("Enter rows number: ");
     scanf("%d", &rows);


  for (int i = 1; i <= rows; ++i) {

        for (int j = 1; j <= i; ++j) {
  ```

```
            printf("*");
        }
        printf("\n");
    }

    return 0;
}
```