NAME : VINIT
UID : 22BCS15523
SECTION : 22BCS_IOT-617-"B"

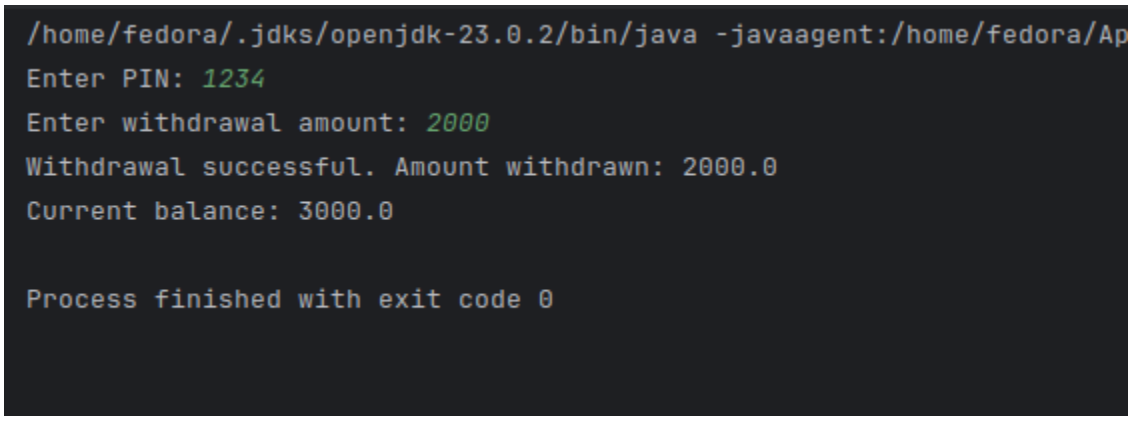
AIM : Write an java program to simulate an ATM withdrawal system.
The program should:
ask the user to enter their pin.
Allow withdrawl if the pin is correct and the balance is sufficient.
Ensure the system always shows the remaining balance, even if an exception occurs.

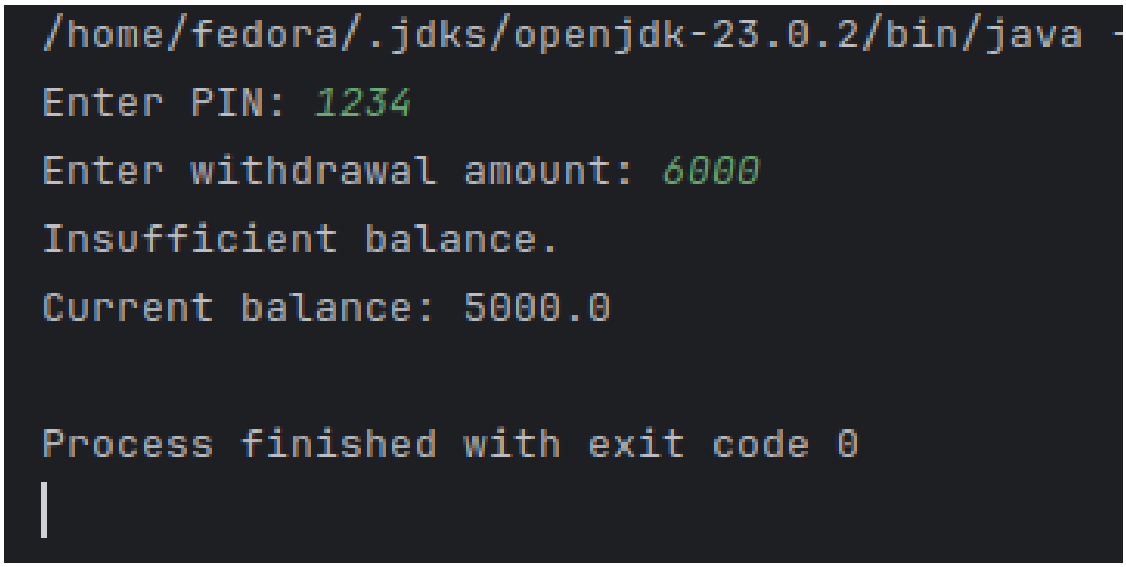
Code :

```
import java.util.Scanner;

class InvalidPinException extends Exception {
    public InvalidPinException(String message) {
        super(message);
    }
}

class InsufficientBalanceException extends Exception {
    public InsufficientBalanceException(String message) {
        super(message);
    }
}

public class Main {
    private static final int CORRECT_PIN = 1234;
    private static double balance = 5000; // Initial balance

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

        try {
            System.out.print("Enter PIN: ");
            int enteredPin = scanner.nextInt();

            if (enteredPin != CORRECT_PIN) {
                throw new InvalidPinException("Invalid PIN.");
            }

            System.out.print("Enter withdrawal amount: ");
            double amount = scanner.nextDouble();

            if (amount > balance) {
                throw new InsufficientBalanceException("Insufficient balance.");
            }
```

```
            balance -= amount;
            System.out.println("Withdrawal successful. Amount withdrawn: " + amount);
        } catch (InvalidPinException | InsufficientBalanceException e) {
            System.out.println(e.getMessage());
        } catch (Exception e) {
            System.out.println("Invalid input. Please enter a valid amount.");
        } finally {
            System.out.println("Current balance: " + balance);
            scanner.close();
        }
    }
}
```

Output :





Learning Outcome:

- Understand how to implement a basic simulation of an ATM system in Java using simple logic and error handling.
- Demonstrate how to manage balance updates and display information based on user input.
- Understand how to handle user authentication in Java using loops.
- Learn to use exception handling (`try-catch-finally`) to manage runtime errors like insufficient funds.