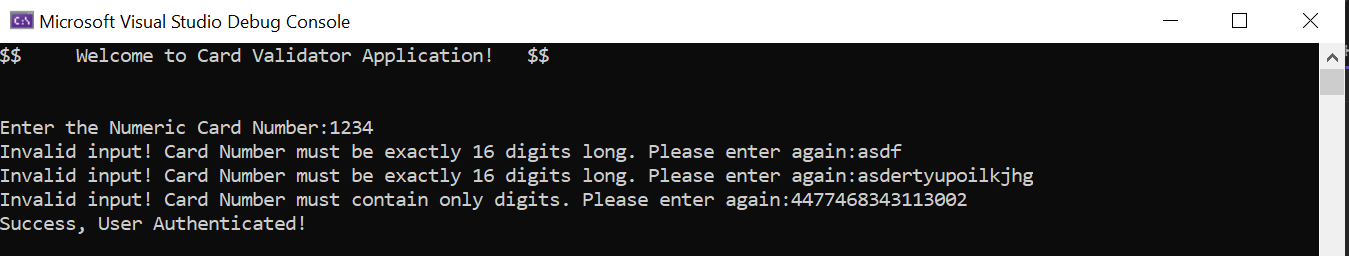
**Assignment 1: Access Card Validator Application**

using System;

using System.Linq;

namespace AccessCardValidator

{

internal class Program

{

/// <summary>

/// Takes a string input from the user and validates it.

/// </summary>

/// <param name="PrintValue">The prompt message to be displayed to the user.</param>

/// <returns>The valid 16-digit numeric string entered by the user.</returns>

static string TakingStringInput(string PrintValue)

{

string? str;

do

{

str = Console.ReadLine();

if (string.IsNullOrWhiteSpace(str) || str.Length != 16)

{

Console.Write($"Invalid input! {PrintValue} must be exactly 16 digits long. Please enter again:");

continue;

}

if (!str.All(char.IsDigit))

{

Console.Write($"Invalid input! {PrintValue} must contain only digits. Please enter again:");

continue;

}

} while (string.IsNullOrWhiteSpace(str) || str.Length != 16 || !str.All(char.IsDigit));

return str;

}

/// <summary>

/// Reverses the order of the digits in the card number.

/// </summary>

/// <param name="CardNumber">The 16-digit numeric card number.</param>

/// <returns>The reversed card number.</returns>

static string ReverseCardNumber(string CardNumber)

{

return new string(CardNumber.Reverse().ToArray());

}

/// <summary>

/// Checks the validity of the card number using the Luhn algorithm.

/// </summary>

/// <param name="CardNumber">The 16-digit numeric card number.</param>

/// <returns>True if the card number is valid, false otherwise.</returns>

static bool CheckValidation(string CardNumber)

{

// Ensuring the card number is not all zeros, all zeros will fail this algorithm

if (CardNumber.All(c => c == '0'))

{

return false;

}

string NewCardNumber = ReverseCardNumber(CardNumber);

int result = CalculateChecksum(NewCardNumber);

return result % 10 == 0;

}

/// <summary>

/// Calculates the checksum of the reversed card number using the Luhn algorithm.

/// </summary>

/// <param name="input">The reversed card number.</param>

/// <returns>The checksum calculated.</returns>

static int CalculateChecksum(string input)

{

int sum = 0;

for (int i = 0; i < input.Length; i++)

{

int digit = (int)Char.GetNumericValue(input[i]);

if (i % 2 != 0) // Even position

{

digit \*= 2;

if (digit > 9) // Check if the result is two digits

{

sum += SumOfDigits(digit.ToString());

}

else

{

sum += digit;

}

}

else // Odd position

{

sum += digit;

}

}

return sum;

}

/// <summary>

/// To find Sum of digits of given string input

/// </summary>

/// <param name="input"> CardNumber in string</param>

/// <returns></returns>

static int SumOfDigits(string input)

{

int sum = 0;

foreach (char c in input)

{

if (char.IsDigit(c))

{

sum += (int)Char.GetNumericValue(c);

}

}

return sum;

}

static void Main(string[] args)

{

Console.WriteLine("$$ Welcome to Card Validator Application! $$\n\n");

Console.Write("Enter the Numeric Card Number:");

string CardNumber = TakingStringInput("Card Number");

bool IsValid = CheckValidation(CardNumber);

if (IsValid)

{

Console.WriteLine("Success, User Authenticated!");

}

else

{

Console.WriteLine("User Authentication Failed, Invalid User!");

}

}

}

}

**Assignment 2: Create Doctors, Print them, and find by speciality Application.**