COMPSCI 230 Tutorial OO Design Week 4

Q1: Remind ourselves of what the following keywords for OO in java mean:

Interface	
Abstract	
Final	
Super	

Q2: What is the result when you run the following code?

```
package q1;

public class Test1 extends SuperClass {

	int x2= 20;
	static int y2 = 20;

	Test1()
	{
		x2 = y2++;
	}
	public int foo2() {
		return x2;
	}
	public static int goo2() {
		return y2;
	}

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		SuperClass s1 = new SuperClass();
		Test1 t1 = new Test1();
		System.out.println("The Base object");
		System.out.println("S1.x = " + s1.x);
		System.out.println("S1.y = " + s1.y);
		System.out.println("S1.foo() = " + s1.foo());
		System.out.println("S1.goo() = " + s1.goo());
		System.out.println("\nThe Derived object");
		System.out.println("\nInherited fields");
		System.out.println("T1.x = " + t1.x);
		System.out.println("T1.y = " + t1.y);
		System.out.println("T1.foo() = " + t1.foo());
		System.out.println("T1.goo() = " + t1.goo());
		System.out.println("\nThe instance/class fields");
		System.out.println("T1.x2 = " + t1.x2);
		System.out.println("T1.y2 = " + t1.y2);
		System.out.println("T1.foo2() = " + t1.foo2());
		System.out.println("T1.goo2() = " + t1.goo2());
	}
}
```

Please fill in the output on the next page.

Output:

Q3: SuperClass will remain the same, but the Test1 class has slightly changed. What is the output when you run the following code?

```
package q1;

public class Test1 extends SuperClass {
	static int x = 15;
	static int y = 15;
	int x2= 20;
	static int y2 = 20;
	Test1()
	{
		x2 = y2++;
	}
	public int foo2() {
		return x2;
	}
	public static int goo2() {
		return y2;
	}
	public static int goo(){
		return y2;
	}
	public static void main(String[] args) {
		// TODO Auto-generated method stub
		SuperClass s2 = new Test1();
		System.out.println("\nThe static Binding");
		System.out.println("S2.x = " + s2.x);
		System.out.println("S2.y = " + s2.y);
		System.out.println("S2.foo() = " + s2.foo());
		System.out.println("S2.goo() = " + s2.goo());
	}
}
```

Output:

As a bonus: re-add the println/variable commands from the original Test1 to the main() and see how the result changes. These print statements are included in the answer code.

Q4: In which class is the method s2.goo() called? Hint: Run the debugger and step through code.

Q5: What is the static type of variable s2?

Q6: Are we able to make a call to method foo2() from variable s2?

Q7: Test1 t2 = new SuperClass();

What is the result from the above line of code?

Q8: Test1 t2 = (Test1) new SuperClass();

What is the result from the above line of code?

Q9: what is the default visibility for constructor?

Q10 Look at the skeleton code Test2.java. You will see an error message "The constructor SuperClass() is not visible", How to solve this problem?

Q11: How many of the Interface's methods must we provide an implementation for?

Q12: When attempting to add a new method definition to an Interface A, which is used by Class X, it is preferred that the Interface A is extended by another Interface: B. Why is this preferred?

Q13: How many Interfaces can be implemented by one class?

Q14: Abstract classes are similar to interfaces in that we don't have to provide method implementations. However, we are able to provide implemented methods inside an abstract class. How do you define a method in an abstract class for which you don't provide an implementation?

Q15: Can abstract methods be defined in a concrete (Non-abstract) class?

Q16: How do you extend an abstract class A and implements Interfaces B,C, and D in Class SubClass1 at the same time?

Q17: Generics is a feature of the Java programming language that allows type safety. The Java Collections Framework provides some data structures such as List, Arraylist, and LinkedList that can be used with Generics.

If you created a ArrayList (List<String> list = new ArrayList<String>();) using String as the generic, What kind of error would you get if you attempted to place an object that was not of type String into the ArrayList: Run-time or Compile-time?