1. (2 pts) Java supports inheritance through the keyword **extends** and only supports **single** inheritance since we can only list one class name after the keyword.

2. (2 pts) Suppose we have the following class definitions.

```java
class A extends B
class B extends C
```

Which two of the following declarations will cause a compile-time error? Assume that there are no constructors defined in classes A, B, or C. Explain the compile-time errors.

- A a = new A(); **No compile-time error**
- B b = new A(); **No compile-time error because A “is a” B**
- C c = new A(); **No compile-time error because A “is a” C**
- A a1 = new B(); **Compile-time error because B is not a A**
- B b1 = new C(); **Compile-time error because C is not a B**
- C c1 = new B(); **No compile-time error because B “is a” C**

3. (2 pts) True/False  We can create an instance of a class A without any constructor from Object.java ever being invoked? If class A extends B, then B is called a **superclass** of A.

   A superclass constructor is always invoked from a subclass constructor, and since Object is a superclass of every class definition, eventually a constructor of Object will be invoked.

4. (2 pts) Along with static and instance initializers, **constructors** and **private variables and methods** are not inherited by subclasses.

5. (2 pts) Consider the following class definitions.

```java
class B {
    private int num;
    public B(int num) {
        this.num = num;
    }
}
class A extends B {
    private int num1;
    public A(int num,int num1) {
        this.num1 = num1;
        super(num);
    }
}
```
class C extends B {
}

There are compile-time errors in the class definitions of both A and C. What are they?

There is a compile-time error in class A because if we make a call to super, it must be the first executable line in the subclass constructor.

There is a compile-time error in class C because since there is no constructor defined in C, a default constructor is generated. The compiler will automatically place a call to the superclass constructor with no parameter list in the default constructor. However, since there is already a constructor in class B, but not one with no parameter list, no default constructor will be created.