OOPS 2MARK

1. Define abstraction and encapsulation.

    Abstraction:

    It hides the essential details of the program.

    Encapsulation:

    It hides the entire implementation of the program from users.

2. What are reference variables in C++? Give an example.

    In C++, a reference variable is a variable that is used to refer to an existing variable. This type of variable is created using the & operator.

Example.

```
#include <iostream>
using namespace std;
int main() {
  int a = 8;
  int& b = a;
  cout << "The variable a : " << a;
  cout << "\nThe reference variable r : " << b;
  return 0;
}
```

Output:

```
The variable a : 8
The reference variable r : 8
```

3. Write a C++ program to convert Fahrenheit to Celsius using functions.

```
#include<iostream>
using namespace std;
int main()
{
   float fahrenheit, celsius;
   cout<<"Enter the Temperature in Fahrenheit: ";
   cin>>fahrenheit;
   celsius = (fahrenheit-32)/1.8;
   cout<<"\nEquivalent Temperature in Celsius: "<<celsius;
   cout<<endl;
   return 0;
}
```

Output:

```
        Enter the Temperature in Fahrenheit : 80
        Equivalent Temperature in Celsius : 26.667
```

4. Write a program to print the following pattern for N terms:

```
5

45

345

2345

12345
```

```
#include<iostream>

using namespace std;

int main()

{

   int i,j;

   for(i=5;i>0;i--)

   {

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

      {

         cout<<j;

      }

      cout<<"\n";

   }

}
```

Output:

```
5

45

345

2345

12345
```

5. State the use of access specifiers in C++.

The public keyword is an **access specifier.** Access specifiers define how the members (attributes and methods) of a class can be accessed.

In C++, there are three access specifiers:

- public - members are accessible from outside the class
- private - members cannot be accessed (or viewed) from outside the class
- protected - members cannot be accessed from outside the class, however, they can be accessed in inherited classes. You will learn more about Inheritance later.

6. Show the class specification and instance creation in C++ with an example.

Class:

A class in C++ is the building block that leads to Object-Oriented programming. It is a user-defined data type, which holds its own data members and member functions, which can be accessed and used by creating an instance of that class. A C++ class is like a blueprint for an object.

Object:

An **Object** is an instance of a Class. When a class is defined, no memory is allocated but when it is instantiated (i.e. an object is created) memory is allocated.

Syntax:

ClassName ObjectName;

Example:

```
#include<iostream>

Using namespace std;

class MyClass {       // The class
  public:             // Access specifier
    int myNum;        // Attribute (int variable)
    string myString;  // Attribute (string variable)
};

main()

class MyClass {       // The class
  public:             // Access specifier
    int myNum;        // Attribute (int variable)
    string myString;  // Attribute (string variable)
};

int main() {
  MyClass myObj;  // Create an object of MyClass

  // Access attributes and set values
  myObj.myNum = 15;
  myObj.myString = "Some text";

  // Print attribute values
  cout << myObj.myNum << "\n";
  cout << myObj.myString;
  return 0;
}
```

7. What are the different ways to define symbolic constants in C++.

In C++ programming language the symbolic constants can be declared in two ways:

- Using 'const' keyword.
- Using 'define' directive.

8. What is the use of scope resolution operator in C++?

- To access a global variable when there is a local variable with same name.
- To define a function outside a class.
- To access a class’s static variables.

- In case of multiple Inheritance:

  If same variable name exists in two ancestor classes, we can use scope resolution operator to distinguish.
- For namespace:

  If a class having the same name exists inside two namespace we can use the namespace name with the scope resolution operator to refer that class without any conflicts.
- Refer to a class inside another class.

9. Write a program to check if an integer is divisible by 3 and 5.

```
#include<iostream>
using namespace std;
int main()
{
        int num;
        cout<<"Enter a number: ";
        cin>>num;
        if((num%3==0)&&(num%5==0))
        {
                cout<<"It is divisible by 3 and 5";
        }
```

```
    else
    {
        cout<<"It is not divisible by 3 and 5";
    }
    return 0;
}
```

Output:

```
Enter the number:15
It is divisible by 3 and 5
```

10. Write a program to count the number of occurrences of the given element in an array.

```
    #include<iostream>
using namespace std;

int countOccurrences(int a[], int n, int x)
{
  int res = 0;
  for (int i=0; i<n; i++)
    if (x == a[i])
     res++;
  return res;
}
```

```
int main()
{
   int a[] = {5,66,9,10,10,76,89,100,100};
   int n = sizeof(a)/sizeof(a[0]);
   int x = 10;
   cout << countOccurrences(a, n, x);
   return 0;
}
```

Output:

```
        2
```

11. How does static data members are created and referenced by objects in C++?

It is a variable which is declared with the static keyword, it is also known as class member, thus only single copy of the variable creates for all objects.

Any changes in the static data member through one member function will reflect in all other object's member functions.

Declaration:

```
        static data_type member_name;
```

Defining the static data member:

It should be defined outside of the class following this syntax:

data_type class_name :: member_name =value;

12. Compare static and dynamic objects in C++.

Static object	Dynamic object
Memory is acquired automatically.	Memory is acquired by program with an allocation request. new operation
Memory is returned automatically when object goes out of scope.	Dynamic objects can exist beyond the function in which they were allocated. Object memory is returned by a deallocation request delete operation

5MARKS

1. What are inline functions? Give an example. List any 2 advantages of it.

Inline function is a function that is expanded in line when it is called. When the inline function is called whole code of the inline function gets inserted or substituted at the point of inline function call. This substitution is performed by the C++ compiler at compile time.

Example:

```
#include <iostream>
using namespace std;
inline int cube(int s)
{
        return s*s*s;
}
int main()
{
        cout << "The cube of 3 is: " << cube(3) << "\n";
        return 0;
}
```

Output:

```
 The cube of 3 is: 27
```

Advantages:

1) Function call overhead doesn't occur.

2) It also saves the overhead of push/pop variables on the stack when function is called.

3) It also saves overhead of a return call from a function.

2. Explain nested member functions with an example program.

A member function can call another member function of the same class directly without using the dot operator. This is called as nesting of member functions.

Example:

```
#include<iostream>

using namespace std

class nest

{

   int a;

   int square_num( )

   {

      return a* a;

   }

   public:


  void input_num( )


   {


   cout<<"\nEnter a number ";
```

```
    cin>>a;
    }
    int cube_num( )
    {
        return a* a*a;
    }
void disp_num()
{
    int sq=square_num();       //nesting of member function
    int cu=cube_num();  //nesting of member function
    cout<<”\nThe square of “<<a<<” is  ” <<sq;
    cout<<”\nThe cube of “<<a<<” is  ” <<cu;

}

};

int main()

{

    nest n1;

    n1.input_num();

    n1.disp_num();

    return 0;

}
```

Output:

```
Enter a number 5

The square of 5 is 25

The cube of 5 is 125
```

3.Differentiate between C and C++.

C	C++
Polymorphism, encapsulation, and inheritance are not supported in C, hence object-oriented programming is not possible.	Because it is an object-oriented programming language, C++ provides polymorphism, encapsulation, and inheritance.
C contains 32 keywords.	C++ contains 63 keywords.
C is a subset of C++.	C++ is a superset of C.
A program written in the C programming language has a .c file extension.	A C++ program's file extension is .cpp.
In C, data and functions are kept separate.	In C++, both data and functions are enclosed together.
Memory allocation is handled by the calloc() and malloc() routines.	For memory allocation, the new operator and free() are used.

4. Explain static member functions with an example program.

When we declare a member of a class as static it means no matter how many objects of the class are created, there is only one copy of the static member. A static member is shared by all objects of the class. All static data is initialized to zero when the first object is created, if no other initialization is present.

Syntax:

class_name::function_name (parameter);

Example:

```
#include <iostream>
using namespace std;
class Note
{

static int num;

public:
static int func ()
{
return num;
}
};
```

```
  int Note :: num = 5;


  int main ()

  {

 cout << " The value of the num is: " << Note:: func () << endl;

  return 0;

  }
```

Output

```
The value of the num is: 5
```