COSC 24L3: Lab Assignment #4
Computer Science Department @ Dallas Baptist University
Fall 2023

Credit Card Validator

Credit card numbers follow the following patters. Take 5387821309382461 as an example

1. A credit card number must have between 13 and 16 digits. It must start with:
    - 4 for Visa cards
    - 5 for Master cards
    - 37 for American Express cards
    - 6 for Discover cards

2. Double the digit in the even places from right to left. If doubling of a digit results in a two-digit number, add up the two digits to get a single-digit number.
    - 6 * 2 = 12 (1 + 2 = 3)
    - 2 * 2 = 4
    - 3 * 2 = 6
    - 0 * 2 = 0
    - 1 * 2 = 2
    - 8 * 2 = 16 (1 + 6 = 7)
    - 8 * 2 = 16 (1 + 6 = 7)
    - 5 * 2 = 10 (1 + 0 = 1)

3. Now add all single-digit numbers from Step 1.
    - 3 + 4 + 6 + 0 + 2 + 7 + 7 + 1 = 30

4. Add all digits in the odd places from right to left in the card number.
    - 1 + 4 + 8 + 9 + 3 + 2 + 7 + 3 = 37

5. Sum the results from Step 2 and Step 3.
    - 30 + 37 = 67

6. If the result from Step 5 is divisible by 10, the card number is valid; otherwise, it is invalid. For example, the number 5387821309382461 is invalid, but the number 5387821309362060 is valid.

Write a program that asks the user to enter a credit card number as a long integer and checks if it is a valid credit card number.
Define the following static methods and use these methods for the program:

```
boolean isValid(long number)
int sumDoubleDigitEvenPlaces(long number)
int sumDigitsOddPlaces(long number)
int numberOfDigits(long number)
int startingWith(long number)
```

Here are sample runs of the program:

Sample 1:

```
Enter a credit card number as a long integer: 5387821309382461
5387821309382461 is invalid
```

Sample 2:

```
Enter a credit card number as a long integer: 5387821309362060
5387821309362060 is valid
```