CIS 110 Fall 2016 — Introduction to Computer Programming
13 Oct 2016 — Midterm Exam
Answer Key

1.) The Easy One (1 point total)

Check cover sheet for name, recitation #, PennKey, and signature.

2.) Value Judgments (8 points total)

Fill in the data type and final value of the variable `a`. (Assume `a` is always declared with the most appropriate data type.) Write "**CE**" as the data type if the statements will cause compiler error, or "**RE**" if they will cause a run-time error. Give the reason for the error in the third column, but don't worry about giving the precise error message as Java would print it. The first row has been filled in for you.

	Type of `a`	Value of `a`/Error explanation
`_____ a = 9;` `a++;`	`int`	`10`
`int x = 100;` `_____ a = x.length;`	`CE`	`x` doesn't have an `x.length` field
`int x = 5;` `_____ a = x / 2;`	`int` or `double`	`2`

`int x = 1;`
`int y = 1;`
`_____ a = (x = y);`
	CE	Use `==` for comparisons

`String a = "CIS";` `_____ a += 1 + 1 + 0;`	`String`	`"CIS110"`
`_____ a = Integer.parseInt('2');`	CE	`parseInt()` takes a `String` argument, not a `char`
`String s = "Will Adam Brown?";` `_____ a = s.charAt(16);`	`RE`	`StringIndexOutOfBoundsException`
`_____ a = 'a' < 'b';`	boolean	`true`
`double x = 2.0;` `_____ a = x / 0;`	`double`	`Infinity`

3.) A Bug's Life (12 points total)

The following function is supposed to calculate the highest possible sum of any two entries in an array of positive integers. For example, given the array `{4, 5, 3, 2, 8}`, it should return 13 (the sum $5 + 8$). Unfortunately it is infested with bugs. Write the corrected code in the space below. Make sure your corrected function is reasonably well indented, is syntactically correct, and **is the same number of lines as the buggy version** because we will be grading your answer line by line. (You may place open curly braces on their own line, if you prefer, and you may add blank lines. But do not make any other changes to the total number of lines.)

```
 1. public static int highestSum(int[] arr) {
 2.     int maxSum = 0;
 3.     for (i = 0; i < arr.length; i++) {
 4.         for (int j = i + 1; j < < arr.length(); i++) {
 5.             int sum == arr[i] + arrj[j];
 6.             if (sum < maxSum) {
 7.                 int maxSum = sum
 8.             }
 9.         }
10.         return maxSum;
11.     }
12. }
```

```
 1. public static int highestSum(int[] arr) {
 2.     int maxSum = 0;
 3.     for (int i = 0; i < arr.length; i++) {
 4.         for (int j = i + 1; j < arr.length; j++) {
 5.             int sum = arr[i] + arr[j];
 6.             if (sum > maxSum) {
 7.                 maxSum = sum;
 8.             }
 9.         }
12.     }
11      return maxSum;
12. }
```

4.) I Scream, "Ice Cream" (15 points total)

Study the following program, then answer the questions.

```
public class IScreamIceCream {
    public static void toppings(int y) {
        if (y <  2) return;
        System.out.println("sprinkles");
        toppings(y - 1);
    }

    public static String flavor(String f, String p, int x) {
        if (x % 2 == 0) {
            System.out.println(x + " scoop " + f);
            if (x % 4 == 0) {
                toppings(x - 1);
                return flavor(p, f, x + 2);
            } else {
                return flavor(f, p, x - 1);
            }
        } else {
            System.out.println(x + " scoop " + p);
            if (x <= 2) return "Coming right up!";
            else        return flavor(f, p, x - 2);
        }
    }

    public static void main(String[] args) {
        int num = Integer.parseInt(args[0]);
        System.out.println(flavor("vanilla", "chocolate", num));
    }
}
```

Give the **exact** output of each of the two following commands:

```
java IScreamIceCream 3
3 scoop chocolate
1 scoop chocolate
Coming right up!
```

```
java IScreamIceCream 4
4 scoop vanilla
sprinkles
sprinkles
6 scoop chocolate
5 scoop vanilla
3 scoop vanilla
1 scoop vanilla
Coming right up!
```

5.) Drop It Like It's Hot (14 points total)

The following function is supposed to compute your average homework percentage according to the CIS 110 homework policy: given an array of homework percentages between 0 and 100%, it drops the lowest score as long as there are at least two scores in the array **and** the lowest score is at least 30%, then returns the average. Unfortunately, the TAs who were writing the function ran off screaming after the limo of Snoop Dogg and Pharrell Williams before finishing the code. Fill in the blanks to complete the function.

```
public static double homeworkGrade(double[] scores) {
    if (scores == null) return 0; // error checking

    int numScores = scores.length;
    if (numScores == 0) return 0;         // error checking

    double lowestScore = 100.0;
    double sum = 0.0;

    for (int i = 0; i < numScores; i++) {
        if (scores[i] < lowestScore) {
            lowestScore = scores[i];
        }

        sum += scores[i];
    }

    if ((lowestScore < 30) || (numScores == 1)) {
        return sum / numScores;
    }

    return (sum - lowestScore) / (numScores - 1);
}
```

6.) Get it on the Green (25 points total)

An enterprising alum of Horton's School of Funny Business with a poor understanding of statistics is planning a weighted roulette ball that is more likely to land on green spaces for his upcoming gambling and golf resort. He believes that the function described below will properly verify that the ball will land on green more often than a normal ball would. Since he's offered to comp you a week in a suite in his new casino, you oblige. Write a function **`roulette`** that takes an integer argument **`n`**, returns a string, and implements the following behavior:

- Use `Math.random()` to simulate the outcome of spinning a roulette wheel with 18 red spaces, 18 black spaces, and 2 green spaces **`n`** times, and count the number of times the ball lands in a green space.
- The ball is twice is likely to land in any given green space than in any given red or black space because of the way it is weighted (which means there is a 1-in-10 chance the ball will land on green);
- If the ball lands on a green space more often than an unweighted ball is expected to over the course of the simulations, return, **`"Good to go!"`**. (An unweighted ball is equally likely to land in any of the squares, so it should land on green about 1/19th of the time.) Otherwise return the number of times the ball landed on green, represented as a string.

Do not write the `class` statement, just the function that would be contained within the class. Your program must be syntactically correct, and you must make a reasonable attempt to indent your code. You do not need to write comments, but you are allowed to if that makes it easier for you. You are also welcome to use more than one color of pen/pencil (but not red), if that help you.

```
public static String roulette(int n) {
    int count = 0;
    for (int i = 0; i < n; i++)
        if (Math.random() < 0.1)
            count++;

    if (count > (n / 19.0)) return "Good to go!";
    else                    return "" + count;
}
```