COMP 215 Exam #1

My name is:

- This is a 50 minute exam.
- The exam is open book and open notes. You are free to bring whatever materials you want to the exam. However, the rules are that (a) you cannot use an internet connection during the exam, and (b) you cannot use a Java compiler to help you on the exam (that is, you can't compile or execute any Java code during the exam).
- Write your answers in the spaces provided. Use extra paper to determine your solutions, if needed, then neatly transcribe them onto these sheets.
- Make sure you clearly write your name. Further, please sign the pledge at the bottom of this page.

Pledge: On my honor, I have neither given nor received any unauthorized aid on this exam.

Signed:

Problem 1 [24 Points]

Consider the following code:

```
public abstract class Base {
  abstract public void method1 ();

  public void method2 () {
    System.out.println("Base2");
  }
}

public class SubA extends Base {
  public void method1 () {
    System.out.println("SubA1");
  }

  public void method2 () {
    System.out.println("SubA2");
  }
}

public class SubB extends Base {
  public void method1 () {
    System.out.println("SubB1");
  }
}
```

What will the following print out?

```
  ...
  Base a1 = new SubA ();
  Base b1 = new SubB ();
  SubB b2 = new SubB ();
  a1.method1 ();
  a1.method2 ();
  b1.method1 ();
  b1.method2 ();
  b2.method1 ();
  b2.method2 ();
  ...
```

Problem 2 [20 Points]

Consider the following interface:

```
interface IShape {

  // moves the shape the specified distance vertically
  void moveVertical (double yDist);

  // moves the shape the specified distance horizaontally
  void moveHorizontal (double xDist);

  // returns the area of the shape
  double computeArea ();

  // computes whether the two shapes intersect or not
  boolean intersects (IShape me);
  ...
}

class AShape implements IShape {
  private double xPosOfCenter;
  private double yPosOfCenter;
  ...
}

class Square extends AShape {...}
class Circle extends AShape {...}
class Rectangle extends AShape {...}
```

(a) Where should the implementation of `moveVertical` and `moveHorizontal` go—the abstract class or the concrete classes—and why?

(b) Where should the implementation of `computeArea` go—the abstract class or the concrete classes—and why?

Problem 3 [30 points]

Consider the following classes `MyInt` and `SomeClass`; please note that they have been put side-by-side so that everyhting fits easily on one page:

```
public class MyInt {
  private int data;

  public MyInt (int dataIn) {
    data = dataIn;
  }

  public void print () {
    System.out.println (data);
  }

  public void addToMe (MyInt addMe) {
    data += addMe.data;
  }

  public void swap (MyInt withMe) {
    int temp = data;
    data = withMe.data;
    withMe.data = temp;
  }
}
```

```
public class SomeClass {
  private MyInt innerVal;

  public SomeClass (MyInt valToStore) {
      innerVal = valToStore;
  }

  public void add40 () {
    MyInt tempVal = new MyInt (40);
    innerVal.addToMe (tempVal);
  }

  public void add60 (MyInt toMe) {
    MyInt tempVal = new MyInt (60);
    tempVal.addToMe (toMe);
    toMe = tempVal;
  }
}
```

(a) What will the following code print out?

```
MyInt intOne = new MyInt (30);
MyInt intTwo = new MyInt (40);
intOne.swap (intTwo);
intOne.print ();
intTwo.print ();
```

(b) What will the following code print out?

```
MyInt myInt = new MyInt (30);
SomeClass whatsIt = new SomeClass (myInt);
whatsIt.add40 ();
whatsIt.add60 (myInt);
myInt.print ();
```

(c) What will the following code print out?

```
MyInt intOne = new MyInt (30);
MyInt intTwo = new MyInt (40);
SomeClass whatsIt = new SomeClass (intOne);
intOne.swap (intTwo);
whatsIt.add40 ();
intOne.print ();
intTwo.print ();
```

Problem 4 [26 Points]

Imagine that I have the following interface:

```
interface MyInterface <T, SomeType implements Iterable <T>> {
  SomeType oneMethod (SomeType param);
  T anotherMethod (SomeType param);
}
```

Then I declare the following class:

```
class SomeRandomClass implements MyInterface <Double, ArrayList <Double>> {
...
}
```

(a) Assuming that `SomeRandomClass` is a concrete class, at a minimum, what methods should appear inside of `SomeRandomClass`? Give **precise** method headers, in the sense that they should be totally correct and "compilable".

(b) Will the following compile? Why or why not?

```
class SomeRandomClass implements MyInterface <Double, Double> {
...
}
```

Hint: look at the printout of the first page of the JavaDoc for the `Double` class, which is attached to the end of the exam.