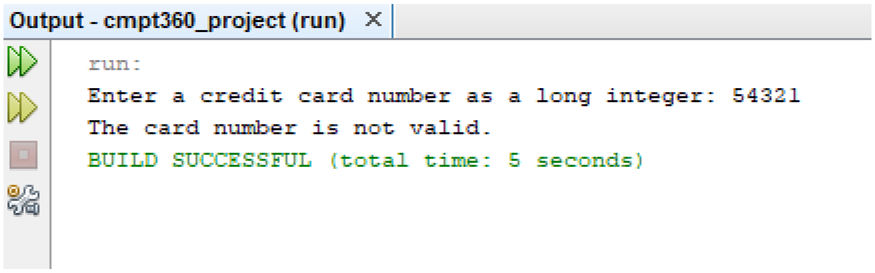
Giorgi Samushia

Ashly Vargas

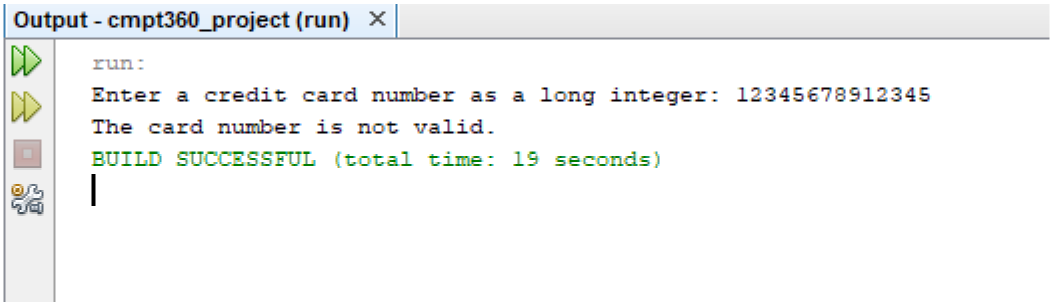
```
import java.util.Scanner;
public class cardNumber {
  public static void main(String[] args) {
    //Scanner object
    Scanner input = new Scanner(System.in);
    long cardNumber;
    int numOfDigits;
    int sum;
    System.out.print("Enter a credit card number as a long integer: ");
    cardNumber = input.nextLong();
    //Checking if the number is between 13 and 16 digits
    if(getSize(cardNumber) < 13 || getSize(cardNumber) > 16) {
      System.out.println("The card number is not valid.");
      return;
    }
    //Checking if the prefix is valid
    if(!prefixMatched(cardNumber, 4)
     || !prefixMatched(cardNumber, 5)
     || !prefixMatched(cardNumber, 37)
     || !prefixMatched(cardNumber, 6)) {
      System.out.println("The card number is not valid.");
      return;
    }
    numOfDigits = getSize(cardNumber);
    System.out.println("The number of digits is " + numOfDigits);
  sum = sumOfDoubleEvenPlace(cardNumber) + sumOfOddPlace(cardNumber);
  System.out.println("Sum from Step 4 is " + sum);
  if(isValid(cardNumber)) {
    System.out.println(cardNumber + " is valid");
  }
  else
    System.out.println(cardNumber + " is invalid");

}
//Checking the validity
public static boolean isValid(long number){
  return (sumOfDoubleEvenPlace(number) + sumOfOddPlace(number)) % 10 == 0;
}
//Going from right to left and summing up the numbers at even indexes
public static int sumOfDoubleEvenPlace(long number) {
  int evenSum = 0;
  String strnum = number + "";
  for(int i = getSize(number) - 2; i >= 0; i -= 2){
    evenSum += getDigit(Integer.parseInt(strnum.charAt(i) + "") * 2);
  }
  return evenSum;
}
//Returning either the number or the sum of its digits.
public static int getDigit(int number) {
  if(number >= 10){
    return (number % 10) + (number / 10);
  }
  else
    return number;
```

```
}
//Summing up odd places from right to left
public static int sumOfOddPlace(long number) {
   int oddSum = 0;
   String strnum = number + "";
   for(int i = getSize(number) - 1; i >= 0; i -= 2){
      oddSum += Integer.parseInt(strnum.charAt(i) + "");
   }
   return oddSum;
}
//Returning true if the prefix of the number is valid
public static boolean prefixMatched(long number, int d) {
   while(number >= 100){
      number /= 10;
   }
   if(number == 37){
      return true;
   }
   else {
      number /= 10;
   }
   switch((int)number){
      case 4:
      case 5:
      case 6:
            return true;
      default: return false;
   }
}
```

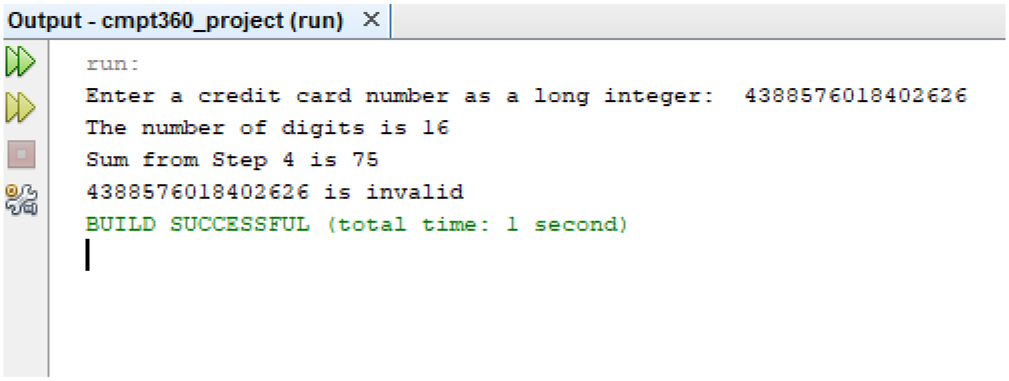
```
    //Returning the number of digits in d
    public static int getSize(long d) {
        String str = d + "";
        return str.length();
    }
}
```



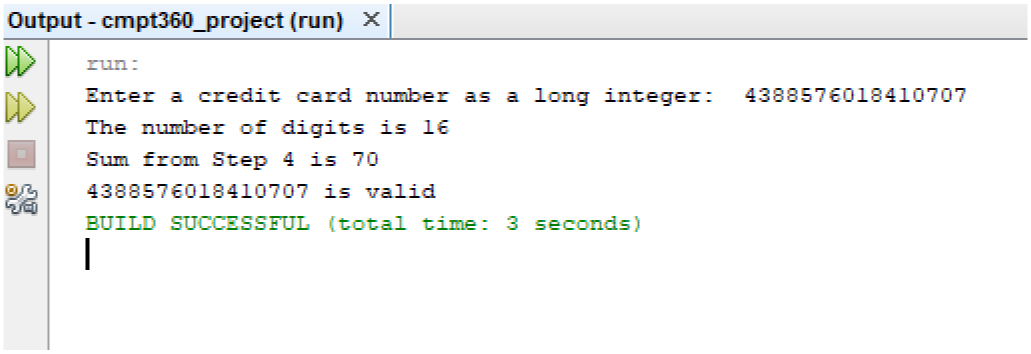
```
run:
Enter a credit card number as a long integer: 54321
The card number is not valid.
BUILD SUCCESSFUL (total time: 5 seconds)
```



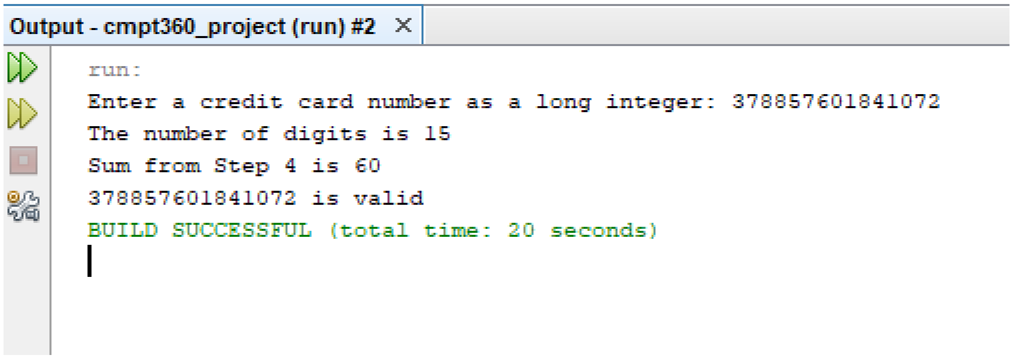
```
run:
Enter a credit card number as a long integer: 12345678912345
The card number is not valid.
BUILD SUCCESSFUL (total time: 19 seconds)
```



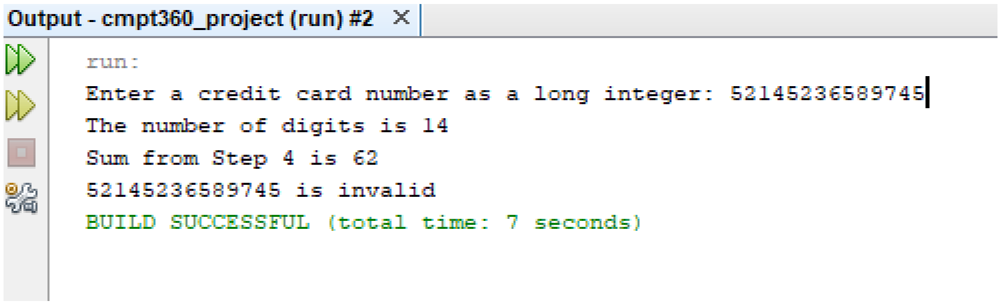
```
run:
Enter a credit card number as a long integer:  4388576018402626
The number of digits is 16
Sum from Step 4 is 75
4388576018402626 is invalid
BUILD SUCCESSFUL (total time: 1 second)
```



```
run:
Enter a credit card number as a long integer:  4388576018410707
The number of digits is 16
Sum from Step 4 is 70
4388576018410707 is valid
BUILD SUCCESSFUL (total time: 3 seconds)
```



```
run:
Enter a credit card number as a long integer: 378857601841072
The number of digits is 15
Sum from Step 4 is 60
378857601841072 is valid
BUILD SUCCESSFUL (total time: 20 seconds)
```



```
run:
Enter a credit card number as a long integer: 52145236589745
The number of digits is 14
Sum from Step 4 is 62
52145236589745 is invalid
BUILD SUCCESSFUL (total time: 7 seconds)
```