HW15

1) Suppose someone designed a stack abstract data type in which the function top returned an access path (or pointer) rather than returning a copy of the top element. This is not a true data abstraction. Why? Give an example that illustrates the problem.

Sol: The problem with this is that the user is given access to the stack through the returned value of the "top" function. For example, if \( p \) is a pointer to objects of the type stored in the stack, we could have:

\[
p = \text{top}(\text{stack1});
\]
\[
*p = 42;
\]

These statements access the stack directly, which violates the principle of a data abstraction.

2) Explain why allowing a class to implement multiple interfaces in Java does not create the same problems that multiple inheritance in C++ creates.

Sol: A significant problem with multiple inheritance is that two of the parents can define a method with the same name and the same protocol. A class that implements an interface must define all of the methods declared in the interface. So, if both the parent class and the interface include methods with the same name and protocol, the subclass must re-implement that method, thereby avoiding the name conflict.