BANKING SYSTEM-OOPS,COLLECTIONS AND EXCEPTION HANDLING

TASK 8: INHERITANCE AND POLYMORPHISM

Overloading the deposit and withdraw methods in Account Class

- Method Overloading allows multiple methods with the same name but different parameter types.
- Three versions of deposit and withdraw methods should be implemented for float, int, and double.

Creating Subclasses (SavingsAccount and CurrentAccount)

SavingsAccount:

- Has an additional attribute interestRate.
- Overrides calculateInterest() to calculate interest based on balance and interest rate.

CurrentAccount:

- Has an additional attribute overdraftLimit.
- Overrides withdraw() to allow overdraft within a set limit.

Implementing Bank Class

- Displays a menu allowing users to create either a Savings Account or Current Account.
- Uses a switch-case to handle user selection.
- Performs deposit, withdrawal, and interest calculation operations.

ACCOUNT CLASS:

```
public class Account {
    private int accountNumber;
    private String accountType;
    private double balance;

    public Account() {}

    public Account(int accountNumber, String accountType, double balance) {
        this.accountNumber = accountNumber;
        this.accountType = accountType;
        this.balance = balance;
    }
```

```
  public int getAccountNumber() {
     return accountNumber;
  }
  public void setAccountNumber(int accountNumber) {
     this.accountNumber = accountNumber;
  }

  public String getAccountType() {
     return accountType;
  }
  public void setAccountType(String accountType) {
     this.accountType = accountType;
  }

  public double getBalance() {
     return balance;
  }
  public void setBalance(double balance) {
     this.balance = balance;
  }

public void deposit(float amount) {
     if (amount > 0) {
        balance += amount;
        System.out.println("Deposited (float): " + amount);
     }
  }

  public void deposit(int amount) {
     if (amount > 0) {
        balance += amount;
        System.out.println("Deposited (int): " + amount);
     }
  }

  public void deposit(double amount) {
     if (amount > 0) {
        balance += amount;
        System.out.println("Deposited (double): " + amount);
     }
  }

  public void withdraw(float amount) {
     if (amount > 0 && balance >= amount) {
        balance -= amount;
        System.out.println("Withdrawn (float): " + amount);
     } else {
        System.out.println("Insufficient balance!");
     }
  }
```

```
    public void withdraw(int amount) {
        if (amount > 0 && balance >= amount) {
            balance -= amount;
            System.out.println("Withdrawn (int): " + amount);
        } else {
            System.out.println("Insufficient balance!");
        }
    }

    public void withdraw(double amount) {
        if (amount > 0 && balance >= amount) {
            balance -= amount;
            System.out.println("Withdrawn (double): " + amount);
        } else {
            System.out.println("Insufficient balance!");
        }
    }

    public void printAccountInfo() {
        System.out.println("Account Number: " + accountNumber);
        System.out.println("Account Type: " + accountType);
        System.out.println("Balance: " + balance);
    }
}
```

CURRENT ACCOUNT CLASS EXTENDS ACCOUNT CLASS:

```
public class CurrentAccount extends Account{

    private static final double OVERDRAFT_LIMIT = 5000;

    public CurrentAccount(int accountNumber, double balance) {
        super(accountNumber, "Current", balance);
    }

    @Override
    public void withdraw(double amount) {
        if (amount > 0 && (getBalance() + OVERDRAFT_LIMIT) >= amount) {
            setBalance(getBalance() - amount);
            System.out.println("Withdrawn (Current Account): " + amount);
        } else {
            System.out.println("Overdraft limit exceeded!");
        }
    }
}
```

SAVINGS ACCOUNT CLASS EXTENDS ACCOUNT CLASS:

```
public class SavingsAccount extends Account {
   private double interestRate = 0.045;

   public SavingsAccount(int accountNumber, double balance) {
      super(accountNumber, "Savings", balance);
   }


   public void calculateInterest() {
      double interest = getBalance() * interestRate;
      setBalance(getBalance() + interest);
      System.out.println("Interest added: " + interest);
   }
}
```

BANK CLASS:

```
import java.util.*;
public class Bank {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
       Account account = null;

       System.out.println("Select Account Type:");
       System.out.println("1. Savings Account");
       System.out.println("2. Current Account");
       int choice = scanner.nextInt();

       System.out.print("Enter Account Number: ");
       int accountNumber = scanner.nextInt();
       System.out.print("Enter Initial Balance: ");
       double balance = scanner.nextDouble();

       switch (choice) {
          case 1:
             account = new SavingsAccount(accountNumber, balance);
             break;
          case 2:
             account = new CurrentAccount(accountNumber, balance);
             break;
          default:
             System.out.println("Invalid choice!");
             return;
       }

       account.printAccountInfo();

       System.out.print("Enter Deposit Amount: ");
```

```
        double depositAmount = scanner.nextDouble();
        account.deposit(depositAmount);

        System.out.print("Enter Withdraw Amount: ");
        double withdrawAmount = scanner.nextDouble();
        account.withdraw(withdrawAmount);

        if (account instanceof SavingsAccount) {
            ((SavingsAccount) account).calculateInterest();
        }

        account.printAccountInfo();
        scanner.close();
    }
}
```

OUTPUT:

```
Select Account Type:
1. Savings Account
2. Current Account
1
Enter Account Number: 5000
Enter Initial Balance: 8000
Account Number: 5000
Account Type: Savings
Balance: 8000.0
Enter Deposit Amount: 3000
Deposited (double): 3000.0
Enter Withdraw Amount: 1000
Withdrawn (double): 1000.0
Interest added: 450.0
Account Number: 5000
Account Type: Savings
Balance: 10450.0
BUILD SUCCESSFUL (total time: 26 seconds)
```

```
Select Account Type:
1. Savings Account
2. Current Account
2
Enter Account Number: 4000
Enter Initial Balance: 6000
Account Number: 4000
Account Type: Current
Balance: 6000.0
Enter Deposit Amount: 2000
Deposited (double): 2000.0
Enter Withdraw Amount: 1000
Withdrawn (Current Account): 1000.0
Account Number: 4000
Account Type: Current
Balance: 7000.0
BUILD SUCCESSFUL (total time: 21 seconds)
```