**Internet Programming (handout)**

**Some differences of Java from C++ (on the topic of inheritance)**

The purpose of inheritance is same in C++ and Java. There are following differences in the way both languages provide support for inheritance.

1. **In Java, all classes inherit from the**[**Object class**](http://download.oracle.com/javase/1.5.0/docs/api/java/lang/Object.html)**directly or indirectly**. Therefore, there is always a single inheritance tree of classes in Java, and [Object class](http://download.oracle.com/javase/1.5.0/docs/api/java/lang/Object.html)is root of the tree. In Java, if we create a class that doesn’t inherit from any class then it automatically inherits from Object class . In C++, there is forest of classes; when we create a class that doesn’t inherit from anything, we create a new tree in forest.
2. **In Java, members of the grandparent class are not directly accessible**. (e.g. as in C++ with multiple use of scope resolution operator)
3. The meaning of**protected member access specifier is somewhat different in Java**. In Java, like C++ protected members are accessible to child classes (doesnot matter if bith classes are in same or different package). BUT, unlike C++, protected members of a class “A” are ALSO accessible in other class “B” of same package; even if B doesn’t inherit from A (they both have to be in the same package).
4. Java uses extend keyword for inheritence. Unlike C++, **Java doesn’t provide an inheritance specifier like public, protected or private**. Therefore, we cannot change the protection level of members of base class in Java, if some data member is public or protected in base class then it remains public or protected in derived class. Like C++, private members of base class are not accessible in derived class.  
   Unlike C++, in Java, we don’t have to remember those rules of inheritance which are combination of base class access specifier and inheritance specifier.
5. **In Java,** **methods are virtual by default**. In C++, we explicitly use virtual keyword.
6. Java uses a separate keyword interface for interfaces, and abstract keyword for abstract classes and abstract functions.
7. Unlike C++, **Java doesn’t support multiple inheritance**. A class cannot inherit from more than one class. A class can implement multiple interfaces though.
8. In C++, default constructor of parent class is automatically called, but if we want to call parametrized constructor of a parent class, we must use [Initializer list](http://www.geeksforgeeks.org/archives/13797). Like C++, default constructor of the parent class is automatically called in Java, but if we want to call parametrized constructor then we must use super to call the parent constructor. See following Java example.