Assignment # 1  
Muhammad Hussnain  
19L-2260

**Question. # 1**

The primary challenge discussed in this context is known as the "diamond problem." This issue arises when a class inherits from two classes that share a common base class. Consequently, this common base class can introduce ambiguity within the derived class. This ambiguity stems from the fact that there are two distinct paths to access the same base class via its parent classes, leading to confusion for the compiler and potentially unexpected program behavior.   
  
In Java, a language that does not support traditional multiple inheritance, a similar functional outcome can be achieved by utilizing interfaces and composition. Rather than directly inheriting from multiple classes, Java encourages the creation of a class that combines functionalities from various classes. For example, consider an "All-Rounder" class that consolidates features from "Batsman" and "Bowler" classes, accomplishing a multifaceted role.   
  
The second issue addressed in this context pertains to Java's limitations, as it does not permit traditional multiple inheritance. In Java, a class can only inherit from a single base class, rendering the direct inheritance from multiple classes impossible. To mitigate this constraint and implement the essence of multiple inheritance, Java relies on interfaces, as explained earlier.