CSCI-142 Exam 1 Review

September 21, 2017

Presented by the RIT Computer Science Community

http://csc.cs.rit.edu

1. Provide a detailed explanation of what the following code does:

```
1 public boolean checkString(String a, String b) {
2     return a == b;
3 }
```

2. If an instance variable is declared with `protected` access, who can access it?

3. What is the difference between overriding and overloading methods? Give a example situation where each should be used.

4. Explain the differences between:

   (a) `class` vs `object`

   (b) constant vs non-constant field (variable). Declare a constant.

   (c) final vs non-final method.

   (d) class vs instance variable. Declare variables of both types.

5. Define polymorphism and explain a situation in which you would use it.

6. What is the difference between an abstract class and an interface? Why might you want to use an interface over an abstract class?

7. Describe the difference between `Comparable` and `Comparator`.

8. What is the Java Collections Framework?

9. What gets printed by the following code?

```
1 public class Class1 {
2     public Class1() {
3         System.out.println( "Class1()" );
4     }
```

```
5     public void print1() {
6         System.out.println( "Class1.print1()" );
7     }
8     public void print2() {
9         System.out.println("Class1.print2()" );
10    }
11 }
12
13 public class Class2 extends Class1 {
14    public Class2() {
15        System.out.println( "Class2()" );
16    }
17    public void print1() {
18        System.out.println("Class2.print1()");
19    }
20 }
21
22 public class Class3 extends Class1{
23    private Class2 class2;
24    public Class3() {
25        System.out.println( "Class3()" );
26        class2 = new Class2();
27    }
28    public void print1() {
29        class2.print1();
30    }
31    public void print2(){
32        System.out.println("Class3.print2()");
33        super.print2();
34    }
35 }
36
37 public class TestClass {
38    public static void main( String[] args ) {
39        Class1 c1 = new Class2();
40        c1.print1();
41        c1.print2();
42        System.out.println();
43        Class1 c2 = new Class3();
44        c2.print1();
45        c2.print2();
46    }
47 }
```

10. Find at least 3 errors related to inheritance and interfaces in the following code:

```
1  public interface Vehicle {
2      public int getSpeed();
3      public void accelerate(int speed_inc);
4      public void brake(int speed_dec);
5  }
6  public class Car implements Vehicle {
7      private int speed;
8      public Car(int initialSpeed){
9          this.speed = initialSpeed;
10     }
11     public int getSpeed(){
12         return speed;
13     }
14     public int accelerate(int speed_inc) {
15         speed += speed_inc;
16         return speed;
17     }
18     public int brake(int speed_dec) {
19         speed -= speed_dec;
20         return speed;
21     }
22 }
23
24 public class Toyota extends Car {
25     public long getSpeed(){
26         return speed;
27     }
28 }
29 public class Truck implements Vehicle {
30     public void accelerate(int speed_inc) {
31         super.accelerate(speed_inc/2);
32     }
33     public void brake(int speed_dec) {
34         super.brake(speed_dec/2);
35     }
36 }
37 public class demolitonDerby {
38     public static void main(String[] args) {
39         Vehicle prius, mack, impreza;
40         prius = new Toyota();
41         mack = new Truck();
42         impreza = new Car();
43
44         impreza.accelerate(5);
45         prius.brake(2);
46         prius.accelerate(impreza.getSpeed());
47         mack.accelerate(5);
48     }
49 }
```

11. Convert the following code to use generics.

```
interface StringCondition {
    boolean checkString(String s);
}

interface IntegerCondition {
    boolean checkInteger(Integer i);
}

class StringContainer {
    ArrayList<String> values;

    // Add and other methods are defined correctly here...

    String getFirstWhereHolds(StringCondition condition) {
        for (String s : values) {
            if (condition.checkString(s))
                return s;
        }
        return null;
    }
}

class IntegerContainer {
    ArrayList<Integer> values;

    // Add and other methods are defined correctly here...

    Integer getFirstWhereHolds(IntegerCondition condition) {
        for (Integer i : values) {
            if (condition.checkInteger(i))
                return i;
        }
        return null;
    }
}
```