ASSIGNMENT 2

COMP-202, Winter 2019

Due: Monday, February 18th, 2019 (23:59)

Please read the entire PDF before starting. You must do this assignment individually.

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. :)

Up to 30% can be removed for bad indentation of your code as well as omitting comments, or poor coding structure.

To get full marks, you must:

- Follow all directions below
  - In particular, make sure that all classes and method names are **spelled and capitalized exactly** as described in this document. Otherwise, you will receive a **50% penalty**.
- Make sure that your code compiles
  - **Non-compiling code will receive a 0**.
- Write your name and student ID as a comment in all .java files you hand in
- Indent your code properly
- Name your variables appropriately
  - The purpose of each variable should be obvious from the name
- Comment your work
  - A comment every line is not needed, but there should be enough comments to fully understand your program

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. You are responsible for knowing all of the material in these questions.*

Warm-up Question 1 (0 points)

Write a method `swap` which takes as input two int values `x` and `y`. Your method should do 3 things:

1. Print the value of `x` and `y`
2. Swap the values of the variables `x` and `y`, so that whatever was in `x` is now in `y` and whatever was in `y` is now in `x`
3. Print the value of `x` and `y` again.

For example, if your method is called as follows: `swap(3,4)` the effect of calling your method should be the following printing

```
inside swap:  x is:3 y is:4
inside swap:  x is:4 y is:3
```

Warm-up Question 2 (0 points)

Consider the program you have just written. Create two integer variables in the main method. Call them `x` and `y`. Assign values to them and call the swap method you wrote in the previous part using `x` and `y` as input parameters.

After calling the `swap()` method—inside the main method— print the values of `x` and `y`. Are they different than before? Why or why not?

Warm-up Question 3 (0 points)

Write a method that takes three integers `x`, `y`, and `z` as input. This method returns `true` if `z` is equal to 3 or if `z` is equal to the sum of `x` and `y`, and `false` otherwise.

Warm-up Question 4 (0 points)

Let's write a method incrementally:

1. Start by writing a method called `getRandomNumber` that takes no inputs, and returns a random `double` between 0 (included) and 10 (excluded).
2. Now, modify it so that it returns a random `int` between 0 and 10 (still excluded).
3. Finally, let the method take two integers `min` and `max` as inputs, and return a random integer greater than or equal to `min` and less than `max`.

Warm-up Question 5 (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 (using `args`) 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 6 (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 22 25
```

Warm-up Question 7 (0 points)

Write a method `getFirstHalf()` that takes as input a String and returns a String composed of the first half of the characters from the specified String. For example, `getFirstHalf("cucumber")` returns the String `"cucu"`, while `getFirstHalf("apple")` returns the String `"ap"` (thus, if the number of characters is odd, you should round down).

Warm-up Question 8 (0 points)

Write a method `alphaString()` which takes two Strings as input and returns the String between the two that comes first in the alphabet. For example, `alphaString("banana", "apple")` returns the String `"apple"`, while `alphaString("snake", "squirrel")` returns the String `"snake"`.

Warm-up Question 9 (0 points)

Write a method `replaceAll()` which takes as input a String and two characters. The method returns the String composed by the same characters of the given String where all occurrences of the first given character are replaces by the second given character. For example, `replaceAll("squirrel", 'r' , 's')` returns the String `"squissel"`, while `replaceAll("squirrel", 't', 'a')` returns the String `"squirrel"`.

Part 2

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

Throughout this assignment you are allowed to use everything we have learned in class up to and including loops (`while` and `for`). This does not mean however that you are allowed to change any of the headers of the methods described below. You need to make sure to follow precisely the instructions provided.

For this assignment, you are **NOT** allowed to use the following:

- Any other method from the `String` class beside `length()`, `charAt()`, `equals()`, `equalsIgnoreCase()`, `toLowerCase()`, and `toUpperCase()`.In particular, you are **NOT** allowed to use `substring()`, `split()`, `contains()`, `indexOf()`, or `lastIndexOf()`.
- Any other class beside `Math`. In particular, this means that no import statements are allowed.

Question 1: Bunco (40 points)

Bunco is a dice game composed by six rounds. To play, all that is required is three dice. Normally Bunco is played with 12 or more players divided into groups of four, but, for the purpose of this assignment, we'll be looking at a simplified version of Bunco played between 2 players. The players will take turns rolling three dice trying to score points. At the end of the 6 rounds, the player with the highest number of points wins then game.

The goal of this question is to write several methods in order to create a program that simulates two players playing a game of Bunco. All the code for this question must be placed in a file named `Bunco.java`.

A round of Bunco. As explained above, Bunco is a game of rounds. For the purpose of this question we will assume that each of the two players is allowed to roll the three dice only once per round. Points are awarded as such:

- 21 points if all three dice match the current round number (this is a "Bunco").
- 5 points if all three dice match each other, but not the current round number (this is a "mini Bunco")
- 1 point for each die matching the current round number.

Each game consists of six rounds, progressing in order from one to six. The player with the highest points at the end of the game wins.

Now that we know how the game works, let's look at the methods we need in order to simulate two players playing the game of Bunco.

1a. A method to simulate a dice roll

In a game of Bunco, each player rolls three six-sided dice. Write a method called `diceRoll()` that simulates the roll of one six-sided dice. Such method takes no inputs, and it returns an integer between 1 and 6 (both included). Your method must use `Math.random()` in order to achieve the desired result. If you have any doubts on how to write the method, make sure you first understand the warm-up Question 4, where you are asked to build a method called `getRandomNumbers()`.

1b. A method to compute the score of a player

Write a method `getScore()` that computes the score of a player after he/she takes his/her turn rolling the dice. This method takes as input three integer values that corresponds to the numbers the player has rolled, as well as a fourth integers indicating the current round number. It then returns an integer indicating the score achieved. Remember that, the points for each round of Bunco are awarded as described in the paragraph above titled "A round of Bunco".

1c. A method to simulate one round of Bunco

Write a method `playOneRound()` that simulates one player playing one round of Bunco. This method takes as input a String with the name of the player rolling the dice, and an integer indicating the current round that has been played. The method returns an integer representing the score obtained by such player. The method must use `diceRoll()` to simulate all the dice rolls necessary, and `getScore()` to retrieve the points obtained. The method also prints the result of the dice roll as well as the score obtained by the player. For example, if the method is called with inputs `"Jikun"` and 2, then it should print something similar to the following:

```
Jikun rolled 2 5 2 and scored 2 points
```

In this case, the method also returns the value 2. On the other hand, if the method is called with input `"Lita"` and 2, then it should print something similar to the following:

```
Lita rolled 1 6 4 and scored 0 points
```

In this case, the method returns the value 0. Note that, if the input indicating the round number is not an integer between 1 and 6 (both included), then the method should display an error message indicating that the input was not valid and return the value -1.

1d. A method to simulate a game of Bunco

Finally, write a method `playBunco()` that simulates a game of Bunco between two players. This method takes two Strings as input indicating the names of the players. The method does not return any value.

The method simulates the two players playing rounds of Bunco. While doing so, it keeps track of the total score of each player and at the end of each round it displays it on the screen. (Hint: think about how to use a couple of extra variables to do so. Where should these variables be declared? Where and how should they be updated?) The method simulates 6 rounds of Bunco and then determines the winner of the game. A message indicating the winner should then be displayed. If there is a tie, the method should print out a statement declaring that there was a tie.

To get full marks the method must use `playOneRound()`. Moreover, there should not be repetitions in your code (for instance, you cannot repeat your code 6 times in order to simulate 6 rounds).

1g. Setting up the main method (Optional - No extra marks)

Set up the main method so that the program receives two inputs of type String via command line arguments indicating the names of the players. From the main method, call `playBunco()` with the appropriate inputs in order to simulate the two players playing Bunco. In the next pages you can find a couple of examples of how your program should run. (Note that due to the randomness of the dice rolls, the output will change each time you run your program)

```
> run Bunco Shayan Yuzhou
ROUND #:1

Shayan rolled 1 6 1 and scored 2 points
Yuzhou rolled 1 6 3 and scored 1 points

Shayan's score after round 1 is 2
Yuzhou's score after round 1 is 1
---------------------------------
ROUND #:2

Shayan rolled 1 6 2 and scored 1 points
Yuzhou rolled 4 6 2 and scored 1 points

Shayan's score after round 2 is 3
Yuzhou's score after round 2 is 2
---------------------------------
ROUND #:3

Shayan rolled 6 3 3 and scored 2 points
Yuzhou rolled 6 5 3 and scored 1 points

Shayan's score after round 3 is 5
Yuzhou's score after round 3 is 3
---------------------------------
ROUND #:4

Shayan rolled 3 1 5 and scored 0 points
Yuzhou rolled 6 6 2 and scored 0 points

Shayan's score after round 4 is 5
Yuzhou's score after round 4 is 3
---------------------------------
ROUND #:5

Shayan rolled 2 1 5 and scored 1 points
Yuzhou rolled 3 4 1 and scored 0 points

Shayan's score after round 5 is 6
Yuzhou's score after round 5 is 3
---------------------------------
ROUND #:6

Shayan rolled 3 3 3 and scored 5 points
Yuzhou rolled 3 2 1 and scored 0 points

Shayan's score after round 6 is 11
Yuzhou's score after round 6 is 3
---------------------------------

Shayan wins!
```

```
> run Bunco Maria Christopher
ROUND #:1

Maria rolled 6 1 4 and scored 1 points
Christopher rolled 3 1 4 and scored 1 points

Maria's score after round 1 is 1
Christopher's score after round 1 is 1
--------------------------------
ROUND #:2

Maria rolled 2 2 6 and scored 2 points
Christopher rolled 1 2 6 and scored 1 points

Maria's score after round 2 is 3
Christopher's score after round 2 is 2
--------------------------------
ROUND #:3

Maria rolled 5 4 1 and scored 0 points
Christopher rolled 1 4 1 and scored 0 points

Maria's score after round 3 is 3
Christopher's score after round 3 is 2
--------------------------------
ROUND #:4

Maria rolled 4 4 4 and scored 21 points
Christopher rolled 1 3 5 and scored 0 points

Maria's score after round 4 is 24
Christopher's score after round 4 is 2
--------------------------------
ROUND #:5

Maria rolled 5 3 2 and scored 1 points
Christopher rolled 4 6 2 and scored 0 points

Maria's score after round 5 is 25
Christopher's score after round 5 is 2
--------------------------------
ROUND #:6

Maria rolled 3 1 1 and scored 0 points
Christopher rolled 5 4 6 and scored 1 points

Maria's score after round 6 is 25
Christopher's score after round 6 is 3
--------------------------------

Maria wins!
```

Question 2: Email Validation (60 points)

For this question, you will write a Java program that helps validate email addresses. Email addresses are often requested as input to websites as a way to validate the identity of the user. To assure that the email provided is actually good, a combination of various validation techniques is required. For the purpose of this question, we will focus on checking whether or not a given string represents a syntactically correct email address.

A valid email address consists of a prefix, an '@' symbol, and an email domain. Both the prefix and the domain must be written in acceptable formats. For instance, in the address *john.smith@mail.com*, "john.smith" is the prefix, and "mail.com" is the domain.

Note that, we say that a character is *alphanumeric* if it is a letter of the alphabet, 'A' to 'Z' or 'a' to 'z', or one of the arabic numerals, '0' to '9'. For instance, 'G' is an alphanumeric character while '&' is not.

Acceptable prefix formats. For a prefix to be acceptable it must adhere to the following constraints:

- It contains at least one character.
- It contains only alphanumeric characters, underscores ('_'), periods ('.'), and dashes ('-').
- An underscore, a period, or a dash must always be followed by one or more alphanumeric characters.
- The first character must be alphanumeric.

Examples of valid prefixes are: "abc-d", "abc.def", "abc", "abc_def".
Examples of invalid prefixes are: "abc-", "abc..d", ".abc", "abc#def".

Acceptable domain formats. For a domain to be acceptable it must adhere to the following constraints:

- It is made up of two portions separated by a period.
- The first portion contains at least one character.
- The second portion contains at least two characters.
- The first portion contains only alphanumeric characters, periods, and dashes. More over, a period or a dash must always be followed by one or more alphanumeric characters.
- The second portion contains only letters of the alphabet.

Examples of valid domains are: "mail.cc", "mail-archive.com", "mail.org", "mail.mcgill.ca" (here the first portion of the domain is "mail.mcgill", while "ca" is the second portion)
Examples of invalid domains are: "mail.c", "mail#archive.com", "mail", "mail..com", ".com", "mail.c9".

To complete your task, you need to implement all the methods listed below. All the code for this question must be placed in the file named `EmailValidation.java`. Note that you are free to write additional methods if they help the design or readability of your code.

2a) Method to check if a character is alphanumeric

Write a method `isAlphanumeric()` that takes as input a character. The method returns `true` if such character is a letter of the English alphabet (uppercase or lower case) or one of the arabic numerals. The method returns `false` otherwise. For example:

- `isAlphanumeric('g')` returns `true`
- `isAlphanumeric('B')` returns `true`
- `isAlphanumeric('3')` returns `true`
- `isAlphanumeric('-')` returns `false`

2b) Methods to check is a character is a valid prefix/domain character

Write the following two methods:

- A method `isValidPrefixChar()` that takes as input a character and returns `true` if the character can be used in the prefix of a valid email address, `false` otherwise. Note that a valid prefix can contain only alphanumeric characters, dashes, periods, or underscores. For example, `isValidPrefixChar('_')` returns `true`, while `isValidPrefixChar('&')` returns `false`.
- A method `isValidDomainChar()` that takes as input a character and returns `true` if the character can be used in the domain (first portion) of a valid email address, `false` otherwise. Note that a valid first portion of a domain can contain only alphanumeric characters, dashes, or periods. For example, `isValidDomainChar('-')` returns `true`, while `isValidDomainChar('_')` returns `false`.

To get full marks, your method must use the `isAlphanumeric()` method defined above.

2c) Method to check if a String contains exactly one '@'

Write a method `exactlyOneAt()` that takes as input a `String` representing a possible email address, and returns `true` if the string contains exactly one `'@'`, `false` otherwise.

For example:

- `exactlyOneAt("example@email.com")` returns `true`.
- `exactlyOneAt("b@n@n@s")` returns `false`
- `exactlyOneAt("@pple")` returns `true`

2d) Method to get the prefix of a possible email address

Write a method `getPrefix()` that takes as input a `String` representing a possible email address. The method returns a `String` containing the prefix of the possible email address. In this method, you can assume that the String received as input contains exactly one '@'.

For example:

- `getPrefix("example@email.com")` returns `"example"`.
- `getPrefix("cats @nd dogs")` returns `"cats "`.
- `getPrefix("@pple")` returns `""`.

2e) Method to get the domain of a possible email address

Write a method `getDomain()` that takes as input a `String` representing a possible email address. The method returns a `String` containing the domain of the possible email address. In this method, you can assume that the String received as input contains exactly one '@'.

For example:

- `getDomain("example@email.com")` returns `"email.com"`.
- `getDomain("cats @nd dogs")` returns `"nd dogs"`.
- `getDomain("@pple")` returns `"pple"`.

2f) Methods to check if the prefix and the domain are valid

Write the following two methods:

- `isValidPrefix()` takes a `String` as input representing the prefix of a possible email address. The method returns `true` if the input adhere to all the constraints listed in the above paragraph titled "Acceptable prefix formats", `false` otherwise.

- `isValidDomain()` takes a `String` as input representing the domain of a possible email address. The method returns `true` if the input adhere to all the constraints listed in the above paragraph titled "Acceptable domain formats", `false` otherwise.

Examples:

- `isValidPrefix("abc_def")` returns `true`.
- `isValidPrefix("mail.cc")` returns `true`.
- `isValidPrefix("abc..d")` returns `false`.
- `isValidPrefix("abc#d")` returns `false`.
- `isValidDomain("mail.cc")` returns `true`.
- `isValidDomain("abc-def.ghi")` returns `true`.
- `isValidDomain("abc..d")` returns `false`.
- `isValidDomain(".com")` returns `false`.

To get full marks, your method must use at least `isValidPrefixChar()` and `isValidDomainChar()`.

2g) Methods to check if a string is a valid email address

Write the method `isValidEmail()` which takes as input a `String` and returns `true` if the string is a valid email address, `false` otherwise. To get full marks, your method must use all the methods you have written up to now (either directly or indirectly).

For example:

- `isValidEmail("abc..def@mail.com")` returns `false`.
- `isValidEmail("abc#def@mail.com")` returns `false`.
- `isValidEmail("abc.def@mail")` returns `false`.
- `isValidEmail("abc.def@mail..com")` returns `false`.
- `isValidEmail("abc_d@mail.com")` returns `true`.
- `isValidEmail("abc.def@mail.com")` returns `true`.
- `isValidEmail("abc@mail.com")` returns `true`.
- `isValidEmail("abc.def@mail-archive.com")` returns `true`.

What To Submit

Please put all your files in a folder called *Assignment2*. Zip the folder (DO NOT RAR it) and submit it through MyCourses.

Inside your zipped folder, there must be only the following files. **Do not submit any other files, especially .class files.** Any deviation from these requirements may lead to lost marks.

`Bunco.java`
`EmailValidation.java`
`Confession.txt` (optional) In this file, you can tell the TA about any issues you ran into doing this assignment.

Note that the *Assignment2* folder must **not** contain any subfolder!