CS215: Introduction to Program Design, Abstraction and Problem Solving (Spring, 2025)

Lab Assignment 4

(20 points)

Today's Date: Monday, February 3

Demonstration Due Date: the end of Lab5 class

Submission Due Date: Friday, February 14

The purpose of this lab assignment is
- to continue practicing conditional statements
- to practice loop statements

Problem Statement

Write a program that performs *Credit Card Number Check*. Your program should repeatedly ask the user to input an 8-digit number to check if it is valid, until the user enters **-1** to quit. (Please note that for simplicity, now you can assume the user will always enter 8-digit numbers to check except for quitting the program with **-1**. In the coming week, we will learn how to handle the user input validation). Please note that this Lab assignment is from the Programming Project 4.22.21 of the ZyBook.
Your program will implement the following algorithm:
The last digit of a credit card number is the ***check digit***, which protects against transactions errors such as an error in a single digit or switching two digits. The following method is for numbers with 8 digits:

- Starting from the rightmost digit, form the sum of every other digit. For example, if the credit card number is **43589795**. Then you form the sum **5** + **7** + **8** + **3** = 23
- Double each of the digits, which were not included in the preceding step. Add all digits of the resulting numbers. For example, with the number given above, doubling the digits, starting with the next-to-last one, yields **18 18 10 8**. Adding all digits in these values yields 1 + 8 + 1 + 8 + 1 + 0 + 8 = 27.
- Add the sums of the two preceding steps. If the last digit of the result is 0, the number is valid. In our case, 23 + 27 = 50, so the number is valid.

After the user supplies an 8-digit number, your program should implement the above algorithm, then print out whether the number is a valid credit card number or not. If it is not valid, you should print out the value of the check digit that would make the number valid.

(Please note that for this Lab assignment, you are not allowed to store a credit card number as a string, you must use integer variable to store it!)

The following show some examples of running your program:

Sample output 1:

```
Please enter the 8-digit credit card number (enter -1 to
quit): -1↵
Thank you for using "Credit Card Number Validation"!
```

Sample output 2:

```
Please enter the 8-digit credit card number (enter -1 to
quit): 43589795↵
Number is valid.

Please enter the 8-digit credit card number (enter -1 to
quit): 43589794↵
Number is invalid.
Check digit should have been 5

Please enter the 8-digit credit card number (enter -1 to
quit): 43589793↵
Number is invalid.
Check digit should have been 5

Please enter the 8-digit credit card number (enter -1 to
quit): 43589791↵
Number is invalid.
Check digit should have been 5

Please enter the 8-digit credit card number (enter -1 to
quit): 43589790↵
Number is invalid.
Check digit should have been 5

Please enter the 8-digit credit card number (enter -1 to
quit): 43889795↵
Number is invalid.
Check digit should have been 9

Please enter the 8-digit credit card number (enter -1 to
quit): 43889799↵
Number is valid.

Please enter the 8-digit credit card number (enter -1 to
quit): 43889794↵
Number is invalid.
Check digit should have been 9

Please enter the 8-digit credit card number (enter -1 to
quit): 33334444↵
Number is invalid.
Check digit should have been 2
```

```
Please enter the 8-digit credit card number (enter -1 to
quit): 33334442↵
Number is valid.

Please enter the 8-digit credit card number (enter -1 to
quit): 11111111↵
Number is invalid.
Check digit should have been 9

Please enter the 8-digit credit card number (enter -1 to
quit): 11111119↵
Number is valid.

Please enter the 8-digit credit card number (enter -1 to
quit): 12345678↵
Number is invalid.
Check digit should have been 4

Please enter the 8-digit credit card number (enter -1 to
quit): 12345674↵
Number is valid.

Please enter the 8-digit credit card number (enter -1 to
quit): -1↵
Thank you for using "Credit Card Number Validation"!
```

(Note that the blue part represents the user input, and "↵" represents the return key from the user input.)

Extra (No Credit)

Think about how you can make your program check for the real credit card number (16-digit number instead of 8-digit number as in this lab assignment). Please note that you need to declare a variable as the data type of "long long integer (`long long`)" to store 16-digit number, for example:

```
    long long cardNumber = 1234567891234567;
```

The following shows one sample output of running your program to check each 16-digit credit card number:

```
Please enter the 16-digit credit card number (enter -1 to
quit): 1234567891234567↵
Number is invalid.
Check digit should have been 3

Please enter the 16-digit credit card number (enter -1 to
quit): 1234567891234563↵
Number is valid.

Please enter the 16-digit credit card number (enter -1 to
quit): 9876543219876543↵
Number is invalid.
Check digit should have been 8

Please enter the 16-digit credit card number (enter -1 to
quit): 9876543219876548↵
Number is valid.

Please enter the 16-digit credit card number (enter -1 to
quit): 9876543213654321↵
Number is valid.

Please enter the 16-digit credit card number (enter -1 to
quit): 4358979543589795↵
Number is valid.

Please enter the 16-digit credit card number (enter -1 to
quit): 4358979543589790↵
Number is invalid.
Check digit should have been 5

Please enter the 16-digit credit card number (enter -1 to
quit): 4358979543589791↵
Number is invalid.
Check digit should have been 5

Please enter the 16-digit credit card number (enter -1 to
quit): -1↵
Thank you for using "Credit Card Number Validation"!
```

Demonstration and Submission

1. Each Lab assignment needs to demonstrate to your TA to be graded. You can demonstrate Lab4 during Lab4 class (with possible bonus 3 points) or no later than the end of Lab5 class (this is the **demonstration deadline** for Lab4).
If you finish Lab4 assignment during Lab4 class, you may demonstrate your program to your TA and answer your TA's questions, you can get up to 3 extra points for this lab assignment. (Note you can also demonstrate your program to your TA during Lab5 class. However, any demonstration later than the end of the Lab4 class cannot get bonus 3 points.)
If you need extra time, you can continue working on Lab4 assignment after the Lab class, and try to finish it before the next Lab class. Then demonstrate your Lab4 during Lab5 class.

If you do not demonstrate your code, even if you submit it in Canvas, you will receive a grade of 0!! *The TA may ask you to make some corrections. If so, make the corrections and demonstrate again…repeat until you have 100%!*

2. After the successful demonstration, submit the code on Canvas. Open the link to Course Canvas page (https://www.uky.edu/canvas) and log in to your account using your LinkBlue ID and password. Please submit your **source code in a .cpp file** through link "**`Lab 4`**".
Even if you successfully demonstrated it to the TA, if you do not submit it on Canvas by **the submission deadline,** ***you will receive a grade of 0!***

Grading (20 points + Bonus 3 points)

1. Attend the lab session or have a documented excused absence. (5 points)
2. Demonstrate your program to your TA and submit it in Canvas. (15 points)
    - Include comments as specified in the lecture notes. (2 points)
    - Repeatedly ask the user to input an 8-digit number until the user enters -1 to quit your program correctly. (4 points)
    - Generate the correct answer from checking if the number is valid. (5 points)
    - Generate the correct check digit if the number is not valid. (4 points)

Demonstrate your program to your TA and answer TA's questions during Lab class when the same Lab assignment is given. (Bonus 3 points)