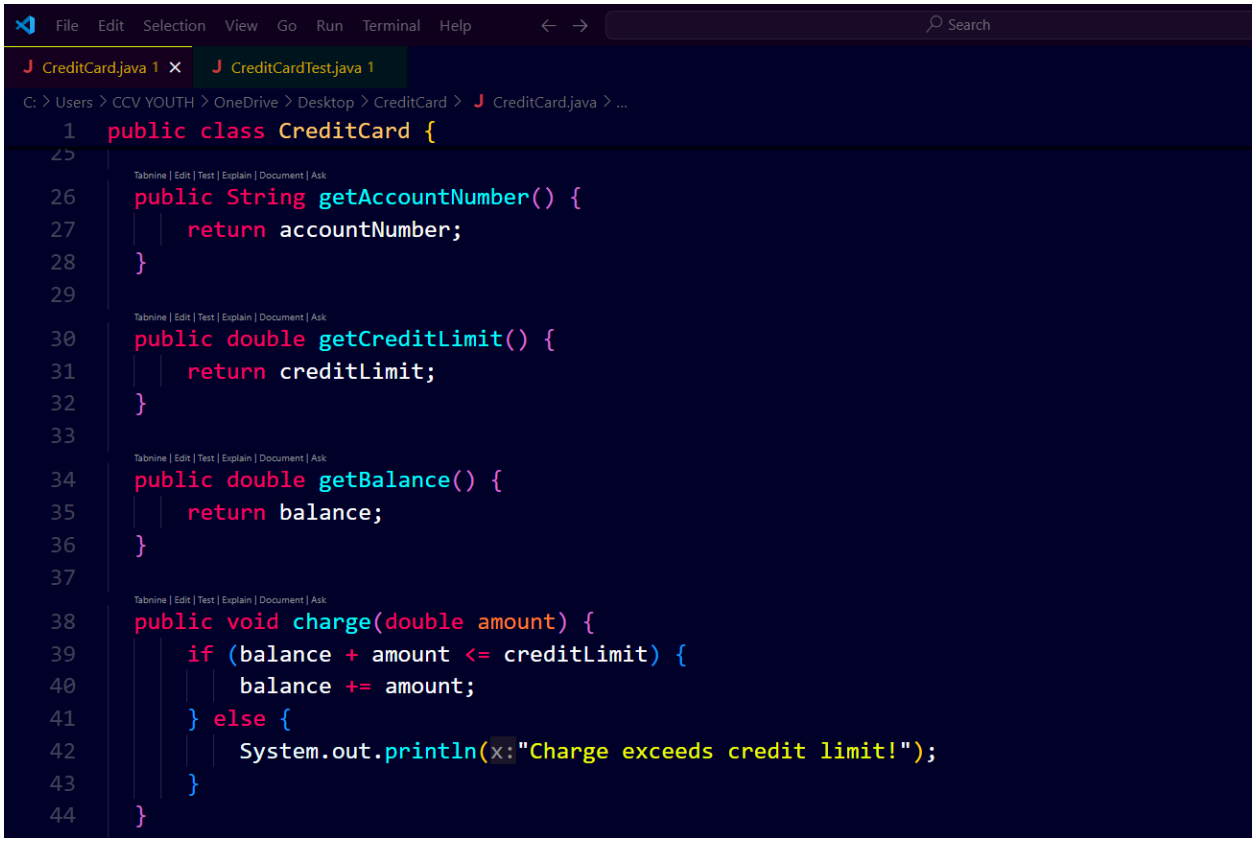
Good early morning Sir Gayo time check 4:am right now.File 1: CreditCard.java

```java
public class CreditCard {
  private String customerName;
  private String bankName;
  private String accountNumber;
  private double creditLimit;
  private double balance;

  // Constructor
  public CreditCard(String customerName, String bankName, String accountNumber, double creditLimit, double balance) {
      this.customerName = customerName;
      this.bankName = bankName;
      this.accountNumber = accountNumber;
      this.creditLimit = creditLimit;
      this.balance = balance;
  }

  // Getter and Setter methods
  public String getCustomerName() {
      return customerName;
  }

  public String getBankName() {
      return bankName;
  }
```



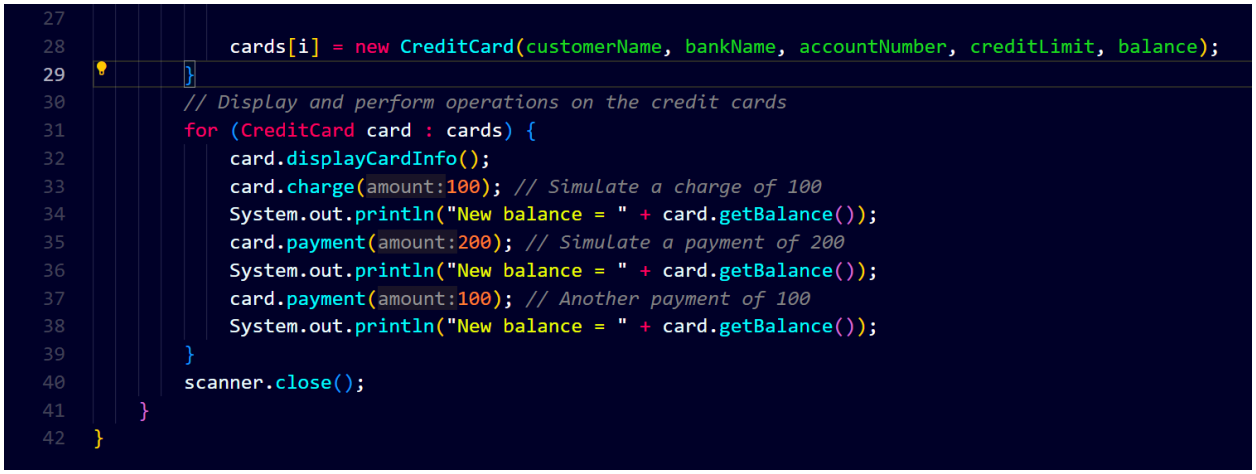
```java
public class CreditCard {

  public String getAccountNumber() {
      return accountNumber;
  }

  public double getCreditLimit() {
      return creditLimit;
  }

  public double getBalance() {
      return balance;
  }

  public void charge(double amount) {
      if (balance + amount <= creditLimit) {
          balance += amount;
      } else {
          System.out.println(x:"Charge exceeds credit limit!");
      }
  }
```

This file defines a class called `CreditCard`, which represents a credit card. The class has several attributes and methods that describe and manipulate credit card information.

Key components of `CreditCard.java`:

1. **Attributes (Variables)**:
   - `customerName`: The name of the cardholder.
   - `bankName`: The name of the bank that issued the card.
   - `accountNumber`: The unique number assigned to the cardholder's account.
   - `creditLimit`: The maximum amount that can be charged to the card.
   - `balance`: The current balance of the card (how much is owed).
2. **Constructor**:
   - This is used to create a new `CreditCard` object with the cardholder's name, bank name, account number, credit limit, and initial balance.
3. **Methods**:
   - **charge(amount)**: Adds a specified amount to the balance (this simulates making a purchase with the credit card).
   - **payment(amount)**: Reduces the balance by a specified amount (this simulates making a payment to pay off part of the balance).
   - **displayCardInfo()**: Displays the cardholder's details, including name, bank, account number, balance, and credit limit.

File 2: CreditCardTest.java



This is the main file that runs the program. It interacts with the user to get information about

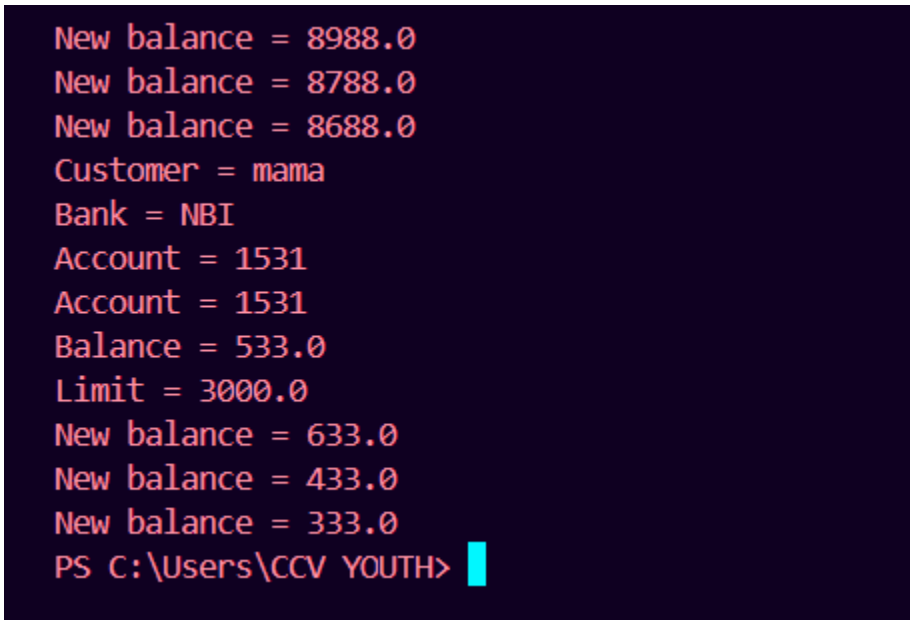
multiple credit cards, performs operations on those cards (charging and making payments), and displays the updated information.

Key components of `CreditCardTest.java`:

1. **Input from the user**:
    - The program asks the user to enter the number of credit cards they want to simulate.
    - For each card, the user is prompted to enter the customer's name, bank name, account number, credit limit, and initial balance.
2. **Card Operations**:
    - For each card, the program performs a series of actions:
        - It displays the initial information (balance, credit limit, etc.).
        - It "charges" $100 to the card (simulating a purchase).
        - It then makes a payment of $200 to the card.
        - Finally, it makes another payment of $100 to the card.
3. **Displaying Results**:
    - After each action (charge or payment), the updated balance of the card is shown.

The output

```
Enter the number of credit cards:

Card 1:
Customer name: MARK
Bank name: PDO
Account number: 2413 3142 3143 1413
Credit limit: 5000
Initial balance: 2000

Card 2:
Customer name: JESUS CHRIST
Bank name: HEAVEN
Account number: 1234 1234 1234 1234
Credit limit: 9999
Initial balance: 8888

Card 3:
Customer name: mama
Bank name: NBI
Account number: 1531
Credit limit: 3000
Initial balance: 533
Customer = MARK
Bank = PDO
Account = 2413 3142 3143 1413
Balance = 2000.0
Limit = 5000.0
New balance = 2100.0
New balance = 1900.0
New balance = 1800.0
Customer = JESUS CHRIST
Bank = HEAVEN
Account = 1234 1234 1234 1234
Balance = 8888.0
Limit = 9999.0
New balance = 8988.0
New balance = 8788.0
New balance = 8688.0
```



How It Works Together:

1. The `CreditCardTest.java` file creates multiple `CreditCard` objects, each representing a credit card with specific details (such as the cardholder's name, bank, etc.).
2. For each card, the program performs charges and payments, updating the balance.
3. After each operation (charge or payment), it shows the new balance of the card.

Summary in Simple Terms:

1. **Card 1 (MARK)**:
    - Starts with a balance of $2000.
    - A charge of $100 is added, making the balance $2100.
    - A $200 payment reduces it to $1900.
    - Another $100 payment reduces it to $1800.
2. **Card 2 (JESUS CHRIST)**:
    - Starts with a balance of $8888.
    - A charge of $100 is added, making the balance $8988.
    - A $200 payment reduces it to $8788.
    - Another $100 payment reduces it to $8688.
3. **Card 3 (mama)**:
    - Starts with a balance of $533.
    - A charge of $100 is added, making the balance $633.

- A $200 payment reduces it to $433.
- Another $100 payment reduces it to $333.

In each case, the program simulates adding charges and making payments, showing how the balance changes after each transaction.