L6b

Re-submit Assignment

Due Oct 16, 2018 by 11:59pm **Points** 10 **Submitting** a file upload **File Types** zip
Available after Oct 1, 2018 at 8am

Problem #1

Write a program that takes a user ID as a String input from the keyboard and steps (using a for loop) through every character, counting how many characters in the user ID are digits (utilize `Character.isDigit` method). If there are exactly two digits, the program should output that the user ID is valid; otherwise that it is invalid.

You must use a **for** loop.

Problem#2

The last digit of a credit card number is the check digit, which protects against transcription errors such as an error in a single digit or switching two digits. The following algorithm is used to verify the actual credit card numbers but, for simplicity, we will describe it for numbers with 8 digits instead of 16:

- all digits at odd index should be added to form the sum.
  - For example, if the credit card number is **43589795**, then you form the sum **3 + 8 + 7 + 5 is 23**
- double each of the digits that are at even index and add all digits of the resulting numbers to the sum.
  - For example, with the number given above, doubling the digits yields **8 10 18 18**. Adding all digits in these values yields **8 + 1 + 0 + 1 + 8 + 1 + 8** is **27**, so the final sum total is **23 + 27** which is **50**
- if the last digit of the final sum is **0**, the credit card number is valid. In our case the last digit of **50** is **0**, so the number is valid.

Write a program that implements this algorithm. The user should supply an **8-digit number**, and the program should print the calculated sum and the message if the given card is valid or not. The program must allow user to check another card if the user wishes to do so.

Use the following flow of the program:

```
do
{
   do
   {
     prompt the user for credit card number
   }while (the entered card does not match the specified pattern)
```

```
  Using a for loop go over the input card number, one character at the time
    starting with the first character (see Lecture Notes Chapter 6 "Processing a String" slide)
  {
      - utilize Character.digit method to convert the char to and int
      - if the current character is at odd index add the int to sum total
      - if the current character is at even index double the int
           and then utilize division by 10 and modulus by 10 to get
           the first and last digit respectively, add the digits to the sum total
   }
   Check if the calculated sum ends with zero (again use % operator)
   Display the results
   Ask the user if (s)he wants to convert another card
}while (user wants to convert another card)
```

Sample Run:

```
Enter 8-digit credit card number
 43589799
The checksum is 54
The credit card number "43589799" is NOT valid

Would you like to check another credit card? (yes/no)
 yes
Enter 8-digit credit card number
 1234
Enter 8-digit credit card number
 43589795
The checksum is 50
The credit card number "43589795" is valid

Would you like to check another credit card? (yes/no)
 no
Bye!
```

Problem #3

Write a program that uses nested **for** loops to create the pattern of Xs and Os, in which on every line each letter is displayed one additional space to the right. Use a **toggle** variable (a boolean flag) to alternate between `X` and `O` to produce output as shown in the sample run below:

```
Enter the number of lines: 7


X
 O
  X
```

O

X

O

X