@ashish-gupta003

**C++ IMPORTANT QUESTIONS**

1. key difference between C and C++.

* C is a procedure-oriented language whereas C++ is an object-oriented language.
* C++ supports exception handling.
* C++ supports type-safe linking.
* C follows a top-down approach whereas C++ follows a bottom-up approach.

1. what are cout and cin?

* cout and cin are the predefined object.
* cout calls its operator called insertion like cout<<
* cin calls its operator called extraction like cin>>

1. what is endl?

* endl is a manipulator used for the new line

1. what is <iostream>?

* iostream is a standard C++ header file containing declarations of cout and cin.
* iostream.h was an old and non-standard way to declare cout and cin.

1. what is namespace?

* namespace in C++ allowed us to group the named entities like variables, functions, classes, etc.
* namespace declarations appear only in the global section.
* It’s used to prevent name conflict.

1. define inline function.

* an inline function is a function that is expanded in line from where it is invoked that is compiler replaces the corresponding function code
* whereas inline is the keyword and request to the compiler so it depends on the compiler to expand the function code or not.

1. call by value | call by address | call by reference

* when we pass the ordinary variable to the actual argument then it is called a **call by value.**
* when we pass the address of the variable to the actual argument then it is called a **call by address.**
* when we pass the ordinary variable to the actual argument but the formal argument receives them as reference variables then it is called a **call by reference.**

1. define early binding and late binding.

* when function calls bind with their appropriate function definition at a compile time is known as early binding.
* when function calls bind with their appropriate function definition at a Run time is known as late binding.

1. define Polymorphism.

* when two or more tasks can be done by a single entity then it is called polymorphism.
* for example, “+” operator work as addition as well as concatenation.
* There are two types of polymorphism **Compile time polymorphism** and **Run time polymorphism.**
* Compile time polymorphism can be achieved through *function overloading* or *operator overloading.*
* whereas Run time polymorphism can be achieved through *virtual function.*

1. define Encapsulation.

* Rapping of variable(properties) and function(method) related to the same entity is known as Encapsulation.
* we can implement Encapsulation using structure or class

1. what is Data Hiding?

* Securing encapsulated data is known as data hiding.
* And we secure data by using access specifiers that are public, protected, and private.

1. Difference between structure and class?

* by default, members of the class are private and members of the structure are public

1. define class?

* class is a keyword to create a custom data type (like structure)
* it is a collection of variables, functions, and operators.
* class is a blueprint or description of an object.

1. define Constructor.

* Constructor is a special member function of the class whose name is the same as the class name.
* Constructor has no return type.
* Constructor is implicitly invoked at the time of object creation.
* Constructor is responsible for initializing the object.
* Types: parameterized constructor, default constructor, copy constructor.

1. define Default Constructor.

* when we do not define any constructor in the class then the compiler makes a constructor with an empty body which is called the default constructor.

1. define Parameterized Constructor.

* when we define a constructor with parameters then it is called the parameterized constructor.
* Constructor arguments are passed at the object creation.

1. define Copy Constructor.

* Copy constructor is invoked for a newly created object which is initialized with the object of the same class.
* Formal argument of copy constructor must be a reference variable of the same class.

1. define Destructor.

* Destructor is a special member of the class whose name is the same as the name of the class but preceded with a tilde (~) symbol.
* Destructor has no return type.
* Destructor takes no argument i.e. overloading is not possible.
* Destructor is invoked implicitly when an object is about to destroy.
* The job of the destructor should be to free up the memory resources handled by the object.

1. define Operator Overloading.

* when one operator symbol is overloaded with multiple interpretations is known as operator overloading.
* we need to use the operator keyword to define the meaning of the predefined operator to overload them.

* For example, ‘+’ operator works as addition for integer and concatenation for string.
* Operator that can’t be overloaded i.e. scope resolution (: :), sizeof, member access (.), pointer to member (.\*), conditional operator (? :)

1. define Friend Function.

* friend function is a non-member function of the class but the ability to access the private and protected properties of that class.
* friend function is declared inside of the class with the friend keyword but defined outside

1. what is Exception Handling?

* An unwanted event that occurs during the execution of the program is known as an Exception.
* C++ provides the following specialized keywords to Handle the Exception.

try: Represents a block of code that can throw an exception.

catch: Represents a block of code that is executed when a particular exception is thrown.

throw: Used to throw an exception. Also used to list the exceptions that a function throws but doesn’t handle itself.

1. define Pointer.

* Pointer is a variable that contains the address of another variable.
* Types: wild pointer, null pointer, dangling pointer, void pointer, this pointer

1. define wild pointer.

* An Uninitialized pointer is known as a wild pointer.

1. define null pointer.

* A pointer containing a NULL (special address) is known as a NULL pointer.
* If a pointer contains NULL, we consider it as it is not pointing to any location.
* As a safeguard against the illegal use of pointers you can check for NULL before accessing the pointer variable.

1. define dangling pointer.

* A pointer pointing to a memory location that has been deleted is called a dangling pointer.

1. define void pointer.

* Void pointer is a generic pointer that has no associated data type with it. i.e., the void pointer can hold the address of any data type.
* However, void pointer can’t be dereferenced directly but we can dereference by typecasting.

1. define this pointer.

* this is a keyword and local object pointer in every instance member function which contains the address of the current object.

1. define memory leak.

* when we allocate memory dynamically and put the address in a pointer and after some time, we change the address of that pointer without deallocating dynamically created memory then it causes a memory leak.

1. what is inheritance?

* deriving a new class by inheriting properties and methods of an old class is known as inheritance.
* type: single inheritance, multilevel inheritance, multiple inheritances, hierarchical inheritance, hybrid inheritance.

1. define single inheritance.

* The inheritance in which a single derived class is inherited from a single base class is known as Single Inheritance.

1. define Multilevel inheritance.

* Suppose there are two derived classes B and C, where B is inherited from base class A, and C is inherited from B, then it is known as Multilevel Inheritance.

1. define Multiple inheritance.

* The inheritance in which a single derived class is inherited from two base classes is known as Multiple Inheritance.

1. define hierarchical inheritance.

* The inheritance in which two derived classes are inherited from the same base class is known as Hierarchical Inheritance.

1. define Hybrid inheritance.

* The combination of multiple inheritance and hierarchical inheritance is known as Hybrid Inheritance, but Hierarchical Inheritance cause Diamond Problem to solve this we need to use Virtual Inheritance.

1. define Virtual Function.

* virtual is the keyword used to instruct the function or class to execute at run time.
* virtual function is the function that executes at run time by its caller function, this is why it is called run time polymorphism.

1. define Pure Virtual Function.

* pure virtual function is the function that makes the class incomplete so that we can’t create an object of that class.
* we can make an abstract class by using pure virtual function.
* it is initialized by 0, i.e.

virtual type functionName () = 0;

* if we inherit the abstract class into derived class then we need to redefine the pure virtual function else derived class also become abstract class.

Thank You