C++ - Lab Viva Questions

1) **Who developed C++?**
   Bjarne Stroustrup
2) **Basic concepts of OOP.**
   Classes
   Objects
   Abstraction
   Encapsulation
   Inheritance
   Polymorphism
3) **2 operators that cannot be overloaded.**

2 operators that cannot be overloaded.

Conditional or Ternary Operator (?:) cannot be overloaded.

Size of Operator (sizeof) cannot be overloaded.

4) **3 steps in implementing functions.**
   3 steps in implementing functions.
   •declare a function
   •define it
   •call it.
5) **Function overloading.**

In C++, two functions can have the same name if the number and/or type of arguments passed is different. These functions having the same name but different arguments are known as overloaded functions.

6) **Operator overloading.**

Operator overloading is a compile-time polymorphism. It is an idea of giving special meaning to an existing operator in C++ without changing its original meaning

7) **Nesting of member functions.**

A member function can be called by using its name inside another member function of the same class. This is known as nesting of member functions.

8) **Peculiarity of Static data members & member functions.**

Static data members are class members that are declared using static keywords. Static member functions in C++ are the functions that can access only the static data members.

9) **String handling functions.**
   Strcpy
   Strcmp
   Strcat
   Strlen
10) **Inheritance – Concept of reusability.**
   The capability of a class to derive properties and characteristics from another class is called Inheritance. The new class created is called "derived class" or "child class" and the existing class is known as the "base class" or "parent class". The 5 types of inheritance in C++ are:
   Single Inheritance.
   Multiple Inheritance.
   Multilevel Inheritance.
   Hierarchical Inheritance.
   Hybrid Inheritance.
11) **Differences in implementation of member functions & friend functions.**
   Friend function is a non-member function that has access to private and protected members of a class. It is not in the scope of the class in which it is declared.
   Member function is in scope of the class in which it is declared. A friend function cannot be called using object of the class.

12) **Differences between friend function & friend class.**
   A friend function in C++ is defined as a function that can access private, protected and public members of a class.
   A friend class can access private and protected members of other classes in which it is declared as a friend.
13) **Data hiding \ Information hiding.**

Data abstraction refers to providing only essential information to the outside world and hiding their background details

14) **Visibility labels (modes) \ Access specifiers.**
   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.

15) **Why class is called ADT?**

A class containing varoius objects implies a set of data members alomg with their operations to be performed. The handling of instance variables is done through member methods of a class . This is the reason why a class is known as an abstract data type.

16) **How message passing is implemented in C++?**

Message passing in C++ is the method of communication between two or more objects. A message is used to communicate with other objects by invoking the function of other objects.

17) **How Symbolic constants are generated in C++?**

A symbolic constant can be defined by using it as a label or by using it as a . set statement.

18) **How 'private' member can be made inheritable?**

The private members of a class can be inherited but cannot be accessed directly by its derived classes. They can be accessed using public or protected methods of the base class.

19) **Arrays of objects.**

In C++, an array of objects is a collection of objects of the same class type that are stored in contiguous memory locations.

20) **Abstract class.**

An abstract class is a class that is designed to be specifically used as a base class.

21) **Virtual base class.**

Virtual base classes in C++ are used to prevent multiple instances of a given class from appearing in an inheritance hierarchy when using multiple inheritances.

22) **Difference between 'main()' in C & C++.**

In C++ we cannot call a main() function from any other point. The main() function is the single execution point. However, in C language , we can have a main() function called by the other functions in the code.

23) **Difference between structure & class.**

A structure is a collection of variables of different data types with the same name.
A class in C++ is a single structure that contains a collection of related variables and functions.

24) **Difference between Constructors & Local functions.**

Constructor is a block of code that initializes a newly created object.
Function is a group of statements that can be called at any point in the program using its name to perform a specific task.

25) **Difference between Top-down & Bottom-up approach.**

In the top-down approach, a bigger module/problem is divided into smaller modules.

In the bottom-up approach, the smaller problems are solved and then they are integrated to find the solution of a bigger problem.

26) **Difference between Procedure-oriented & Object-oriented programming concept.**
POP follows a top-down approach, whereas OOPs follow a bottom-up approach.
•Procedural programming is about writing procedures or functions that perform operations on the data, while object-oriented programming is about creating objects that contain both data and functions.

27) **Storage capacity of basic data types.**
char - 1 byte
int - 2 or 4 bytes
float - 4 bytes
double- 8 bytes
boolean- 1 byte

28) **Difference in arithmetic operations of pointer variables & ordinary variables.**

The value of the pointer will get increased by the size of the data type to which the pointer is pointing.

29) **Purpose of using 'this' pointer.**

The this pointer is a pointer accessible only within the nonstatic member functions of a class , struct , or union type. It points to the object for which the member function is called.

30) **Manipulators.**
Manipulators are helping functions that can modify the input/output stream. It does not mean that we change the value of a variable, it only modifies the I/O stream using insertion (<<) and extraction (>>) operators.

31) **C++ Streams.**
A C++ stream is a flow of data into or out of a program, such as the data written to cout or read from cin.

32) **Type compatibility \ Implicit Conversions \ Type casting.**
A type cast is basically a conversion from one type to another. There are two types of type conversion:
•Implicit Type Conversion Also known as 'automatic type conversion'.Done by the compiler on its own, without any external trigger from the user.Generally takes place when in an expression more than one data type is present.
•Explicit Type Conversion: This process is also called type casting and it is user-defined. Here the user can typecast the result to make it of a particular data type.
In C++, it can be done by two ways:
Converting by assignment and Conversion using Cast operator.

33) What happens when private, public & protected members of a class are inherited in private mode?

With private inheritance, public and protected member of the base class become private members of the derived class.

34) What happens when private, public & protected members of a class are inherited in protected mode?

With protected inheritance, the public and protected members become protected, and private members stay inaccessible.

35) What happens when private, public & protected members of a class are inherited in public mode?

36) In public visibility mode, the public, private, and protected members of the base class remain public, private, and protected members respectively in the derived class as well.