← Back to study plan

Work on project. Stage 2/4: Luhn algorithm

Project: Simple Banking System

Hard | 19 minutes

365 users solved this problem. Latest completion was 1 minute ago.

Description

5 / 5 Prerequisites

- Package (Stage 2)
- Constructor (Stage 2)
- Instance methods (Stage 2)
- Access modifiers (Stage 2)
- Getters and setters (Stage 2)

In this stage, we will find out what the purpose of the checksum is and what the Luhn algorithm is used for.

The main purpose of the check digit is to verify that the card number is valid. Say you're buying something online, and you type in your credit card number incorrectly by accidentally swapping two digits, which is one of the most common errors. When the website looks at the number you've entered and applies the Luhn algorithm to the first 15 digits, the result won't match the 16th digit on the number you entered. The computer knows the number is invalid, and it knows the number will be rejected if it tries to submit the purchase for approval, so you're asked to re-enter the number. Another purpose of the check digit is to catch clumsy attempts to create fake credit card numbers. Those who are familiar with the Luhn algorithm, however, could get past this particular security measure.

Luhn Algorithm in action

Luhn algorithm is used to validate a credit card number or other identifying numbers, such as Social Security. Luhn algorithm, also called Luhn formula or modulus 10, checks the sum of the digits in the card number and checks whether the sum matches the expected result or if there is an error in the number sequence. After working through the algorithm, if the total modulus 10 equals zero, then the number is valid according to the Luhn method.

While the algorithm can be used to verify other identification numbers, it is usually associated with credit card verification. The algorithm works for all major credit cards.

Here is how it works for a credit card with the number 4000008449433403:

Step	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	Total
Original number:	4	0	0	0	0	0	8	4	4	9	4	3	3	4	0	3	
Drop the last digit:	4	0	0	0	0	0	8	4	4	9	4	3	3	4	0		
Multiply odd digits by 2:	8	0	0	0	0	0	16	4	8	9	8	3	6	4	0		
Subtract 9 to numbers over 9:	8	0	0	0	0	0	7	4	8	9	8	3	6	4	0		
Add all numbers:	8	0	0	0	0	0	7	4	8	9	8	3	6	4	0	3	60

If the received number is divisible by 10 with the remainder equal to zero, then this number is valid; otherwise, the card number is not valid. When registering in your banking system, you should generate cards with numbers that are checked by the Luhn algorithm. You know how to check the card for validity. But how do you generate a card number so that it passes the validation test? It's very simple!

First, we need to generate an Account Identifier, which is unique to each card. Then we need to assign the Account Identifier to our BIN (Bank Identification Number). As a result, we get a 15-digit number 400000844943340, so we only have to generate the last digit, which is a checksum.

To find the checksum, it is necessary to find the control number for 400000844943340 by the Luhn algorithm. It equals 57 (from the example above). The final check digit of the generated map is `57+X`, where `X` is checksum. In order for the final card number to pass the validity check, the check number must be a multiple of 10, so `57+X` must be a multiple of 10. The only number that satisfies this condition is 3.

Therefore, the checksum is 3. So the total number of the generated card is 4000008449433403. The received card is checked by the Luhn algorithm.

You need to change the credit card generation algorithm so that they pass the Luhn algorithm.

Instruction

You should allow customers to create a new account in our banking system.

Once the program starts you should print the menu:

1. Create an account
2. Log into account
0. Exit

If the customer chooses 'Create an account', you should generate a new card number that satisfies all the conditions described above. Then you should generate a PIN code that belongs to the generated card number. PIN is a sequence of 4 digits; it should be generated in the range from 0000 to 9999.

If the customer chooses 'Log into account', you should ask to enter card information.

After the information has been entered correctly, you should allow the user to check the account balance; after creating the account, the balance should be 0. It should also be possible to log out of the account and exit the program.

Example

The symbol `>` represents the user input. Notice that it's not a part of the input.

```
1. Create an account
2. Log into account
0. Exit
>1

Your card has been created
```

```
Your card number:
4000004938320895
Your card PIN:
6826

1. Create an account
2. Log into account
0. Exit
>2

Enter your card number:
>4000004938320895
Enter your PIN:
>4444

Wrong card number or PIN!

1. Create an account
2. Log into account
0. Exit
>2

Enter your card number:
>4000004938320895
Enter your PIN:
>6826

You have successfully logged in!

1. Balance
2. Log out
0. Exit
>1

Balance: 0

1. Balance
2. Log out
0. Exit
>2

You have successfully logged out!

1. Create an account
2. Log into account
0. Exit
>0

Bye!
```


Write a program

Code Editor | IDE

IDE / Checking the plugin's status

Solve in IDE

src/banking/Main.java

```java
package banking;

import java.util.Random;
import java.util.Scanner;

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

        int[] cardNumber=new int[10];
        int[] PIN=new int[4];
        while(true){

            System.out.println("1. Create an account");
            System.out.println("2. Log into account");
            System.out.println("0. Exit");

            int n=in.nextInt();
            boolean exit=false;

            if(n==1){

                System.out.println("Your card has been created");
                System.out.println("Your card number: ");
                System.out.print(400000);

                Random generator=new Random();

                for(int i=0;i<10;++i) {
                    cardNumber[i]=generator.nextInt(10);
                }
                checkSumGenerator(cardNumber);
                System.out.println();

                System.out.println("Your card PIN: ");

                for(int i=0;i<4;++i){
                    PIN[i]=generator.nextInt(10);
                    System.out.print(PIN[i]);
```

```
                }
                System.out.println();

            }

            if(n==2){

                System.out.println("Enter your card number:");
                char[] enteredCardNumber=in.next().toCharArray();
                System.out.println("Enter your PIN:");
                char[] enteredPIN=in.next().toCharArray();
                boolean wrong=false;
                int[] enteredCardNumberInt=new int[10];
                if(enteredCardNumber.length==16) {
                    for (int i = 0; i < 10; ++i) {
                        enteredCardNumberInt[i] = enteredCardNumber[i + 6] - 48;
                    }
                }
                else wrong=true;

                for (int i = 0; i < 10; ++i) {
                    if (enteredCardNumberInt[i] != cardNumber[i]) {
                        wrong = true;
                        break;
                    }
                }

                for(int i=0;i<4;++i){
                    if(enteredPIN[i]-48!=PIN[i]){
                        wrong=true;
                        break;
                    }
                }


                if(!wrong){

                    System.out.println("You have successfully logged in!\n");
                    while(true) {

                        System.out.println("1. Balance");
                        System.out.println("2. Log out");
                        System.out.println("0. Exit");

                        int choice=in.nextInt();

                        if(choice==1){
                            System.out.println("\nBalance: 0\n");
                        }
                        if(choice==2){
                            System.out.println("You have successfully logged out!\n");
                            break;
                        }
                        if(choice==0){
                            exit=true;
                            break;
                        }

                    }

                }
                else{
                    System.out.println("Wrong card number or PIN!\n");
                }

            }

            if(exit || n==0){
                System.out.println("Bye!");
                break;
            }

        }

    }

    public static void checkSumGenerator(int[] cardNumber){
        int sum=8;
        for(int i=0;i<9;++i){
            int temp=cardNumber[i];
            if(i%2==0) temp*=2;
            if(temp>9) temp-=9;
            sum+=temp;
            System.out.print(cardNumber[i]);
        }
        cardNumber[9]=10-sum%10;
        System.out.print(cardNumber[9]);
    }
}
```

src/banking/Main.class

```

  // IntelliJ API Decompiler stub source generated from a class file
  // Implementation of methods is not available

public class Main {
    public Main() { /* compiled code */ }

    public static void main(java.lang.String[] strings) { /* compiled code */ }
}
```

✓ Correct

20 users liked this problem. 0 didn't like it. What about you?

Continue Solutions (32)

Show discussion (64)