## Q1. What are the key differences between Procedural Programming and Object-Oriented Programming (OOP)?

## ANS:-

| Procedural oriented language | Object oriented programming |
|---|---|
| In pop , program is divided into small parts called function. | In oop , program is divided into parts called object. |
| Pop follows top down approach. | Oop follows bottom up approach. |
| We cannot declare namespace directly. | We can declare namespace directly. |
| Concepts like inheritance , polymorphism , data encapsulation , abstraction not available. | Concepts like inheritance , polymorphism , data encapsulation , abstraction available. |
| Access Specifiers (public , private , protected )not available | Access Specifiers (public , private , protected ) available. |
| Data hiding is not available. | Pop use to encapsulation for data hiding. |
| Pop less flexible for changes than oop. | Oop more flexible for changes than pop. |
| Pop is structure oriented. | Oop is object oriented. |

## Q2. List and explain the main advantages of OOP over POP.

## ANS :-

**Advantages of oop:-**

1) code reusability :- oop allows to developers to reuse block of code , pop not support.

2) maintainability :- oop very update version easily big program solve , pop not take big problem.

3) security :- data handling is possible to encapsulation in oop , pop does not support encapsulation.

4)time cost : oop saving development time and cost by using existing module.

5) Reusability :oop reuse code through inheritance , pop not support inheritance.

6) easy to understand :- pop and oop both easy to understand.

## Q3. Explain the steps involved in setting up a C++ development environment.

## ANS:

Step 1 : go to your google and search download dev C++(version 5.11.0) or latest version .

Step 2 : Download opera in your system and click to Download button for Dev C++.

Step 3 : After download you install Dev C++ in your windows , linux .

Step 4 : Dev C++ complete install in your system to set your fonts , theme and size etc.

Step 5 : after installation open Dev c++ and create a new file and write your code.

Step 6 : you save your file any place and save the extension in .cpp and run program.

## Q4. What are the main input/output operations in C++? Provide examples.

## ANS:-

C++ comes with libraries that provide us with many ways for programming input and output.

**1)input operation :-**

- if the direction of flow of bytes is from the device called input.
- In c++ cin (standard input stream):-

- Object for input statement.
- Cin operator used to (>>) right shift operator to taking value from the user.

**2)output operation :-**

- if the direction of flow of bytes opposite called output.
- In c++ cout (standard output stream):-
- Object for output statement.
- Cout operator used to (<<) left shift operator to taking output from the code and print them.

Example:-

```
#include<iostream>

using namespace std;

main()

{

        int num;

        cout<<"Enter Any Number =  ";          //output operator

        cin>>num;                              //input operator

}
```

**TOPIC :- (2. Variables, Data Types, and Operators)**

## Q1. What are the different data types available in C++? Explain with examples.

- **Data type:**

the data type of any variable tells that what kind of values will be stored inside the variable.

In simple words data type means store a data in variable.

**Data type is two type :**

1.primitive datatype.

2.nonprimitive datatype.

**We can divide data types into three parts**

1. Primary data type

2. Derived data type

3. User defined data type

- **Primitive datatype :**

primitive datatype is provide ianguage.

Ex :int , float , char etc…

**Integer :** this data type is used to store a integer value and size of bytes is 2 bytes.

**Float :** this data type is used to store decimal value and size of bytes is 4 bytes.

**Char :**This data type is used to store a single character .

use to store the character in code and use to ‘’(single quotes).

**Long integer :** declare store the positive value high range.size of 4 bytes.

**double :** declare store the decimal value high range.size of 8 bytes.

- **Non primitive datatype :**

nonprimitive datatype is provide developer.

Ex : string , array , structure , function etc…

**String :** string means collection of elements. Use to store multiple character .

**Array :**An array is a fixed-size sequenced collection of elements of the same data type.

**Function:**A Group of statements or code combined in one block for some special purpose.

## Q2. Explain the difference between implicit and explicit type conversion in C++.

## ANS :

### 1) Implicit type conversion :-

Implicit type conversion automatically convert data from one type to another.

Example :-

```
int a=12 , c;

float b=12.5 ;

c=a+b;

cout<<"\n\n\t sum of a+b = "<<c;
```

### 2) explicit type conversion :-

Explicit type conversion requires manually by the programmer.

Example :-

```
int x=12 ;

float y=12.5,c ;

c=(float)x+y;

cout<<"\n\n\t sum of x+y = "<<c;
```

## Q3. What are the different types of operatorsin C++? Provide examples of each.

**ANS :** In C++ programming operator means to perform some operations on the data or values.

Many types of operator in C++ language :

1. arithmetic
2. relational
3. logical
4. assignment
5. increment/decrement
6. bitwise
7. conditional operators

**1. Arithmetic operator :** The arithmetic operator are used perform the some operations on the value. Arithmetic operator mainly used to mathematical or logical operation.

| Operators | meaning |
|---|---|
| + | Addition |
| - | substraction |
| * | multiplication |
| / | division |
| % | moduls |

```
#include<iostream>
Using namespace std;
Main()
{
Int num1=20 , num2=10;
cout<<"\n\n\t Addition operator :- "<<num1+num2;
cout<<"\n\n\t Sub-straction operator :- "<<num1-num2;
cout<<"\n\n\t Multiplication operator :- "<<num1*num2;
cout<<"\n\n\t division operator :- "<<num1/num2;
cout<<"\n\n\t Module operator :- "<<num1%num2;
}
```

**2.Relational operator :** Relational operator is also known as a comparison operator. Relational operator used to comparison two values and make the relation between.

| Operators | meaning |
|---|---|
| > | Greater than |
| < | Less than |
| >= | Greater than & Equal to |
| <= | Less than & Equal to |
| == | Equal to |
| != | Not equal to |

```
cout<<"\n\n\t equal to :- "<<(num1==num2);

cout<<"\n\n\t not equal to :- "<<(num1!=num2);

cout<<"\n\n\t less than :- "<<(num1<num2);

cout<<"\n\n\t greater than :- "<<(num1>num2);

cout<<"\n\n\t less than equal to :- "<<(num1<=num2);

cout<<"\n\n\t greater than equal to :- "<<(num1>=num2);
```

**3.logical operator :**Logical operators are used to test more than one condition and make decisions

| operator | meaning |
|---|---|
| && | Logical AND |
| \|\| | Logical OR |
| ! | Logical NOT |

&& - And (All the expressions must be true)
|| - Or (One of the any condition must be true)
! - Not (true expression will turn into false)

**4.Assignment operator :** The assignement operator is used to assign the value to the variable.

| | | |
|---|---|---|
| += | a=a+b | a+=b |
| -= | a=a-b | a-=b |
| *= | a=a*b | a+=b |
| /= | a=a/b | a+=b |
| %= | a=a%b | a+=b |

## 5.Increment / Decrement operator :

| operator | meaning |
|---|---|
| ++a | Prefix increment |
| a++ | Postfix increment |
| --a | Prefix increment |
| a-- | Postfix increment |

Prefix : ++i, --i
Postfix :i++, i—

unary op. - a++, b-- (1 operand, 1 operator)
binary op. - a+b (2 operands, 1 operator)

## Q4. Explain the purpose and use of constants and literals in C++.

## ANS :

**Purpose of Constants :**

- Constants is a part of tokens it refers to the fixed value.
- A constants must be initialized when created , and new values cannot be assigned.

**Use of constats :-**

- ‘const’ keyword is used to declare constant variable of any type.
- We cannot change its value during execution of program.
- Syntax: constDataTypeVariable_Name=value;
- Ex: constint a=2;
- Now ‘a’ is a integer type constant.

**Two ways to use constants :-**

1) used to const keyword.
2) used to #define preprocessor

**Literals**

- Literals also known as the constant .
- Literals are data used for representing fixed values.
- Constint a = 10;
- Literals used to a constant value that is assigned to the variable.

**Many types of literals :**

1) integer literals

2) floating point literals

3) string literals

**TOPIC :- (3. Control Flow Statements)**

## Q1. What are conditional statements in C++? Explain the if-else and switch statements.

## ANS :

Conditional statement are allow developers to control the flow of program based on specific conditions.

**1. If stetment :**The condition is true code to be executed.

**Syntax :**

```
if(condition)
{
        //code to b execute
}
```

**2.if_else stetment :** IF statement condition is false so else statement condition is true.

**Syntax :**

```
if(condition)
        {
        //code to be execute
        }
        Else
        {
        // code to be execute
        }
```

**3.nested if stetment :** check if condition within if condition (if into if) .

Used to check two condtion also check inside condition.

**4.switchstetment :** multiple choice value and one choice use switch stetment.

Not use relation operator , switch use int , charcter datatype , switch use keyword switch , break , case , default.

Use with switch statement to select one of many code blocks to be executed.

**Syntax :**

```
switch(choice)
{
        Case 1:  // stetment
        Break;
        Case 2 : //stetment
        Break;
        Default : //stetment
```

```
        Break;

}
```

## Q2. What is the difference between for, while and do-while loops in C++?

**ANS :** in C++ language loops is repeat the same code a number of times.
in c language two types of loops :
**1)entry loop**
**2)exit loop**

**1)entry loop :**1)while loop
2)for loop
**2)exit loop :**1)do..while loop

**1)while loop :** A while loop is the an entry controlled loop. in while loop given condition is true then the loop is executed.and given condition false the loop is not executed.

**Syntax of while loop :**

```
intialization
While(condition)
{
        //block of code
                Increment / decrement
}
```

**Example :**

```
Inti = 1;
While(i<=10)
{
Cout<<"\n"<<i
i++;
}
```

**2) for loop :**for loop is easier to compare than while loop. A for loop is the an entry controlled loop. in for loop given condition is true then the loop is executed and given condition false the loop is not executed.

**Syntax :**

```
For(initialization ; condition ; increment/decrement)
{
        //block of code;
}
```

**Example:**

```
For(inti=1;i<=10;i++)
{
Cout<<"\n"<<I;
}
```

**3)do-while loop :** do-while loop is a exit control loop . in do – while loop given condition is true d0-while loop first time execute and after loop is repeat given condition is false loop is not execute.

**Syntax :**

```
initialization
do
{
        //block of code
        Increment/decrement

}while(condition)
```

## Q3. How are break and continue statements used in loops? Provide examples.

## ANS:

**The break statement :**

- The break statement is mainly used in the switch statements .it is also useful for immediately stopping a loop.
- IN simple words break statement is used to break the code and rest of the code will not be executed.

**Example of break statement:**

```
#include<iostream>
Using namespace std;
main()
{
```

```
for(inti=5;i>0;i--)
{
        if(i==3)
        break;
        cout<<"\n"<<i;
}
}
```

O/P : 5 4

**2) The continue statement :**

- When you skip the current iteration but remain in the loop you should use the continue statement .
- IN simple words you skip the current iteration and rest of the code will not be executed then.

**Example of Continue statement :**

```
#include<iostream>
Using namespace std;
main()
{
for(inti=5;i>0;i--)
{
        if(i==3)
        continue;
        cout<<"\n"<<i;
}
}
```

O/P : 5 4 2 1

## Q4. Explain nested controlstructures with an example.

**ANS:- nested control structure**: nested control structure refer the control structure within another control structure.

## Syntex :

```
if(num > 0)

{
```

```
        If(num % 2 == 0)
        {
        cout << num << " is a positive even number." ;
        }
        else
        {
         cout << num << " is a positive odd number.";
        }
else
{
cout << num << " is zero."
}
```

**TOPIC :- (4. Functions and Scope)**

## Q1 . What is a function in C++? Explain the concept of function declaration, definition, and calling.

## ANS :

Function is a block of code that perfrom specific task and can be called form other functions or the main progrm.

A function is a set of statements or group of block of code that performs a specific task.

- Every C++ Program has at least one function , which is main().

- In function you have not return any value from the Function to use void () function.

**IN c language Two types of function :**
1) In built function
2) User Defined Function

**1) In built function :** In built function is provided by system means already all rules and function defined by in c compiler.
**2) User Defined Function :** User Defined function is used to Reusability of the code.

- User defined functions , the user give any name to the functions except the name of key words.

- **Types of User Defined Function :**

1) function without argument without return value
2) Function with argument without return value
3) function without argument with return value
4) function with argument with return value

- **User defined functions mainly three parts follows :**

1) function Declaration (After Header file)
2) Function calling (in main function)
3) Function Definition (After main Function or Function body)

**Example of Function :**

```
#include<iostream>

using namespace std;

void aritmatic(int , int );

void arithmatic(int num1 , int num2)

{

        cout<<"\n\n\t sum of = "<<num1+num2;
```

```
}
main()
{
        int num1,num2 , choice;
        cout<<"\n\n\t Enter Number One :- ";
        cin>>num1;
        cout<<"\n\n\t Enter Number Second :- ";
        cin>>num2;
        arithmatic(num1 , num2 );
}
```

## Q2 . What is the scope of variables in C++? Differentiate between local and global scope.

**ANS :**Scope of the variable is the block of code In the entire program where the variable is declared , used and can be modified.

**Two types of variable :**

1)local variable

2)global variable

**local variable:**

- local variables are declared inside the function.
- Local variables are accessed only by the function they are declared in.
- Local variables are alive within the function they are declared.

**global variable :**

- Global variables are declared in outside the main function.

- Global variables are accessed by all functions in the program.
- Global variables declared to the with function throughout the program.
  Example :

```
#include<iostream>
using namespace std;
// gloabal variable declaration
float a=20.5;
main()
{
// local variable declaration
        int a=10;
        int A=30;
        cout<<"\n\n\t this is my local variable :- "<<a;
        cout<<"\n\n\t this is my local variable :- "<<A;
        cout<<"\n\n\t this is my global variable :- "<<::a;
}
```

## Q3. Explain recursion in C++ with an example.

## ANS :

Recursion is a type of function , is a programming technique that making a function call itself.

**Syntax :** Return_typerecursive_func{}

**Example:**

```
#include<iostream>

using namespace std;

int factorial(int);

main()

{
```

```
        int n;
        cout<<"\n\n\t Input any number :- ";
        cin>>n;
        cout<<"\n\n\t The factorial of "<<n<<" is  = "<<factorial(n);
}
int factorial(int n)
{
        if(n<=1)
        {
                return 1;
        }
        else
        {
                return n  * factorial(n-1);
        }
}
```

## Q4.What are function prototypes in C++? Why are they used?

## ANS :

- All the function In C++ need to be prototyped.
- A function prototype in C++ informs the compiler about the intention to use a function , along with its return type , parameters , and name.
- Function prototype are usually written at the beginning of the program, ahead of any programmer-defined functions(including main()).

**type of function prototype :**

1. Function declaration
2. Function definition
3. Function calling
4. Function main

**Why function prototype are used :-**

The function prototypes are used to inform the compiler about the number of arguments , the needed datatypes of a function parameter and the functions return type.

**TOPIC :- (5. Arrays and Strings)**

## Q1. What are arrays in C++? Explain the difference between single-dimensional and multi- dimensional arrays.

## ANS :-

An array is a Collection of elements or values with similar data types.

- syntax : data_type array_name[size of array];
- Each elements refers to the identification number called index number.
- Always array index will be started from "0".
- The array is the simplest data structure where each data element can be randomly accessed by using its index number.
- Mainly three Types of Arrays :

1) One Dimentional Array e.g int arr[5];
2) Two Dimentional Array e.g int arr[3][3];
3) Multi Dimentional Array e. g int arr[4][3][3];

**Differentiate between one-dimensional and multi-dimensional arrays:**

**One dimensional :**it has only one dimensional. Array store single line multiple element. One dimensional array isexecuteseries type.

**Multi dimensional**: it has three or multiple dimensional. Multi dimensional array store row and column multiple element. And multi dimensional array is execute table or matrix.

## Q2. Explain string handling in C++ with examples.

**ANS :-**

- **Strlen () :** strlen() function is a built-in function that returns the length of the string and it doesn't count the null character.

**Example :**

```
String str[20];

cout<<"enter the string";

cin>>str;

cout<<strlen(str));
```

- **Strcpy() :** strcpy() function is a built-in function that copies a string from one location to another Or copies the contents from source string to destination string.

**Example :**

```
String str1[20], str2[20];

Cout<<"enter the string1";

Cin>>str1;

Cout<<strcpy(str2, str1));
```

- **Strcat() :** strcat() function is concats or joined first string with second string.

**Example :**

```
string str1[20], str2[20], str3[20];

cout<<"enter the string1";

cin>>str1;

cout<<"enter the string2";

cin>>str2;
```

```
strcpy(str3, strcat(str1, str2));

cout<<str3;
```

- **Strcmp() :** strcmp() is a built-in functions that is used to compare two strings and both strings are equal it returns 0.

**Example :-**

```
String str1[20],str2[20];

if(strcmp(str1, str2)==0)

        cout<<"\n\n\t Strings are equal.";

else

        cout<<"\n\n\t Strings are not equal.";
```

- **Strrev() :** strrev() is built-in functions that reverse the string .

## Q3. How are arrays initialized in C++? Provide examples of both 1D and 2D arrays.

**ANS :-**

**One dimensional :-**

**How initialized in 1D in c++ :-**

```
int arr[5]={8,6,4,2,9};
```

**example :**

```
int arr[5]={10,20,30,40,50};

int i;

for(i=0;i<5;i++)

{
```

```
        Cout<<arr[i];
}
```

**How 2D initialized in C++ :-**

two dimensional initialized : int arr[2][2]={[8,9],[3,5],[1,2]}

## Example:

```
#include<iostream>
Using namespace std;
main()
{
        int arr[2][2]={{10,20},{30,40}};
        int r,c;
        for(r=0;r<2;r++)
        {
        for(c=0;c<2;c++)
                {
                        Cout<<arr[r][c];
                }
                Cout<< "\n";
}
}
```

## Q4. Explain string operations and functions in C++.

**ANS :-**

- **gets()** : to read the string with space.
- **puts()** : to print the string
- **strlen()** : to find out the length of the string
- **strrev()** : to reverse the string
- **strlwr()** : to convert into lower case
- **strupr()** : to convert into upper case
- **strcpy()** : to copy one string to another.
- **strcat()** : To concate two strings.
- **strcmp()** : To compare two strings. (by default refer the case)
- **stricmp()** : To compare two strings. (by ignoring the case)

## functions:

Function is block of code in c++. use the function () round breaket .

## function is two type :

- 1) Build in function : build in function is already c++ language provide.it is not created by developer.
- 2) User define function : user define function is provide by developer.

## Mainly four type of user define function:

- without argument and without return value
- without argument and with return value
- with argument and without return type
- with argument and with return type

## Function mainly three part :

- Function Declaration
- Function call
- Function definition

**TOPIC :-( 6. Introduction to Object-Oriented Programming)**

## Q1. Explain the key concepts of Object-Oriented Programming (OOP).

## ANS :

**Key concepts of Object-oriented programming :**

1) class
2) object
3) encapsulation
4) inheritance
5) polymorphism
6) abstraction

**class : -**

- A class is a template or blueprint that specifies data member and data function.
- A class is a prototype or blueprint from which objects are created.
- Data member know as a class member declared in variable inside the class.
- Data function known as a method declared in function inside the class.

**object:**

- object is a instance of the class.
- Object is a Basic unit of Object Oriented programming.
- It is basically used to assigned the memory to the class (data member, member function).
- Class and object are dependent on each other.

**Encapsulation :-**

- A wrapping of data into single unit is known as encapsulation.
- In c++ the data is not accessible to the outside world.
- Only those functions can access it which is wrapped together within single unit.

**inheritance :-**

- Inheritance is the process, by which class can acquire the properties and methods of another class.
- The mechanism of deriving a new class from an old class is called inheritance.
- The new class is called derived class and old class is called base class.
- The derived class may have all the features of the base class.
- Programmer can add new features to the derived class.
- For example, Student is a base class and Result is derived class.

**polymorphism :-**

- Polymorphism means one name to many forms.
- A Greek word Polymorphism means the ability to take more than one form.
- Polymorphism allows a single name to be used for more than one related purpose.
- The concept of polymorphism is characterized by the idea of 'one interface, multiple methods',

**Two types of polymorphism :-**

- compile time
- run time

1. **Compile time :-**
   - compile time also known as a overloading.
   - Mainly two types of compile time :
     1) function overloading
     2) operator overloading
2. **Run time :-**
   - run time also known as overriding.
   - Run time polymorphism only function overriding alive to program.

**Abstraction :**

- Abstraction is one of the most important concepts of object-oriented programming.
- It refers to the showing only relevant information to the outside world.
- In simple words, hiding any background information from the outside world.

- It is used to implement in class to provide data security.

## Q2. What are classes and objects in C++? Provide an example.

## ANS :-

**Class:-**

- A class is a template or blueprint that specifies data member and data function.
- A class is a prototype or blueprint from which objects are created.
- Data member know as a class member declared in variable inside the class.
- It defines attributes and methods.
- Data member always private in class.
- Data function known as a method declared in function inside the class.

**object:-**

- object is a instance of the class.
- Object is a Basic unit of Object Oriented programming.
- It is basically used to assigned the memory to the class (data member, member function).
- Class and object are dependent on each other.

## Example :-

```
#include<iostream>

using namespace std;

class Employee

{

private:

        int emp_id;

        string emp_name;
```

```
        int emp_salary;
        string dep;
public:
        void get_employee()
                {
                        cout<<"\n\n\t Input emloyee id :";
                        cin>>emp_id;
                        cout<<"\n\n\t Input employee name :";
                        cin>>emp_name;
                        cout<<"\n\n\t Input employee salary :";
                        cin>>emp_salary;
                        cout<<"\n\n\t Input employee deparment :";
                        cin>>dep;
                }
                void print_employee()
                {
                        cout<<"\n\n\t employee id = "<<emp_id;
                        cout<<"\n\n\t emplyee name = "<<emp_name;
                        cout<<"\n\n\t employee salary = "<<emp_salary;
                        cout<<"\n\n\t deparment = "<<dep;
                }
};
```

```
main()
{
        Employee E;
        E.get_employee();
        E.print_employee();
}
```

## Q3. What is inheritance in C++? Explain with an example.

**ANS :-** To derive the properties of one class to another class.

- Inheritance is the process, by which class can acquire the properties and methods of another class.
- The mechanism of deriving a new class from an old class is called inheritance.
- The new class is called derived class and old class is called base class.
- The derived class may have all the features of the base class.
- Programmer can add new features to the derived class.
- For example, Student is a base class and Result is derived class.

**5 types of inheritance :-**

- single inheritance
- multilevel inheritance
- multiple inheritance
- Hierarchical inheritance
- Hybrid inheritance

**Example of inheritance :-**

```
#include<iostream>
using namespace std;
class stu_detail
{
        protected:
                int roll_no;
```

```
                string sname;
                string city;

        public:
                void get_data();
                void print_data();
};
void stu_detail::get_data()
{
                cout<<"\n\n\t Enter Student Roll no :- ";
                cin>>roll_no;
                cout<<"\n\n\t Enter Student name :-  ";
                cin>>sname;
                cout<<"\n\n\t Enter Student city :- ";
                cin>>city;
}
void stu_detail::print_data()
{
        cout<<"\n\n\t Student roll no :- "<<roll_no;
        cout<<"\n\n\t Student name :- "<<sname;
        cout<<"\n\n\t Student City :- "<<city;
}
class stu_marks :public stu_detail
{
        private:
                int sub[5];
                float per , total;

        public:
                void get();
                void put();
};

void stu_marks::get()
{
        for(int i=0;i<5;i++)
        {
```

```
                cout<<"\n\n\t sub["<<i+1<<"] = ";
                cin>>sub[i];
                total=total+sub[i];
        }
        per=total/5;
}
void stu_marks::put()
{
        for(int i=0;i<5;i++)
        {
        cout<<"\n\n\t sub["<<i+1<<"] = "<<sub[i];
        }
        cout<<"\n\n\t student subject total :- "<<total;
        cout<<"\n\n\t student percentage :- "<<per;
}
main()
{
        stu_marks sm;
        sm.get_data();
        sm.get();
        sm.print_data();
        sm.put();
}
```

## Q4. What is encapsulation in C++? How is it achieved in classes?

**ANS :-**

**Encapsulation :-**

- A wrapping of data into single unit is known as encapsulation.
- In c++ the data is not accessible to the outside world.
- Only those functions can access it which is wrapped together within single unit.

How Achieved encapsulate in classes :-

**Access Specifiers**: Access specifiers determine the visibility of the class members. The three primary access specifiers are:

1. **public**: Members declared as public are accessible from outside the class.

2. **private**: Members declared as private are only accessible within the class itself.
3. **protected**: Members declared as protected are accessible within the class and by derived classes (but not by external code).