CS 201, Summer 2015 — June 4th — Exam 1 Name: Solution

Question 1. [6 points] Write statements to read two `double` values from the user, compute their sum, and then print the sum.

```
Scanner keyboard = new Scanner(System.in);
double a = keyboard.nextDouble();
double b = keyboard.nextDouble();
System.out.printf("Sum is %f\n", a + b);
```

Question 2. [3 points] Fill in the blank: a(n) checked exception must be dealt with using `try/catch` or by using a `throws` declaration to allow the method to throw the exception out of the method.

Question 3. [3 points] Fill in the blank: a(n) unchecked exception does not need to be dealt with using `try/catch` or by using a `throws` declaration to allow the method to throw the exception out of the method.

Question 4. [3 points] Fill in the blank: the try/finally construct is useful for guaranteeing that cleanup actions (such as closing a `FileReader`) are executed even if an exception is thrown.

Question 5. [3 points] Briefly explain the circumstances under which it is appropriate to throw an exception out of a method rather than handling it using `try/catch`.

When there is nothing reasonable the method can do to recover from the exception

Question 6. [3 points] Briefly explain the circumstances under which it is appropriate to handle an exception using `try/catch` rather than throwing it out of the method.

when it is possible to recover from the exception: e.g. if it is a file not found exception, prompt the user to re-enter the file name and then try opening the file again

Question 7. [6 points] For each of the following C terms, state the closest equivalent Java term:

C term	Java term
Function	method
Struct type	class
Struct instance	object

Question 8. [6 points] What output is printed by the following program (which begins on the left and continues on the right)?

```
public class Box {
  private int value;

  public Box(int v)
    { value = v; }
```

```
  public static void main(
      String[] args) {
    Box b1 = new Box(17);
    Box b2 = new Box(42);

    System.out.println(b1.value);
    System.out.println(b2.value);
  }
}
```

17
42

Question 9. [6 points] What output is printed by the following program (which begins on the left and continues on the right)?

```
public class Box2 {
  private int value;

  public Box2(int v)
    { value = v; }
```

```
  public static void main(
      String[] args) {
    Box2 b1 = new Box2(17);
    Box2 b2 = b1;

    b2.value = 42;

    System.out.println(b1.value);
    System.out.println(b2.value);
  }
}
```

42
42

Question 10. [6 points] What output is printed by the following program (which begins on the left and continues on the right)?

```
public class Mystery {
  public static void f(
      int[] arr) {
    arr[0] = arr[1];
  }
```

```
  public static void main(
      String[] args) {
    int[] nums = new int[3];
    nums[0] = 62;
    nums[1] = 84;
    nums[2] = 4;
    f(nums);
    System.out.printf("%d %d %d\n",
      nums[0], nums[1], nums[2]);
  }
}
```

84 84 4

Question 11. [5 points] Briefly explain the roles of the superclass and subclass in an inheritance ("Is-A") relationship.

Superclass: define (abstractly) common operations

Subclasses: define implementations of the common operations with varying behavior

Question 12. [10 points] Show the code for a class called `Adder` as described below. An `Adder` object should store a single `int` value. The class should have a constructor which allows a new `Adder` object's value to be initialized to a specified value. The class should have a `getValue` method which returns the `Adder`'s current value. The class should have an `add` method which takes an integer value as a parameter and adds it to the `Adder`'s current value.

The following JUnit tests show the expected behavior:

```
Adder a1 = new Adder(0);
Adder a2 = new Adder(16);
assertEquals(0, a1.getValue());
assertEquals(16, a2.getValue());
a1.add(9);
assertEquals(9, a1.getValue());
a1.add(5);
assertEquals(14, a1.getValue());
a2.add(3);
assertEquals(19, a2.getValue());
a2.add(10);
assertEquals(29, a2.getValue());
```

```
public class Adder {
private int value;

    public Adder(int v) {
        value = v;
    }

    public void add(int x) {
        value += x;
    }

    public int getValue() {
        return value;
    }
}
```