Assignment No: 04

Department of Computer Science

Iqra University Islamabad

Object Oriented Programming

Maqsood Ahmed

ID: 38186

Problem # 1: [CLO2]

Source Code:

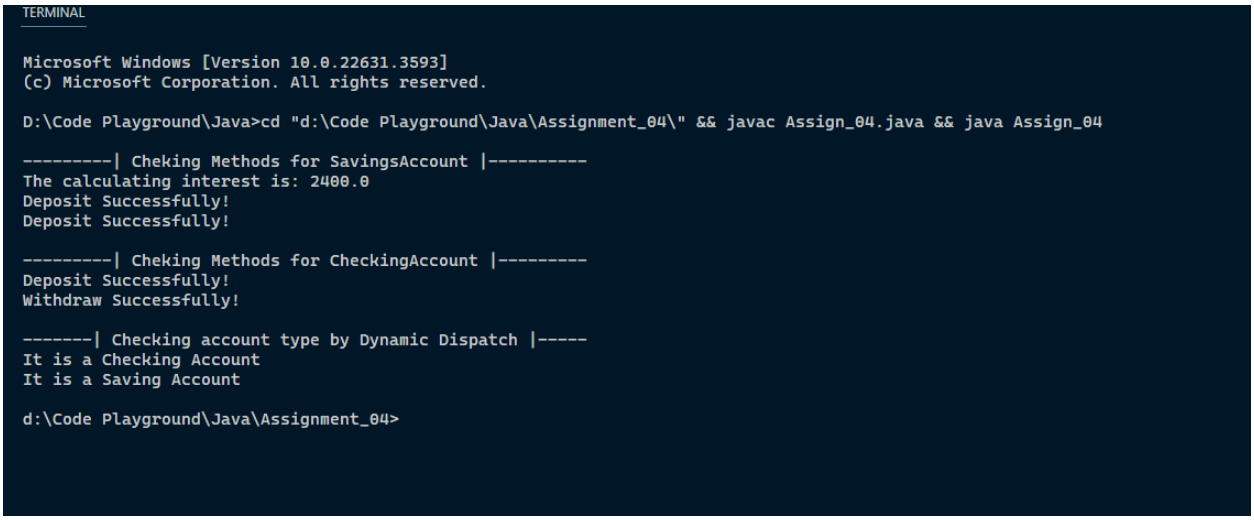
```
public class Assign_04 {
   public static void main(String[] args) {
      SavingsAccount savingsAccount = new SavingsAccount(60000, 0.04); // 4% percent interest Rate
      System.out.println("\n---------| Cheking Methods for SavingsAccount |----------");
      System.out.println("The calculating interest is: " + savingsAccount.calculateInterest());
      if(savingsAccount.credit(5000)) {
         System.out.println("Deposit Successfully!");
      }
      if(savingsAccount.debit(1000)) {
         System.out.println("Deposit Successfully!");
      }
      CheckingAccount checkingAccount = new CheckingAccount(70000, 23.44);
      System.out.println("\n---------| Cheking Methods for CheckingAccount |---------");
      if(checkingAccount.credit(3000)) {
         System.out.println("Deposit Successfully! ");
      }
      if(checkingAccount.debit(2000)) {
         System.out.println("Withdraw Successfully! ");
      }
      System.out.println("\n-------| Checking account type by Dynamic Dispatch |-----");
      // dynamic dispatch;
      Account dynamicDispatch;
      dynamicDispatch = checkingAccount;
      checkingAccount.printAccountType();
      dynamicDispatch = savingsAccount;
```

```
        savingsAccount.printAccountType();
    }
}
abstract class Account {
    protected double balance;
    public Account(double balance) {
        if (balance < 0.0) {
            System.out.println("You entered Invalid Balance!");
            this.balance = 0.0;
        } else {
            this.balance = balance;
        }
    }
    public boolean credit(double amount) {
        this.balance += amount;
        return true;
    }
    public boolean debit(double amount) {
        if (balance < amount) {
            System.out.println("Debit amount exceeded account balance.");
            return false;
        }
        this.balance -= amount;
        return true;
    }
    public double getBalance() {
        return balance;
    }
```

```
    abstract public void printAccountType();
}
class SavingsAccount extends Account {
    public double interestRate;
    public SavingsAccount(double balance, double interestRate) {
        super(balance);
        this.interestRate = interestRate;
    }
    public double calculateInterest() {
        return interestRate * super.balance;
    }
    @Override
    public void printAccountType() {
        System.out.println("It is a Saving Account");
    }
}
class CheckingAccount extends Account {
    private double feePerTranscation;
    public CheckingAccount(double balance, double feePerTranscation) {
        super(balance);
        this.feePerTranscation = feePerTranscation;
    }
    @Override
    public boolean credit(double amount) {
        super.balance += amount;
        super.balance -= feePerTranscation;
        return true;
    }
    @Override
```

```
    public boolean debit(double amount) {
        if (balance < amount) {
            System.out.print("Debit amount exceeded account balance.");
            return false;
        }
        super.balance -= amount;
        super.balance -= feePerTranscation;
        return true;
    }
    @Override
    public void printAccountType() {
        System.out.println("It is a Checking Account");
    }
}
```

OUTPUT:



The End