Programming Exercise 02

Authors: Akul, Andrew, Lucas, Cindy, Kevin, Saira, Vince

Problem Description

This assignment will test your basic knowledge of Conditionals, Strings, and Input/Output.

You will solve 2 problems:

1. Creating a calculator
2. Generating a password

Sample outputs for these two problems are available down below.

Solution Description

Creating a calculator

1. Create a class called `Calculator`
2. Add to this class a `main` method
3. In the `main` method do the following.
4. Create a variable called `operation` which will be the mathematical operation that will be passed in by the user as a sequence of characters
5. Create a variable called `firstNumber` which will always be a **whole number**
6. Create a variable called `secondNumber` which will always be a **whole number**
7. Create a variable called `result` which will always be a whole number
8. Declare a scanner called `takesInput` that will take in input from the user
9. Print to the console `"What operation would you like to perform (Addition, Multiplication, Division, or Subtraction)?"`
10. Read the value inputted by the user, convert the input to all uppercase and assign it to `operation.`
11. Print to the console "`Enter your first number:",` this should be printed below the previous value inputted by the user
    - Read the value inputted by the user and assign it to `firstNumber`
12. Print to the console `"Enter your second number:",` this should be printed below the previous value inputted by the user
    - Read the value inputted by the user and assign it to `secondNumber`
13. Use a **switch (required in order to receive credit) statement** to determine what the answer should be for the operation
    - If the `operation` is "`ADDITION`", you should add the two numbers, assign the answer to `result,` and print to the console.

      ```
      We added {firstNumber} and {secondNumber}. The answer is
      {result}.
      ```

    - If the `operation` is "`MULTIPLICATION`", you should multiply the two numbers, assign the answer to `result,` and print to the console.

```
Here we multiplied {firstNumber} and {secondNumber}. The
answer is {result}.
```

- If the `operation` is "`DIVISION`", you should do `firstNumber/secondNumber` , assign the answer to `result,` and print to the console.

```
This time we divided {firstNumber} and {secondNumber}. The
answer is {result}.
```

- If the `operation` is "`SUBTRACTION`", you should do `firstNumber - secondNumber`, assign the answer to `result,` and print to the console.

```
{firstNumber} and {secondNumber} were subtracted. The
answer is {result}.
```

- If the `operation` is not one of the above choices, you should print the following to the console.

```
That is not a valid operation!
```

NOTE: Prompts, inputs, and output should each be printed on separate lines

NOTE: Only whole numbers will be passed in by the user

NOTE: For the case of division, you do not have to deal with dividing by 0

Generating a password

1. Create a class called `PasswordGenerator`
2. Create a main method in `PasswordGenerator`
3. In the `main` method do the following:
4. Create a variable consisting of letters called `password`
5. Create a scanner called `myScanner` that will take in input from the user
6. Print to the console `"Enter a phrase that will be converted to a password:"`
7. Read the value inputted by the user and assign it to `password`.
8. Use **if/else if/ else conditionals to check for the length** and, you must **use String operations to replace** the letters as listed below. This is **required for full credit**.
   - The password should always be uppercase
   - Replace the following letters in the phrase with values

     "`A`" or "`a`" should be replaced with `"@"`

     `"E"` or `"e"` should be replaced with `"#"`

     `"I"` or "`i`" should be replaced with `"!"`

     "`O`" or "`o`" should be replaced with "`0`" (zero)

     "`U`" or "`u`" should be replaced with `"&"`

9. After making these substitutions get the length of the password
   - If the length of the string is between 2 and 5 (inclusive) append the characters from indices 0 to the end of the string to the back of the password (e.g. `xyz` should become `xyzxyz`). Print the following to the console-

```
Your password could be guessed so the length was adjusted!
Your new password is {password}.
```

- If the length of the string is greater than or equal to 6, then print to the console

```
Your password is the perfect length! Your password is
{password}.
```

- If the length of the string is less than or equal to 1, then print to the console

```
This password will not work. You should try again!
```

NOTE: Prompts, inputs, and output should each be printed on separate lines

NOTE: There are certain string methods that will make these operations much easier. Please refer to the API and course materials.

Testing

This is what a sample output would look like.

For Calculator.java: (items below in bold are inputs provided by the User)

```
What operation would you like to perform (Addition, Multiplication,
Division, or Subtraction)?

Addition

Enter your first number:

4

Enter your second number:

2

We added 4 and 2. The answer is 6.
```

For Password.java: (items below in bold are inputs provided by the User)

```
Enter a phrase that will be converted to a password:

abc

Your password could be guessed so the length was adjusted! Your new
password is @BC@BC.
```

Creating a calculator

Try changing the values passed into the user for `operation` and confirm the output is correct.

Generating a password

Try changing the phrases passed into the user for `password` and confirm the output is correct including the length and symbols in the password.

Import Restrictions

You may not import anything for this homework assignment **except for** `java.util.Scanner`

Feature Restrictions

There are a few features and methods in Java that overly simplify the concepts we are trying to teach or break our auto grader. For that reason, do not use any of the following in your final submission:

- `var` (the reserved keyword)
- `System.exit`

Collaboration

Collaboration Statement

To ensure that you acknowledge collaboration and give credit where credit is due, **we require that you place a collaboration statement as a comment at the top of at least one java file that you submit**. That collaboration statement should say either:

- *I worked on the homework assignment alone, using only course materials.*

or

- *In order to help learn course concepts, I worked on the homework with [give the names of the people you worked with], discussed homework topics and issues with [provide names of people], and/or consulted related material that can be found at [cite any other materials not provided as course materials for CS 1331 that assisted your learning].*

Recall that comments are special lines in Java that begin with `//`.

Turn-In Procedure

Submission

To submit, upload the files listed below to the corresponding assignment on Gradescope:

- `PasswordGenerator.java`
- `Calculator.java`

Make sure you see the message stating "PE02 submitted successfully". From this point, Gradescope will run a basic autograder on your submission as discussed in the next section.

You can submit as many times as you want before the deadline, so feel free to resubmit as you make substantial progress on the homework. We will only grade your last submission: be sure to **submit every file each time you resubmit**.

Gradescope Autograder

For each submission, you will be able to see the results of a few basic test cases on your code. If you fail a test, you can look at the output to determine what went wrong and resubmit once you have fixed the issue.

The Gradescope tests serve two main purposes:

1. Prevent upload mistakes (e.g. non-compiling code)
2. Provide basic formatting and usage validation

In other words, the test cases on Gradescope are by no means comprehensive. Be sure to thoroughly test your code by considering edge cases and writing your own test files. You also should avoid using Gradescope to compile, run, or checkstyle your code; you can do that locally on your machine.

Other portions of your assignment can also be graded by a TA once the submission deadline has passed, so the output on Gradescope may not necessarily reflect your grade for the assignment.

Important Notes (Don't Skip)

- Non-compiling files will receive a 0 for all associated rubric items
- Do not submit `.class` files.
- Test your code in addition to the basic checks on Gradescope
- Submit every file each time you resubmit
- Read the "Allowed Imports" and "Restricted Features" to avoid losing points
- Check on Piazza for all official clarifications