|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **LEARNING PROFILE FOR ASSIGNMENT#\_2\_\_\_\_\_ AND QUESTION#\_3\_\_\_\_\_\_\_** | | | | | |
| *Name* | *:* | *Steven Morrissey* | *Due Date* | *:* |  |
| *Student ID* | *:* | *3300222* | *Submission Date* | *:* |  |

**1. Problem Statement:**

**Credit card numbers follow certain patterns. 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, and 6 for Discover cards. In 1954, Hans Luhn of IBM proposed the following algorithm for validating credit card numbers: a. Double every second digit from right to left (e.g., if number is 3 => 3 \* 2 => 6) and add them together. b. If this doubling results in a two-digit number, then add the two digits to get a single-digit number (e.g., if number is 5 => 5 \* 2 => 10 => 1+0 => 1). So, for the credit card number 4388576018402626, doubling all second digits from the right results in (2 \* 2 = 4) + (2 \* 2 = 4) + (4 \* 2 = 8) + (1 \* 2 = 2) + (6 \* 2 = 12 = 1 + 2 = 3) + (5 \* 2 = 10 = 1 + 0 = 1) + (8 \* 2 = 16 = 1 + 6 = 7) + (4 \* 2 = 8). This totals to 4 + 4 + 8 + 2 + 3 + 1 + 7 + 8 = 37. Add all digits in the odd places from right to left. The leftmost digit of the credit card number is at index 0; 6 + 6 + 0 + 8 + 0 + 7 + 8 + 3 = 38. Add results from steps (a) and (b) and see if divisible by 10. If it is, then the card number is valid; otherwise invalid. 37 + 38 = 75 is not divisible by 10, so it is an invalid credit card number. Implement Luhn’s algorithm in a program to determine whether a given credit card number is valid or not. You must test if the number of digits in the input is in the valid range (13 to 16), run Luhn’s algorithm to test its validity, and if it is valid, print the name of the company that offers that credit card number.**

**2. Description of the Code:**

Implements luhn algorithm by following the api, and the exercise requirements.

**3. Errors and Warnings:**

Table 1: List of Errors and Warnings Encountered in the Program

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Errors / Warnings** | **Details** | **How I solved them** |
| 1 | NumberFormatException | Trying to parse a number from the cc String gave an exception | Added try/catch, which in turn gave me an opportunity to add error handling. I then added it to the other methods that used parsing too |
| 2 |  |  |  |
| 3 |  |  |  |
| 4 |  |  |  |
| 5 |  |  |  |

**4. Sample Input and Output:**

CreditCard cc1 = new CreditCard("4916986239473270"); - Valid

CreditCard cc2 = new CreditCard("5527938827663505"); - Valid

CreditCard cc3 = new CreditCard("6011800466522395"); - Valid

CreditCard cc4 = new CreditCard("370890549413843"); - Valid

CreditCard cc5 = new CreditCard("414141414141414"); - Invalid, doesn’t pass the luhn algorithm check

CreditCard cc6 = new CreditCard("4141414"); - Invalid, doesn’t pass the length verification

CreditCard cc7 = new CreditCard("49169862394732700000000"); - Invalid, doesn’t pass the length verification

CreditCard cc8 = new CreditCard("1916986239473270"); - Invalid, doesn’t pass the luhn algorithm, the card provider is wrong

CreditCard cc9 = new CreditCard("5527ABCDEF663505"); - Invalid, alphanumeric string not accepted

**5. Discussion:**