Credit Card Validation

Objective

Write a program that validates credit card numbers using the Luhn algorithm. The program should take a credit card number as input and determine whether it is valid or not.

Requirements

1. **Input**: The program should accept a credit card number as a string of digits.
2. **Output**: The program should output whether the credit card number is valid or invalid.

Constraints

1. The input credit card number will be a string consisting of only digits (0-9).
2. The input string length will be between 13 and 19 characters inclusive.
3. The program should handle invalid inputs gracefully by checking the length and character set of the input.

Luhn Algorithm

The Luhn algorithm is used to validate a variety of identification numbers, primarily credit card numbers. Here are the steps to validate a credit card number using the Luhn algorithm:

1. Starting from the rightmost digit, double the value of every second digit. If the result of this doubling is greater than 9, then add the digits of the product (or subtract 9 from the product).
2. Sum all the digits of the credit card number (both modified and unmodified digits).
3. If the total modulo 10 is equal to 0, then the number is valid; otherwise, it is invalid.

You can research more about the Luhn Algorithm here

Example

- Input: "4532015112830366"
    - Step 1: Double every second digit from right to left (excluding the rightmost digit).
        - Original: 4532015112830366
        - Double: 8 5 6 2 0 2 2 5 1 1 4 8 6 3 6 6
    - Step 2: Sum all digits.
        - Sum: 8 + 5 + 6 + 2 + 0 + 2 + 2 + 5 + 1 + 1 + 4 + 8 + 6 + 3 + 6 + 6 = 65
    - Step 3: Check if the sum modulo 10 is 0.
        - 65 % 10 = 5, which is not 0, so the number is invalid.
- Output: "Invalid"

Task

1. Implement the validation.
2. Ensure that it correctly validates the credit card number according to the Luhn algorithm.
3. Include appropriate error handling for invalid input lengths and non-digit characters.