This document contains Practice Questions for CMSC B113 Exam #2 in Fall 2023.

This is not a "sample exam"! It is merely a collection of questions that have been used in past offerings of CMSC B113 that cover the same topics that will be on your exam.

Question numbers below are non-consecutive

Question #2

Fill in the blanks in the **averageOfMaxValues** method so that it will use the **findMax** and **findAverage** methods to calculate the average of the maximum values in each array in A; note that A is a two-dimensional array, i.e. an array of arrays.

For instance, if arr2d = { { 3, 5, 6 } , { 2, 8, 1, 4 } , { 9, 3, 4, 5 } }, then:

- the max of arr2d[0] is 6
- the max of arr2d[1] is 8
- the max of arr2d[2] is 9
- their average is (6 + 8 + 9) / 3 = 7.667.

```
   public static int findMax(int[] arr) {
        if (arr.length == 0) return 0;
        int max = arr[0];
        for (int i=0; i<arr.length; i++) {
            if (arr[i] > max) {
                 max = arr[i];
            }
        }
        return max;
    }

    public static double findAverage(int[] arr) {
        int sum = 0;
        for (int i=0; i<arr.length; i++) {
            sum += arr[i];
        return ((double)sum) / arr.length;
    }

    public static double averageOfMaxValues(int[][] arr2d) {
          // this array should hold the max value of each array in arr2d
1:        int[] maxValues = _____________________;
2:        for (int i = 0; i < arr2d.length; i++) {
3:             maxValues[i] = _____________________;
4:        }
5:        return _____________________;
    }
```

Question #5

Write a Java method **stddev()** that calculates the standard deviation of an array ***a***. The standard deviation is defined as follows:

$$\sigma = \sqrt{((a_0 - \mu)^2 + \ (a_1 - \mu)^2 + \ldots + (a_{n-1} - \mu)^2)/(n-1)}$$

Where μ is the average of $(a_0, \ldots a_{n-1})$.

Implement the **stddev()** method so that it uses the **average()** method that is defined below.

```
public static double average (double[] arr) {
   double sum = 0;
   for (int i=0; i<arr.length; i++) {
        sum += arr[i];
   }
   return sum / arr.length;
}

public static double stddev (double[] aarr) {









}
```

Question #6

The following code is attempting to use recursion to compute the summation 1+2+3+…n.

Implement the `sumHelper()` method so that it takes two arguments -- n and the accumulating `sum` -- and is defined as follows:

- sumHelper(0, x) = x
- sumHelper(n, x) = sumHelper(n-1, x+n)

```
public static int sum (int num) {
   return sumHelper(num, 0);
}

// Implement the sumHelper method here
```

Question #7

Write a complete Java program named **Print** that takes 2 arguments from the command line, call those two arguments “N” and “text”, and then prints **text** to standard-out **N** times.

For instance, if the command line contains the following:

```
java Print 3 dog
```

the the program should produce the output:

```
dog
dog
dog
```

And if the file **words.txt** contains the following:

```
java Print 4 cat
```

Then the program should produce the output:

```
cat
cat
cat
cat
```

Your program can assume that there are at least two command line arguments and that the the first command line argument can be converted to an int. The program should not print anything if the int is zero or negative.

Question #8

Assume that the following code compiles:

```
public static void main(String[] args) {
      double m = StdIn.readDouble();
      int x = (int)fun(m * 2);
}
```

Assuming that the **fun** method is defined in the same class as the code above, which of the following are possible legal implementations of the **fun** method? Select all that apply:

A.

```
public static void fun(double a) {
      StdOut.println("fun?" + a);
}
```

B.

```
public static double fun(double a) {
      return a * a;
}
```

C.

```
public static int fun(int a) {
      return -1 * a;
}
```

D.

```
public static int fun(double a) {
      return (int)Math.round(a);
}
```