28 Lecture - CS304

Important Subjective

1. **What is a virtual function in C++ and how is it declared?**
   Answer: A virtual function is a function that can be overridden by a derived class. It is declared using the virtual keyword before the function prototype in the base class.

   What is the difference between a virtual function and a non-virtual function in C++?
   Answer: A virtual function can be overridden by a derived class, while a non-virtual function cannot be overridden.

   Can a virtual function be static or friend in C++?
   Answer: No, a virtual function cannot be static or friend in C++.

   What is the purpose of a pure virtual function in C++ and how is it declared?
   Answer: A pure virtual function is a virtual function that has no implementation in the base class and must be overridden by the derived class. It is declared using the syntax virtual void functionName() = 0;

   Can a virtual function be defined outside of its class in C++?
   Answer: Yes, a virtual function can be defined outside of its class in C++.

   Can a constructor or destructor be virtual in C++?
   Answer: Yes, a constructor or destructor can be virtual in C++.

   What is a virtual function table (vtable) in C++?
   Answer: A virtual function table is a table that stores the addresses of all virtual functions in a class hierarchy.

   What is the difference between early binding and late binding in C++?
   Answer: Early binding is when the function call is resolved at compile-time based on the type of the object pointer, while late binding is when the function call is resolved at runtime based on the type of the object pointed to.

   What is the difference between function overloading and function overriding in C++?
   Answer: Function overloading is when multiple functions have the same name but different parameters, while function overriding is when a derived class provides its own implementation of a virtual function in the base class.

   What is the purpose of a virtual destructor in C++?
   Answer: The purpose of a virtual destructor is to ensure that the destructor of the derived class is called when a derived class object is deleted through a pointer to the base class.