CS101- Algorithms and Programming I

Lab 06

Lab Objectives: Static methods.

- ❏ For all labs in CS 101, your solutions must conform to these CS101 style guidelines (rules!)
- ❏ Create a Lab06 workspace (i.e. the folder H:\private\cs101\lab06). This assignment has parts a, b, c, and d, each of which should be placed in a separate project within the same Lab06 workspace.

---

a. For this lab assignment, we will start writing very basic static methods to warm you up. Create a new project Lab06a. Write a Java program that will utilize two static methods ***toBinary()*** and ***toDecimal().***

toBinary() method takes a decimal integer value as a parameter and converts it into the corresponding binary String and returns it.

toDecimal() takes a `String base2` and converts it into its decimal --base10-- equivalent, as an int and returns it. For example, if base2 is "110", it should return 6, whereas if it was "1011", it should return 11. *You should be able to do this without using Math class methods, and without computing the power of the most significant digit.*

In your program, you should input an integer ***n*** from the user. *For now you don't have to worry about negative input. I recommend you to search about it. What if input is negative? How do you represent a negative decimal value in binary?* After taking the input ***n*** you should convert it into the binary number. Now you have a binary representation of a number. Using the appropriate method convert it into decimal. Normally, we should get back the number ***n***. Verify that this is the case. You may write auxiliary methods if needed. See the sample runs. Note that you are NOT allowed to use Math or Integer class methods. You may write your own version of those methods, if necessary.

Sample Runs:

```
> run Lab06a
Enter an integer n:  8
The binary representation is 1000
The decimal value is: 8
They are equal.
> run Lab06a
Enter an integer n:  31
The binary representation is 11111
The decimal value is: 31
They are equal.
>
```

b. Create a new project Lab06b. Write a Java program that utilizes a static method you will write soon. Your program should input an integer ***n*** from the user which will be passed to the static method ***generateTheString()*** as a parameter. *Repeatedly ask the user an integer n until the user enters 0. And print the returned statement of the method.* You still have to validate your input in case of ***n*** is negative. In that case, print a warning message and continue taking input.

   The static method ***generateTheString()*** should return a string with ***n*** characters such that each character in such string occurs **an odd number of times**. The returned string must contain only lowercase English letters. If there are multiple valid strings, return **any** of them.

   Example:

   Input: n = 4
   Output: "aaab"
   Explanation: "aaab" is a valid string since the character 'a' occurs three times and the character 'b' occurs once. Note that there are many other valid strings such as "ohhh" and "love".

   Note that there is no particular constraint to construct such a string so there are zillions of ways to do it. Therefore we do not give our sample run. Choose your own way!

c. An English text needs to be encrypted using the following encryption scheme. First, the spaces are removed from the text. Let L be the length of this text. Then, characters are written into a grid, whose rows and columns have the following constraints:

$$\lfloor \sqrt{L} \rfloor \leq row \leq column \leq \lceil \sqrt{L} \rceil,$$

   where $\lfloor x \rfloor$ is floor function whereas, $\lceil x \rceil$ is ceil function.

   For example, the sentence

   s = "*if man was meant to stay on the ground god would have given us roots*"

   after removing spaces is 54 characters long. $\sqrt{54}$ is between 7 and 8, so it is written in the form of a grid with 7 rows and 8 columns.

```
ifmanwas
meanttos
tayonthe
groundgo
dwouldha
vegivenu
sroots
```

The encoded message is obtained by displaying the characters in a column, inserting a space, and then displaying the next column and inserting a space, and so on. For example, the encoded message for the above rectangle is:

```
imtgdvs fearwer mayoogo anouuio ntnnlvt wttddes aohghn sseoau
```

You must ensure that $rows \times columns \geq L$. Also If multiple grids satisfy the above conditions, choose the one with the minimum area, i.e. $rows \times columns$.

Create a new project Lab06c. Write a Java program that utilizes a static method ***encryption()*** that takes a String as a parameter and do the encryption that is explained above and return the encoded message. In your main method, take an input from the user and use it properly. Print the encoded message. See the sample run below.

Now you have the encryption method. As an exercise, try to write a decryption method for such encrypted messages. (This part is not included in the lab assignment, but it is highly recommended for you to solve.)

Sample Runs:

```
> run Lab06c
Please enter a message: corona corona corona

Encoded message: cano ocan roca oro nor
> run Lab06c
Please enter a message: stay home

Encoded message: sym the ao
```

d. Create a new project Lab06d. Write a Java program that simulates a dice rolling game. The program inputs the total money from the user (integer). At each time, the user makes a prediction of "even" or "odd" or “extreme” for the sum of face values of the rolled dice, and bets some money in that prediction. The category “extreme” corresponds to both dice being 1 or 6. The case “extreme” is a risky one. Because if the program rolls dice whose category “extreme” is predicted by the user, he/she wins 100% of his/her bet. If not, he/she loses 100% of his/her bet. If the program rolls dice whose category (even or odd) is predicted by the user, he/she wins 50% of his/her bet. If the prediction is wrong, he/she loses 25% of his/her bet. The bet must be smaller than or equal to the current total money. If not, print "Your money is not enough!" and ask the bet from the user again. The program terminates when the user enters 0 or the user does not have any more money to bet.

   Your program must include a static method that rolls the dice, computes and returns the gain according to the sum of the face values of the dice, given the prediction and the bet as parameters.

   Sample run:

```
> run Lab06d
Enter your total money: 90
Enter prediction (1 for ODD, 2 for EVEN, 3 for EXTREME): 1
Enter the amount of money you want to bet: 60
The sum of the dice is 5
Your total money is 120.0
**************************************
Enter prediction (1 for ODD, 2 for EVEN, 3 for EXTREME): 2
Enter the amount of money you want to bet: 130
Your money is not enough!
Enter the amount of money you want to bet: 100
The sum of the dice is 7
Your total money is 95.0
**************************************
Enter prediction (1 for ODD, 2 for EVEN, 3 for EXTREME): 3
Enter the amount of money you want to bet: 75
The sum of the dice is 9
Your total money is 38.75
**************************************
Enter prediction (1 for ODD, 2 for EVEN, 3 for EXTREME): 3
Enter the amount of money you want to bet: 20
The sum of the dice is 6
Your total money is 23.75
**************************************
Enter prediction (1 for ODD, 2 for EVEN, 3 for EXTREME): 0
>
```