ASSIGNMENT 2

COMP-202, Winter 2016, All Sections

Due: Monday, February 22, 2016 (23:59)

Please read the entire pdf before starting.

You must do this assignment individually and, unless otherwise specified, you must follow all the general instructions and regulations for assignments. Graders have the discretion to deduct up to 15% of the value of this assignment for deviations from the general instructions and regulations. These regulations are posted on the course website. Be sure to read them before starting.

Question 1:	30 points
Question 2:	35 points
Question 3:	35 points
	100 points total

It is very important that you follow the directions as closely as possible. The directions, while perhaps tedious, are designed to make it as easy as possible for the TAs to mark the assignments by letting them run your assignment, in some cases through automated tests. While these tests will never be used to determine your entire grade, they speed up the process significantly, which allows the TAs to provide better feedback and not waste time on administrative details. Plus, if the TA is in a good mood while he or she is grading, then that increases the chance of them giving out partial marks. :)

Assignment

Part 1 (0 points): Warm-up

Do **NOT** *submit this part, as it will not be graded. However, doing these exercises might help you to do the second part of the assignment, which will be graded. If you have difficulties with the questions of Part 1, then we suggest that you consult the TAs during their office hours; they can help you and work with you through the warm-up questions.*

Warm-up Question 1 (0 points)

Create a file called `Counting.java`, and in this file, declare a class called `Counting`. This program takes as input from the user a positive integer and counts up until that number. eg:

```
run Counting 10
I am counting until 10: 1 2 3 4 5 6 7 8 9 10
```

Warm-up Question 2 (0 points)

For this question you have to generalize the last question. The user will give you the number they want the computer to count up to and the step size by which it will do so.

```
run Counting 25 3
I am counting to 25 with a step size of 3:
1 4 7 10 13 16 19 21 24
```

Warm-up Question 3 (0 points)

Write a method `reverseString` which takes as input a `String` and returns the string in reverse order. For example if the input `String` is "Comp 202 is awesome" the result should be "emosewa si 202 pmoC"

Hint: Use a new `String` called `reverse` and initially store into it the empty `String`. Then read the input `String` in reverse by using the method .charAt(int i) to get a specific element.

Warm-up Question 4 (0 points)

A prime number is a positive number whose only even divisors are 1 and itself. Write a method `isPrime` that takes as input an integer `n` and returns a `boolean` representing whether `n` is prime or not. Make sure to handled cases where the integer is negative. (Your method should return false in these cases.)

Warm-up Question 5 (0 points)

Write a method `firstPrimeNumbers` which takes as input an `int n` and returns an `int[]` . The `int[]` should contain the first `n` prime numbers.

Warm-up Question 6 (0 points)

This program prints the outline of a square made up of # signs. It should take as input the length of the sides in number of #s. Using two loops, you should be able to draw the outline of a square as follows.

```
##########
#        #
#        #
#        #
#        #
#        #
#        #
#        #
#        #
##########
```

N.B. It is normal that the square does not appear to be a perfect square on screen as the width and the length of the characters are not equal.

Part 2

The questions in this part of the assignment will be graded.

Question 1: Drunken Sailor (30 points)

Consider a sailor who gets out of a bar completely drunk, in a city where all blocks are in perfect squares with size 1×1. The drunken sailor walks from one corner of a block to the next. When in a corner, he forgets where he came from, and where he was going. He takes a direction at random, either north, east, south, or west, with an equal probability of 25%. Then in the following corner he again forgets where he was going, but keeps on walking by following a random direction. You will write a program that takes an integer input N, and simulates the motion of the drunken sailor for N steps. After each step, print the location of the sailor. You can treat the starting point as $(0, 0)$. Also, print the final distance from the origin and the furthest point[1] he has reached. Example:

```
>run DrunkenSailor 10
(0, -1)
(0, 0)
(0, 1)
(0, 2)
(-1, 2)
(-2, 2)
(-2, 1)
(-1, 1)
(-2, 1)
```

[1]We use the Euclidean distance here. In particular, the distance between (x, y) and the origin is $\sqrt{x^2 + y^2}$. The furthest point is the point with the largest Euclidean distance from the origin.

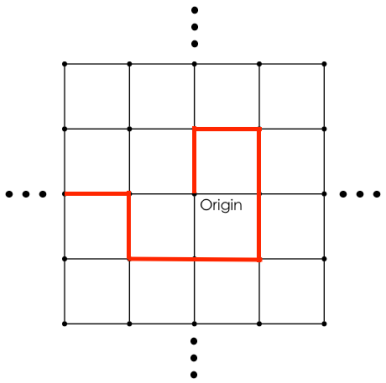

Figure 1: An example of an 8-step walk by the sailor.

```
(-3, 1)
Final distance = 3.1622776601683795
Furthest point = (-3,1)
```

Question 2: Binary Converter (35 points)

You will write a program that converts binary numbers to decimal numbers, where the input binary number is a `String`. This program will first verify whether the input only contains '0's and '1's.[2] If the user enters letters or other invalid inputs, the program prints "The input is not binary!". When the input is valid, the program will then convert the binary number to a decimal number.

Valid input - In order to check if the input is valid your program should check if it is only composed of zeros and ones. More specifically, you should scan the input string using a for or while loop to make sure it only contains '0' or '1' characters. As soon as you find a character that is not '0' or '1', the process should stop. If the process reaches the end of the input string then we have a valid input. You may ignore the case where the input string is empty, as well as any extra '0's at the start of the string.

Converting - At this point we assume that the input string is valid. The binary conversion should work as follows. For each digit in the binary number, if the digit is '1' you should add the corresponding decimal value (2^0 for the rightmost digit, 2^1 for the next digits to the left, 2^2 for the next one, and so on) to a variable that will hold the final result to be printed. This **must** be accomplished by using a loop.

Example:

```
> run Conversion 123456789
The input is not binary!
>run Conversion 101010
The binary number 101010 is 42 in base 10.
```

Question 3: Arrays (35 points)

3a) Find the Missing Numbers

Write a method `findMissingNum` that takes two inputs: a positive integer n and an integer array which contains numbers from 1 to n. Some integer numbers in this range are missing in the array, you need to write a Java program to find those missing numbers in the array and print them out. If no number is missing, print out "`No number is missing!`".

Example:

[2] **IMPORTANT:** You are allowed to use loops, the `length()`, `charAt(int index)` and `equals(String other)` methods of `String`s for this question.

`findMissingNum(6, array)` while array = $\{2, 3, 5, 1, 2, 3\}$ should print out $4, 6$.
`findMissingNum(5, array)` while array = $\{1, 5, 1, 3, 4, 2\}$ should print out "`No number is missing!`".
`findMissingNum(4, array)` while array is empty should print out $1, 2, 3, 4$.

3b) Find the Most Frequent Numbers

Implement a method `findMostFrequent` that again takes a positive integer n and an integer array which contains numbers from 1 to n. Print out all the numbers that have the highest frequency.

Example:

`findMissingNum(6, array)` while array = $\{2, 3, 5, 1, 2, 3\}$ should print out $2, 3$.
`findMissingNum(5, array)` while array = $\{1, 5, 1, 3, 4, 2\}$ should print out 1.
`findMissingNum(4, array)` while array is should print out "`The array is empty!`".

What To Submit

You have to submit one zip file that contains all your files to myCourses - Assignment 2. If you do not know how to zip files, please enquire that information from any search engine or friends. Google might be your best friend with this, and for a lot of different little problems as well.

`DrunkenSailor.java` - Java code
`Conversion.java` - Java code
`arrays.java` - Java code

`Confession.txt` (optional) In this file, you can tell the TA about any issues you ran into doing this assignment. If you point out an error that you know occurs in your problem, it may lead the TA to give you more partial credit. On the other hand, it also may lead the TA to notice something that otherwise he or she would not.