Written Assignment 3
Object Orientation

CS152 – Computer Programming Fundamentals
Instructor: M. Wolverton

Q3-1

Define what the keyword ‘static’ means in Java.

List all code contexts where static can be applied to.

Q3-2

The following Java code within SomeProgram.java compiles and runs. It contains a second class Point which is instantiated twice.

```
public class SomeProgram {
       public static void main(String[] args){
         Point A = new Point(3,8);
         Point B = new Point(-2,4);
         A.z = 10;
         // what values do A.x, A.y, A.z and B.x, B.y and B.z hold on this line?
       }
}
class Point {
    int x,y;
    static int z = 0;
    Point(int x, int y) {
          this.x = x;
    }
}
```

What are the values of A.x, A.y, A.z and B.x, B.y, B.z where the comment is located? Are x, y and z handled the same? Explain what significance the keywords ‘static’ and ‘this’ have in this code.

Q3-3

The keywords ‘public’, ‘private’ and ‘protected’ are called access modifiers in Java. Explain what each does as well as default access (excluding the access modifier).

Access Modifier	**Description**
Private	
Public	
Protected	
Default	

Q3-4

When can the Java keyword ‘super’ be used, and what is it used for?

Q3-5

In Java, **Overriding** and **Overloading** both concern method definitions but are very different types of code. Explain what each means with an emphasis on understanding the difference between the two. In what situation is each used, and what does each do?

Overriding

Overloading

Q3-6

Define what the keyword ‘abstract’ means when applied to a Java class definition.

One property of abstract classes is that they cannot be instantiated. How can you make use of abstract classes in your code if they cannot be instantiated? Explain both the required syntax and what it does.

Q3-7

An ArrayList `numList` is declared and initialized in the Java below, followed by a loop intended to remove all negative values.

```
ArrayList<Integer> numList = new ArrayList();
numList.add(7);
numList.add(-9);
numList.add(-4);
numList.add(6);
numList.add(-1);
numList.add(5);
for(int i = 0; i < numList.size(); i++){
       if (numList.get(i) < 0) numList.remove(i);
}
```

At the end of this block, `numList` still holds a negative value. Explain why the loop didn't work as intended, and a correct way fix it.

Q3-8

Some methods in Java fail by **Throwing Exceptions**. If you wish to call any method that throws an exception it must be handled somehow, or the source code will fail to compile. There are two options for handling exceptions in Java. Briefly discuss the two options and the difference between them.