CSE 142 SPRING 2008 FINAL EXAM

1. Expressions (10 points)

For each expression in the left-hand column, indicate its value in the right-hand column.
Be sure to list a constant of appropriate type (e.g., `7.0` rather than `7` for a `double`, `String`s in "quotes"). If the expression is illegal, then write "error".

Expression	Value
`!(4 > 3.5) || ((6 < 3) && (4 < 9))`	_____
`99 % 100 + 10 / 4 / 1.0`	_____
`"8" + 3 * 2 + (3 * 25) + 3 * 10 + 3 * 3`	_____
`6 / 2 + (14 < 3)`	_____
`-1 + 11 % 7 * 2 + 1000 + 110 * 3`	_____

2. Array Mystery (15 points)

Consider the following method:

```
public static void mystery(int[] array) {
    for (int i = array.length - 2; i >= 0; i--) {
        array[i] = array[i] + array[i + 1];

    }
}
```

Indicate in the right-hand column what values would be stored in the array after the method `mystery` executes if the integer array in the left-hand column is passed as a parameter to `mystery`.

Array	Final Contents of Array
`int[] a1 = {8, 9};` `mystery(a1);`	_____
`int[] a2 = {14, 7, 1};` `mystery(a2);`	_____
`int[] a3 = {7, 1, 3, 2, 0, 4};` `mystery(a3);`	_____
`int[] a4 = {10, 8, 9, 5, 6};` `mystery(a4);`	_____
`int[] a5 = {8, 10, 8, 6, 4, 2};` `mystery(a5);`	_____

3. Mystery (4 points)

What does the following block of code print?

```
int a = 5;
int b = 2;

if (b > 3 || a < b) {
   System.out.println("Go Huskies!!");
} else if (b > 0 && a < 8) {
   System.out.print("Go ");
} else if (b == 2) {
   System.out.println("Huskies!!");
} else {
   System.out.println("Go Huskies!!");
}
```

4. Mystery (10 points)

What does the following program print?

```
public class Mystery {
    public static void main(String[] args) {
        String fish = "one";
        String two = "fish";
        String one = "red";
        String blue = "two";
        String red = "blue";

        mystery(two, blue, fish);
        mystery(one, two, red);
        fish = "blue";
        mystery("fish", fish, one);
    }

    public static void mystery(String fish, String two, String one) {
        System.out.println(one + " " + fish + ", " + two + " " + "fish");
    }
}
```

5. Inheritance Mystery (10 points)

Assume that the following classes have been defined:

```
public class AcmeLabs extends Brain {
    public String toString() {
        return "One is a genius, the other's insane.";
    }

    public void methodA() {
        System.out.println("AcmeLabs A");
    }
}

public class Pinky extends World {
    public void methodA() {
        System.out.println("Pinky A");
    }

    public String toString() {
        return "I think so, Brain, but...";
    }
}

public class World extends Brain {
    public void methodB() {
        System.out.println("World B");
    }

    public String toString() {
        return "Try to take over the world!";
    }
}

public class Brain {
    public void methodB() {
        System.out.println("Brain B");
    }

    public void methodA() {
        System.out.println("Brain A");
    }

    public String toString() {
        return "Pinky!  Are you pondering what I'm pondering?";
    }
}
```

Given the classes above, what output is produced by the following code?

```
Brain[] insane = { new AcmeLabs(), new Brain(), new Pinky(), new World() };
for (int i = 0; i < insane.length; i++) {
    insane[i].methodA();
    insane[i].methodB();
    System.out.println(insane[i]);
    System.out.println();
}
```

6. Programming (10 points)

Write a static method `isSumArray` that accepts an array of integers and returns whether for every group of three elements in the array, the first two elements sum up to the third. If the size of the array cannot be divided into groups of three, then the array does not pass the test.

For example, given the following arrays:

```
int[] array1 = { 1, 2, 3, 8, 7, 15, 9, 3, 12 };
int[] array2 = { 1, 2, 3, 4, 5 };
int[] array3 = { 6, 11, 2008 };
int[] array4 = { -4, 7, 3, 8, -2, 6 };
```

Calling `isSumArray` will result in the following values:

Call	Value Returned
`isSumArray(array1)`	`true`
`isSumArray(array2)`	`false`
`isSumArray(array3)`	`false`
`isSumArray(array4)`	`true`

In the first array, for every group of three numbers (1-2-3, 8-7-15, and 9-3-12), the first two numbers add up to the third. The second array cannot be divided into groups of three. The third array can be divided, but the first two numbers do not add up to the third.

7. Programming (15 points)

Write a static method `printWinner` that accepts a `Scanner` holding a sequence of names and numbers. Following each name is a series of one or more numbers. A person's sum is the total of the numbers until the next name. The method will print out the name of the person who has the highest sum less than or equal to 21. If everyone's sum is over 21, then the method will print "Everyone busted!" You may assume that there is at least one name.

For example, given the following Scanners:

```
Scanner input1 = new Scanner("alpha 10 5 9 bravo 8 charlie 11 9");
Scanner input2 = new Scanner("delta 10 3 9 echo 3 1 10 10");
Scanner input3 = new Scanner("foxtrot 11 5 6");
Scanner input4 = new Scanner("golf 4 8 hotel 10 6 india 9 8 7");
```

Calling `printWinner` will result in the following output:

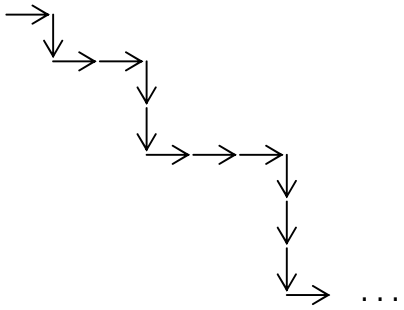
Call	Output
`printWinner(input1)`	`charlie is the winner!`
`printWinner(input2)`	`Everyone busted!`
`printWinner(input3)`	`Everyone busted!`
`printWinner(input4)`	`hotel is the winner!`

In the first example, the respective sums for alpha, bravo, and charlie are 24 (10 + 5 + 9), 8 (8), and 20 (11 + 9). The highest sum that is less than or equal to 21 belongs to charlie. In the second example, the sums are 22 (10 + 3 + 9) and 24 (3 + 1 + 10 + 10). Since no sum is less than or equal to 21, everyone busted.

8. Programming (15 points)

Write the `getMove` method for the class `Dog` that extends the `Critter` class from Homework 8. An instance (object) of the `Dog` class moves as follows: `EAST` 1 time, `SOUTH` 1 time, `EAST` 2 times, `SOUTH` 2 times, `EAST` 3 times, `SOUTH` 3 times, and so on. The `Dog` goes `EAST` and `SOUTH` for the same number of steps and repeats the pattern incrementing the number of steps by one each time. See diagram below.

Use the constants for directions, namely `Direction.NORTH`, `Direction.EAST`, `Direction.SOUTH`, and `Direction.WEST`. You may add anything needed (fields, other methods, constructors, etc.) to implement `getMove` appropriately.




```
public class Dog extends Critter {
    // declare any necessary fields and methods here









    public Direction getMove() {
        // complete the getMove method here




    }
}
```

9. Programming (10 points)

Write a method named `stringLengths` that takes an array of strings as a parameter and returns an array of integers whose elements are the lengths of the corresponding strings in the array parameter.

For example, given the following arrays:

```
String[] array1 = { "you", "say", "goodbye", "and", "i", "say", "hello" };
String[] array2 = { "i", "love", "CSE", "142" };
String[] array3 = { "thisisaverylongstring" };
String[] array4 = { "strings", "strings", "everywhere" };
```

Calling `stringLengths` will result in the following values:

Call	Value Returned
`stringLengths(array1)`	`{ 3, 3, 7, 3, 1, 3, 5 }`
`stringLengths(array2)`	`{ 1, 4, 3, 3 }`
`stringLengths(array3)`	`{ 21 }`
`stringLengths(array4)`	`{ 7, 7, 10 }`

(You will get the +1 point as long as you write anything that appears to have taken more than a moment to write.)

CSE 142 SPRING 2008 FINAL EXAM SOLUTIONS

1.

Expression	Value
`!(4 > 3.5) || ((6 < 3) && (4 < 9))`	`false`
`99 % 100 + 10 / 4 / 1.0`	`101.0`
`"8" + 3 * 2 + (3 * 25) + 3 * 10 + 3 * 3`	`"8675309"`
`6 / 2 + (14 < 3)`	`error`
`-1 + 11 % 7 * 2 + 1000 + 110 * 3`	`1337`

2.

Array	Final Contents of Array
`int[] a1 = {8, 9};` `mystery(a1);`	`{17, 9}`
`int[] a2 = {14, 7, 1};` `mystery(a2);`	`{22, 8, 1}`
`int[] a3 = {7, 1, 3, 2, 0, 4};` `mystery(a3);`	`{17, 10, 9, 6, 4, 4}`
`int[] a4 = {10, 8, 9, 5, 6};` `mystery(a4);`	`{38, 28, 20, 11, 6}`
`int[] a5 = {8, 10, 8, 6, 4, 2};` `mystery(a5);`	`{38, 30, 20, 12, 6, 2}`

3.

```
Go
```

4.

```
one fish, two fish
blue red, fish fish
red fish, blue fish
```

5.

```
AcmeLabs A
Brain B
One is a genius, the other's insane.

Brain A
Brain B
Pinky!  Are you pondering what I'm pondering?

Pinky A
World B
I think so, Brain, but...

Brain A
World B
Try to take over the world!
```

6.

```
public static boolean isSumArray(int[] array) {
   if (array.length % 3 != 0) {
      return false;
   }

   for (int i = 2; i < array.length; i += 3) {
      if (array[i] != array[i-1] + array[i-2]) {
         return false;
      }
   }

   return true;
}

public static boolean isSumArray(int[] array) {
   if (array.length % 3 != 0) {
      return false;
   }

   for (int i = 0; i < array.length; i += 3) {
      if (array[i+2] != array[i+1] + array[i]) {
         return false;
      }
   }

   return true;
}
```

7. Two possible solutions are shown.

```
public static void printWinner(Scanner input) {
   int max = 0;
   String maxName = "";

   while (input.hasNext()) {
      String name = input.next();
      int counter = 0;
      while (input.hasNextInt()) {
         counter += input.nextInt();
      }
      if (counter > max && counter <= 21) {
         max = counter;
         maxName = name;
      }
   }

   if (max > 0) {
      System.out.println(maxName + " is the winner!");
   } else {
      System.out.println("Everyone busted!");
   }
}
```

```
public static void printWinner(Scanner input) {
   int max = 0;
   String output = "Everyone busted!";

   while (input.hasNext()) {
      String name = input.next();
      int counter = 0;
      while (input.hasNextInt()) {
         counter += input.nextInt();
      }
      if (counter > max && counter <= 21) {
         max = counter;
         output = name + " is the winner!";
      }
   }
   System.out.println(output);
}
```

8. Four possible solutions are shown.

```
public class Dog extends Critter {
    private int moves = 0;
    private int max = 1;

    public Direction getMove() {
        moves++;
        if (moves > max) {
            moves = 1;
            max += 2;
        }
        if (moves <= max / 2) {
            return Direction.EAST;
        } else {
            return Direction.SOUTH;
        }
    }
}
```

```
public class Dog extends Critter {
    private int moves = 0;
    private int max = 1;
    private Direction dir = Direction.EAST;

    public Direction getMove() {
        if (moves == max) {
            if (dir == Direction.EAST) {
                dir = Direction.SOUTH;
            } else {
                dir = Direction.EAST;
                max++;
            }
            moves = 0;
        }

        moves++;
        return dir;
    }
}
```

```
public class Dog extends Critter {
    private int moves;
    private int max;
    private Direction dir;

    public Dog() {
        moves = 0;
        max = 1;
        dir = Direction.EAST;
    }

    public Direction getMove() {
        if (moves == max) {
            if (dir == Direction.EAST) {
                dir = Direction.SOUTH;
            } else {
                dir = Direction.EAST;
                max++;
            }
            moves = 0;
        }

        moves++;
        return dir;
    }
}
```

```
public class Dog extends Critter {
    private int max = 1;
    private int east = 0;
    private int south = 0;

    public Direction getMove() {
        if (east < max) {
            east++;
            return Direction.EAST;
        } else if (south < max) {
            south++;
            return Direction.SOUTH;
        } else {
            east = 1;
            south = 0;
            max++;
            return Direction.EAST;
        }
    }
}
```

9.

```
public static int[] stringLengths(String[] strings) {
   int[] lengths = new int[strings.length];
   for (int i = 0; i < strings.length; i++) {
      lengths[i] = strings[i].length();
   }
   return lengths;
}
```