1. What does this code output?

```
public class Polymor {
	class A {
		public String show(D obj) {
			return ("A and D");
		}
		public String show(A obj) {
			return ("A and A");
		}
	}
	class B extends A {
		public String show(B obj) {
			return ("B and B");
		}
		public String show(A obj) {
			return ("B and A");
		}
	}
	class C extends B {}
	class D extends B {}

	public static void main(String[] args) {
		Polymor outerclass = new Polymor();
		A a1 = outerclass.new A();
		A a2 = outerclass.new B();
		B b = outerclass.new B();
		C c = outerclass.new C();
		D d = outerclass.new D();
		System.out.println("1:" + a1.show(b));
		System.out.println("2:" + a1.show(c));
		System.out.println("3:" + a1.show(d));
		System.out.println("4:" + a2.show(b));
		System.out.println("5:" + a2.show(c));
		System.out.println("6:" + a2.show(d));
		System.out.println("7:" + b.show(b));
		System.out.println("8:" + b.show(c));
		System.out.println("9:" + b.show(d));
	}
```

}

2. Can the following code run successfully? If not, where is the error?

```
public class Box<T extends Integer> {
      private T t;
      public void set(T t) {
            this.t = t;
      }
      public T get() {
            return t;
      }
      public <U extends Number> void inspect(U u){
            System.out.println("T: " + t.getClass().getName());
            System.out.println("U: " + u.getClass().getName());
      }
      public static void main(String[] args) {
            Box<Integer> integerBox = new Box<Integer>();
            integerBox.set(new Integer(10));
            integerBox.inspect("some text");
      }
}
```

3. Can a class implement different instantiations of the same generic interface? If so, give an example.

4. What does < ? > mean, and how does its definition change between < ? extends T> and < ? super T >, for some type T?