44 Lecture - CS304

Important Subjective

1. **What is stack unwinding in C++?**
   Answer: Stack unwinding is a process in C++ that happens when an exception is thrown. It involves removing all the objects in the call stack in reverse order of their creation until the corresponding catch block is found.

   How does stack unwinding work?
   Answer: When an exception is thrown, the C++ runtime system starts unwinding the call stack to find a catch block that can handle the exception. It does this by destroying all the objects created in the call stack in reverse order of their creation until the catch block is found.

   What is the role of catch blocks in stack unwinding?
   Answer: Catch blocks in C++ are used to handle exceptions that are thrown by a program. When an exception is thrown, the runtime system starts unwinding the call stack to find a catch block that can handle the exception. Once the catch block is found, the exception is handled and the program continues execution.

   Can stack unwinding cause memory leaks?
   Answer: Yes, stack unwinding can cause memory leaks if the objects created on the stack were not properly deleted before the exception was thrown. In this case, the objects will not be deleted and will cause memory leaks when the stack is unwound.

   How can you prevent memory leaks in stack unwinding?
   Answer: To prevent memory leaks in stack unwinding, it is important to properly manage memory in your program. You should use smart pointers, such as shared_ptr or unique_ptr, to manage objects on the heap. You should also ensure that all objects created on the stack are properly deleted before an exception is thrown.

   What is the difference between stack unwinding and stack trace?
   Answer: Stack unwinding is a process that happens when an exception is thrown in C++. It involves removing all the objects in the call stack in reverse order of their creation until the corresponding catch block is found. A stack trace, on the other hand, is a record of the function calls that led to a particular point in a program's execution.

   Can stack unwinding be disabled in C++?
   Answer: Stack unwinding cannot be disabled in C++. It is a fundamental mechanism that is used to handle exceptions in the language. However, you can use techniques such as RAII (Resource Acquisition Is Initialization) to ensure that resources are properly managed in your program and prevent memory leaks.

   What is the impact of stack unwinding on performance?
   Answer: Stack unwinding can have a significant impact on performance in C++. This is because it involves the destruction of all the objects in the call stack in reverse order of their creation. However, the impact can be minimized by properly managing memory in your program and

using techniques such as RAII.

Can stack unwinding cause data corruption?
Answer: Yes, stack unwinding can cause data corruption if the objects created on the stack were not properly deleted before the exception was thrown. In this case, the objects will not be deleted and can cause data corruption when the stack is unwound.

How does C++ ensure that objects are properly deleted during stack unwinding?
Answer: C++ ensures that objects are properly deleted during stack unwinding by automatically calling the destructors of objects created on the stack as the stack is unwound. This is done in reverse order of the objects' creation to ensure that they are properly cleaned up.