**Tutorial 1**

1.programming language are essential tools in the computer science and software development. The need of programming languages are communication with computers, abstraction and efficiency, problem solving and automation, software development and maintenance etc.

2.soure code vs. machine code

Source code is written by humans using a programming language and is human readable. Machine code, on the other hand, is a low- level language directly understood by computers and is not easily readable by humans.

High level language and low- level language

High level language: Abstraction, easier to use, portable, closer to natural language, slower execution, platform independent.

Low level language: direct hardware interaction, faster execution, complex, less portable, difficult to use.

Compiler vs. Interpreter

Compiler: translate the entire program into machine code before execution.

Interpreter: executes the program directly, translating and executing at line at a time.

Structured language vs. object oriented language

Structured language focuses on sequential execution and uses function or procedures, while object oriented language emphasizes objects and encapsulation of data and behavior.

C vs. C++

C:prodedural, minimalistic, lower level, less features, manual memory management.

C++: object oriented, higher level, automatic memory management.

C++ vs. Java

C++ offers low level control, high performance, and memory management, while java provides portability, simplicity.

Syntax error vs. Logical error

A syntax error occurs due to incorrect programming syntax, while a logical error results in incorrect program logic.

**Tutorial 2**

1. We can write using two forward slashes (//) for single line comments or /\*…\*/ for multi- line comments. Purpose are providing explanation, documenting code, and making it more understandable for developers.

2. main(), printf(), scanf()

3. read and store input values.

4. Yes

5. a) record1 – it starts with a letter and can be followed by letters , numbers, or underscores.

f) name – it consists of only letters.

i) name\_and\_address – it consists of letters, numbers and underscores.

d) return – this is valid, although it is a reserved keyword in many programming languages.

6. a) False – the function printf doesn’t automatically begin printing at the beginning of a new line. It continues printing from the current position.

b) false – comments are ignored by the compiler and do not cause any text to be printed on the screen during program execution.

c)True

d) False – variables can be defined at any point before they are used.

e) false – in some programming languages, variables can be implicitly typed based on their usage or context.

f) false – variable names are case-sensitive, so number and NuMbEr would be considered as 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) no error

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) x=2

e) 5=5

f) nothing

g) nothing

h) nothing

i) a blank line

10. a) true

b) true

c) false – the statement is a print statement not an assignment statement. It prints the text “a=5;” to the console.

d) true

e) true

**Tutorial 3**

1. Using a increment operator (++)

x++;

using the addition assignment operator

X += 1;

Using the pre increment operator

++x;

Using the regular addition operator

X = x+1;

2. a) z=x+y++;x++;

b) product \*= 2;

c) product = product \* 2;

d) if(count>10) printf(“count is greater than 10”);

e) total -= (--x);

f) total += x--; 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)

h) printf(“%d”,power);

**Tutorial 4**

1. If numNeighbors >= 3 or numNeighbors == 4;

numNeighbors += 1

printf(“You are dead!\n)

else:

numNeighbors -= 1

2. if the variable makesBreakthrough is true, it sets the variable

nobelPrizeCandidate to true.

If the variable makesBreakthrough is false, it sets the variable

nobelPrizeCandidate to false