Name: Naga Venkatesh Gavini

NEU ID: 002705198

CSYE 6200 Assignment-1

1. Credit Card Number Validation:

Problem Description:

In this problem we are going to validate the credit card number from the users input and display whether it is valid card or not.

Analysis:

- **Algorithm**:
  - To solve this problem we are taking a reference from the algorithm suggested by the researcher Hans Luhn of IBM. The name of the validity check proposed by Hans was *Luhn Check or the Mod 10 check.*
  - Along with the Hans method, we are also validating whether the credit card starts with certain number or not.
    - 4 for Visa Cards
    - 5 for Master Cards
    - 37 for American Express Cards
    - 6 for Discover Cards

- **Difficulties Faced:**
  - At first, I tried to use all the function signatures that were already given in the program file, but I thought that it would be easier to come up with my own method signatures.
  - At one point, the program is not working as expected, so I have learned how to debug a java program in eclipse IDE. Then it becomes easy to understand the program variables state.

Source Code for 1st problem:

Code Link: Github Gist: Program 1 Source Code Web URL

```
package edu.northeastern.csye6200;

import java.util.Scanner;

public class LAB3_P1 {
	private static final int CARD_MIN_CHAR=13;
	private static final int CARD_MAX_CHAR=16;

	private static final String VISA_START_CHAR="4";
	private static final String MASTER_START_CHAR="5";
	private static final String AM_EXPRESS_START_CHAR="37";
	private static final String DISCOVER_START_CHAR="6";


	public static void main(String[] args) {
		long cardNumber = 0;
		try (Scanner scanner = new Scanner(System.in)) {
			System.out.print("Enter a credit card number as a long
integer:");

			cardNumber = scanner.nextLong();
		}
		catch (Exception e) {
			System.out.println("Failed to read input from user with error:
"+ e);
		}


		boolean isvalid = isValid(cardNumber);

		if(isvalid==true) {
			System.out.println(cardNumber + " is valid");
		}
		else {
			System.out.println(cardNumber + " is invalid");
```

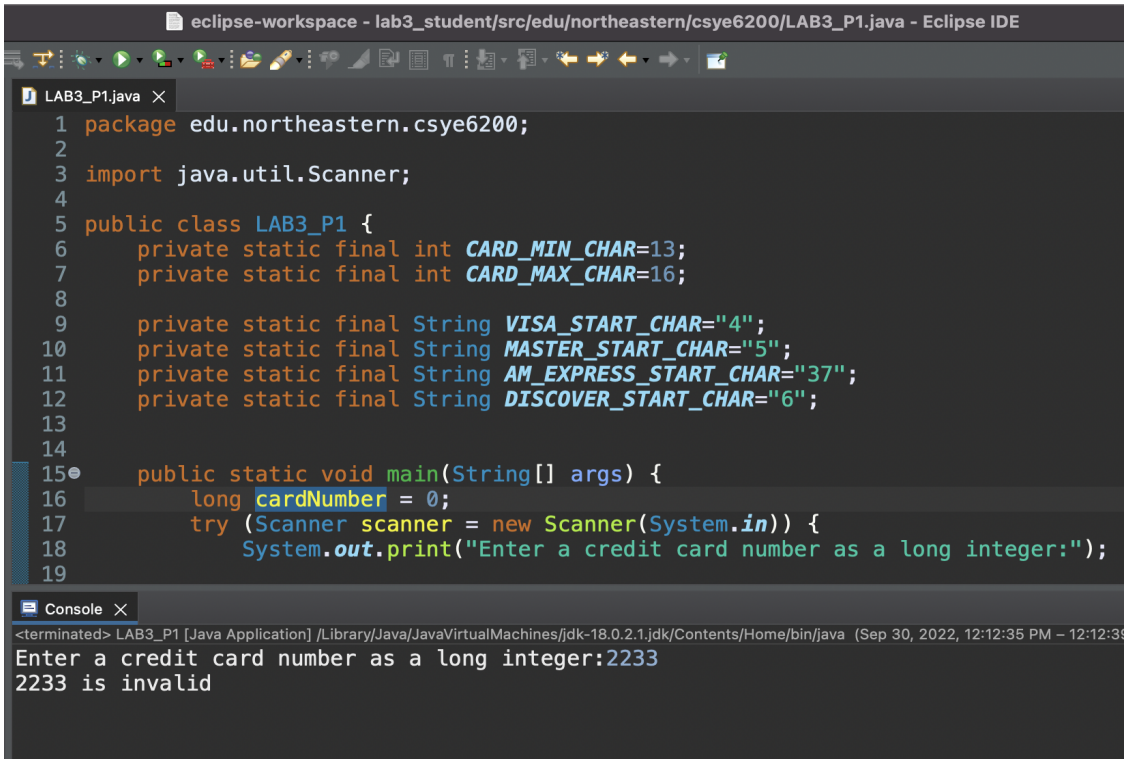
```
            }
      }

      /** Return true if the card number is valid */
      public static boolean isValid(long number) {
            int size = getSize(number);
            if(size > CARD_MAX_CHAR && size <CARD_MIN_CHAR) {
                  return false;
            }

            // Check whether the card starts with 4, 5, 37, 6
            String cardNumberInString = String.valueOf(number);
            if((cardNumberInString.startsWith(VISA_START_CHAR)
                        || cardNumberInString.startsWith(MASTER_START_CHAR)
                        || cardNumberInString.startsWith(AM_EXPRESS_START_CHAR)
                        || cardNumberInString.startsWith(DISCOVER_START_CHAR))
!= true) {
                  return false;
            }

            // Check if the sum divisible by 10 or not
            int evenSum = 0;
            int oddSum = 0;
            String cardNumberInStringReversedString =
getStringReversal(cardNumberInString);
            for(int i=0; i < cardNumberInStringReversedString.length(); i++) {
                  if((i+1) % 2 == 0) {
                        int numChar =
Character.getNumericValue(cardNumberInStringReversedString.charAt(i));
                        int evenNumDigit = sumOfDoubleEvenPlace(numChar);
                        evenSum += evenNumDigit;
                  }
                  else {
                        int oddNumDigit =
Character.getNumericValue(cardNumberInStringReversedString.charAt(i));
                        oddSum += oddNumDigit;
                  }
            }
```

```
        int entireSum = evenSum + oddSum;
        if(entireSum % 10 != 0) {
              return false;
        }

        return true;
  }


  public static String getStringReversal(String cardNumberInString) {
        String finalStr="";
        char ch;
        for (int i=0; i<cardNumberInString.length(); i++) {
              ch = cardNumberInString.charAt(i);
              finalStr = ch+finalStr;
              }
        return finalStr;
  }

  /** Get the result from Step 2 */
  public static int sumOfDoubleEvenPlace(int number) {
        int finalNumberVal = 0;
        int numberModified = number * 2;
        String numberModifiedInString = String.valueOf(numberModified);
        if(numberModifiedInString.length() > 1) {
              int sum = 0;
              char[] numbersArray = numberModifiedInString.toCharArray();
              for(char charNum : numbersArray) {
                    sum += Character.getNumericValue(charNum);
              }
              finalNumberVal = sum;
        }
        else {
              finalNumberVal = numberModified;
        }

        return finalNumberVal;
  }
```

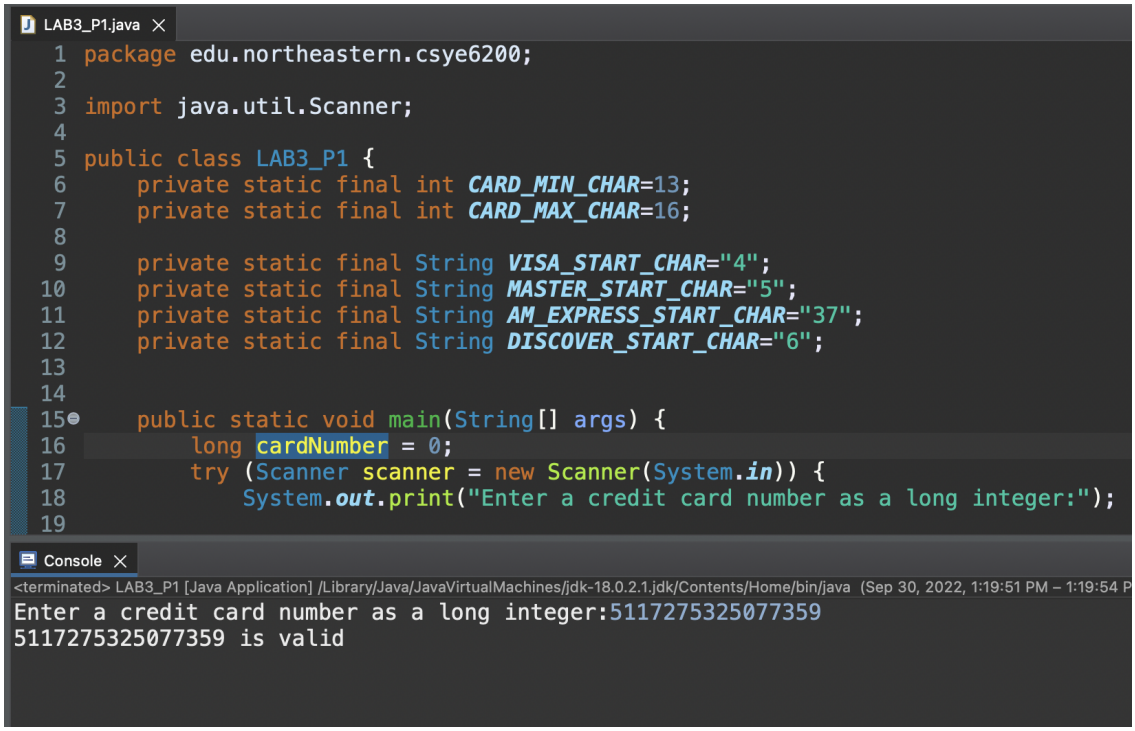
```
	/** Return the number of digits in d */
	public static int getSize(long d) {
		int length = String.valueOf(d).length();
		return length;
	}
}
```

Screenshots for 1st problem:

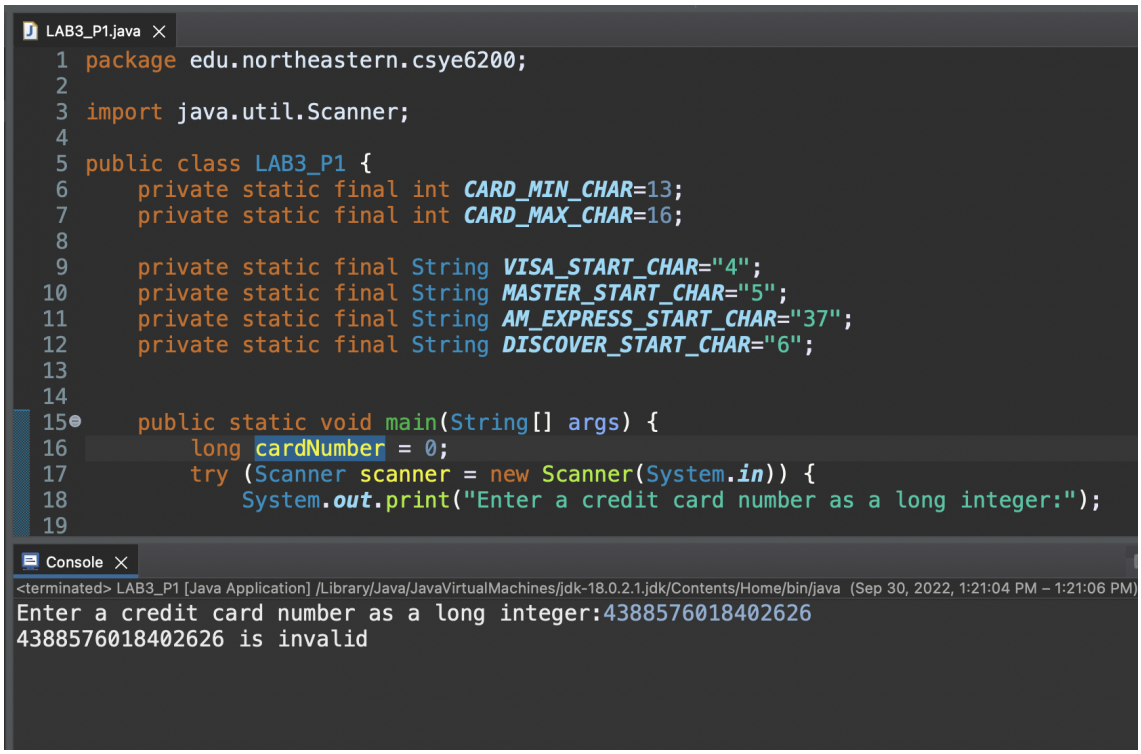
1. Input is 2233, which is not a valid card number and the program resulting the same output as shown in the screenshot.



2. Input is 5117275325077359, and it is valid card number. Program output is valid.



3. Input is 4388576018402626, and it is invalid card number as shown below.



2. Four Consecutives:

Problem Description:

In this problem, we have to check if an array of numbers has four consecutive numbers with a same value.

Analysis:

- **Algorithm:**
  - To solve this problem, I am using two variables, *counter* and *previous.*
  - The main logic starts with the initialization of *counter* and *previous* with 0 and first element of array respectively.
  - Next, start iterating from the 2nd element of array and compare it with the previous value, if previous and current are same, then the counter will be increased and this process will be repeated for the sub sequent elements in the array.
  - If the counter value reaches to 4, program declare that the array has the consecutive four.

- **Difficulties Faced:**
  - After writing the code with the logic, it worked fine for the sample inputs that were given in the assignment pdf, but it failed for the my own input which has consecutives till the end of the array.
  - Then, I have added a condition to handle such cases and it started working fine.
  - The debugging concepts that I learned during my first program has helped here.

Source Code for 2nd problem:

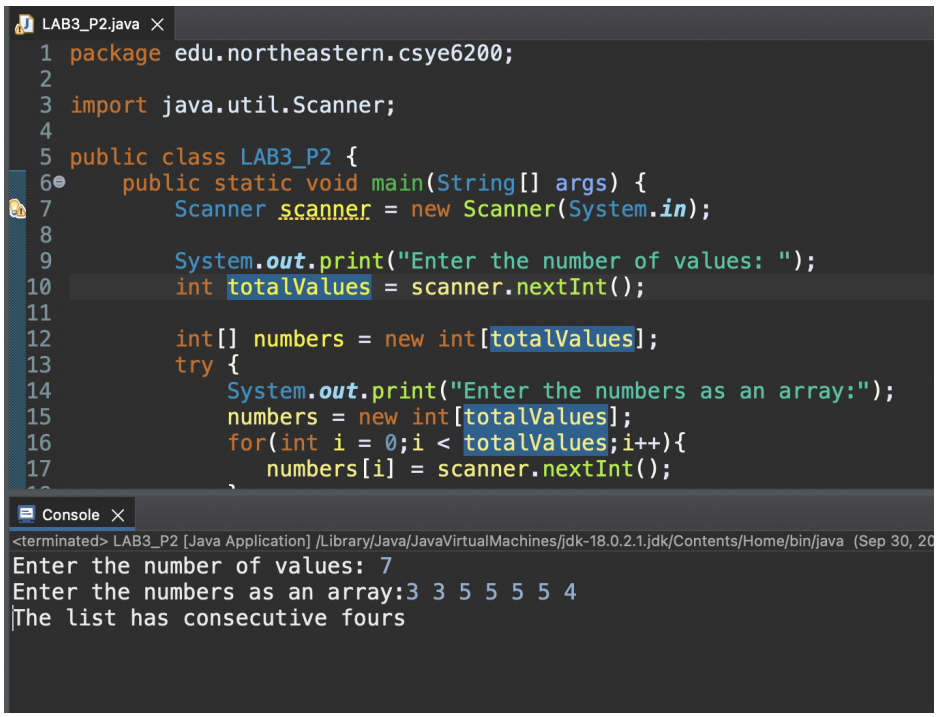
Code Link: Github Gist: Program 2 Source Code Web URL

```
package edu.northeastern.csye6200;

import java.util.Scanner;

public class LAB3_P2 {
	public static void main(String[] args) {
		Scanner scanner = new Scanner(System.in);

		System.out.print("Enter the number of values: ");
		int totalValues = scanner.nextInt();

		int[] numbers = new int[totalValues];
		try {
			System.out.print("Enter the numbers as an array:");

			numbers = new int[totalValues];
			for(int i = 0;i < totalValues;i++){
			   numbers[i] = scanner.nextInt();
			}
		} catch (NumberFormatException e) {
			e.printStackTrace();
		}

		boolean isConsecutiveFourExists =
isConsecutiveFour(numbers);
		if(isConsecutiveFourExists) {
			System.out.println("The list has consecutive
fours");
		}else {
			System.out.println("The list has no consecutive
fours");
		}
	}

	public static boolean isConsecutiveFour(int[] values) {
```

```
            int counter = 0;
            int previous = values[0];
            for(int i=1; i < values.length; i++) {
                  if(values[i] == previous) {
                        counter++;
                  }
                  previous = values[i];

                  if(i==(values.length-1)) {
                        if(values[i]==previous) {
                              counter++;
                        }
                  }

                  if(counter==4) {
                        return true;
                  }
            }
            return false;
      }

}
```

Screenshots:

1. Input array is 3 3 5 5 5 5 4, the program out is correct as it has four consecutive 5's.



2. Input array is 3 4 5 5 6 5 5 4 5, and program said it doesn't have the consecetives which is correct.