

## Day 3:

### 1.overloading:

```
class Calc {
    public String area(String a,int b) {
        return a + b*b; //order
    }
    public String area(double r,String a) {
        return a + Math.PI*r*r; //order
    }
    public double area(double a, double b) {
        return 0.5*a*b; //type of
    }
    public int area(int a, int b) {
        return a*b; //number of , type of
    }
    public int area(int a, int b, int c) {
        return a*b*c; //number of
    }
}

public class Area {

    public static void main(String[] args) {
        Calc obj = new Calc();
        System.out.println("Area of " + obj.area("Square",12));
        System.out.println("Area of " + obj.area(7.5,"Circle"));

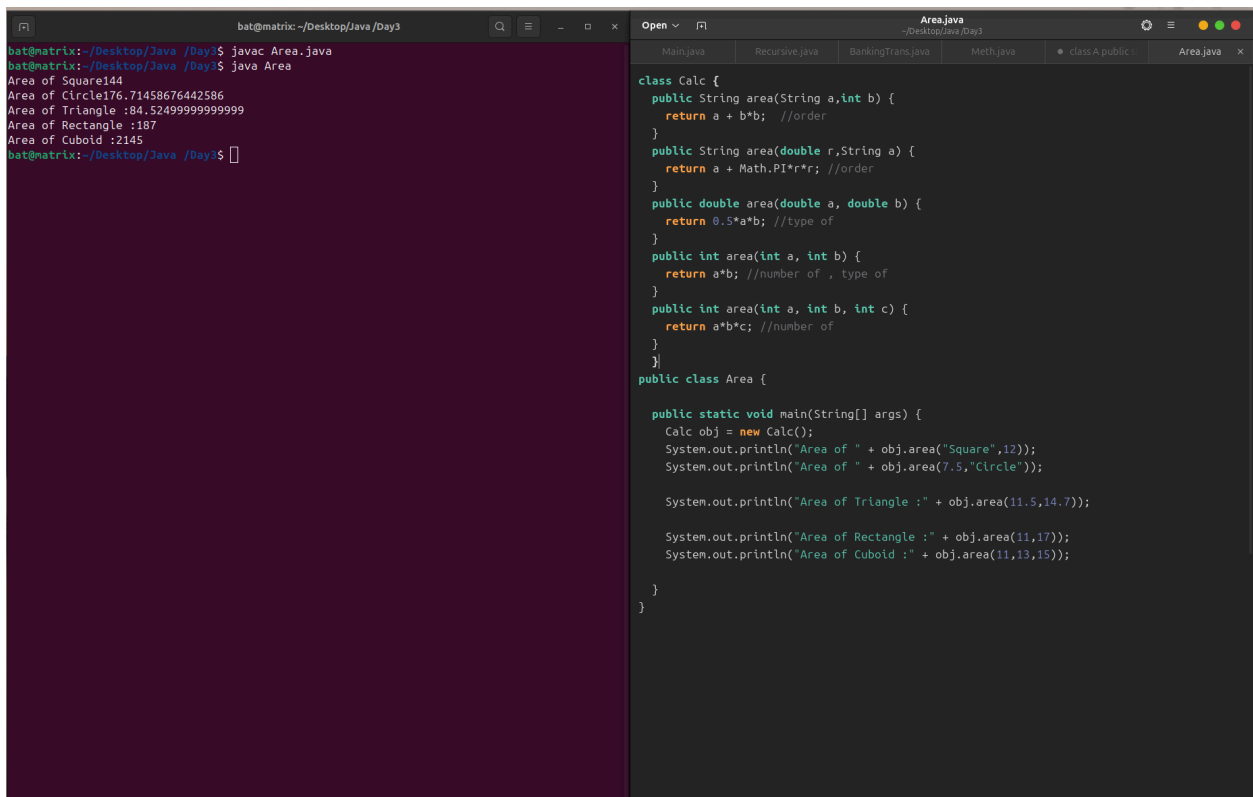
        System.out.println("Area of Triangle :" +
obj.area(11.5,14.7));

        System.out.println("Area of Rectangle :" +
obj.area(11,17));
        System.out.println("Area of Cuboid :" +
obj.area(11,13,15));

    }
}
```

Output:

```
Area of Square144
Area of Circle176.71458676442586
Area of Triangle :84.52499999999999
Area of Rectangle :187
Area of Cuboid :2145
```



```
bat@matrix:~/Desktop/Java/Day3$ javac Area.java
bat@matrix:~/Desktop/Java/Day3$ java Area
Area of Square:144
Area of Circle:176.71458676442586
Area of Triangle :84.52499999999999
Area of Rectangle :187
Area of Cuboid :2145
bat@matrix:~/Desktop/Java/Day3$
```

```
class Calc {
    public String area(String a,int b) {
        return a + b*b; //order
    }
    public String area(double r,String a) {
        return a + Math.PI*r*r; //order
    }
    public double area(double a, double b) {
        return 0.5*a*b; //type of
    }
    public int area(int a, int b) {
        return a*b; //number of , type of
    }
    public int area(int a, int b, int c) {
        return a*b*c; //number of
    }
}

public class Area {

    public static void main(String[] args) {
        Calc obj = new Calc();
        System.out.println("Area of " + obj.area("Square",12));
        System.out.println("Area of " + obj.area(7.5,"Circle"));

        System.out.println("Area of Triangle : " + obj.area(11.5,14.7));

        System.out.println("Area of Rectangle : " + obj.area(11,17));
        System.out.println("Area of Cuboid : " + obj.area(11,13,15));

    }
}
```

**2.Bank Transaction with a note added and transaction history using linked list and tried to include method overloading.**

```
import java.util.*;

public class BankingTrans {

    static class Account {
        String accno;
        String acchname;
        double balance;
        LinkedList<String> transactions;

        public Account(String accno, String acchname, double balance)
        {
            this.accno = accno;
            this.acchname = acchname;
            this.balance = balance;
            this.transactions = new LinkedList<>();
            transactions.add("Opening Balance : Rs. " + balance);
        }
    }
}
```

```

    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            String msg = "Deposited Rs. " + amount + "\n
Available Balance Rs. " + balance;
            transactions.add(msg);
            System.out.println(msg);
        } else {
            System.out.println("Enter valid amount");
        }
    }

    public void withdraw(double amount) {
        if (amount > 0 && balance >= amount) {
            balance -= amount;
            String msg = "Withdrawal of Rs. " + amount + "
Success"+ "\n Current Balance Rs. " + balance;
            transactions.add(msg);
            System.out.println(msg);
        } else if (amount <= 0)
            System.out.println("Enter positive values only");
        else
            System.out.println("Insufficient balance");
    }

    public void transfer(double amount, String reason) {
        if (amount > 0 && balance >= amount) {
            balance -= amount;
            String msg = "Transfer of Rs. " + amount + " Success
" + " Reason :" + reason ;
            transactions.add(msg);
            System.out.println(msg);
        } else if (amount <= 0)
            System.out.println("Enter positive values only");
        else
            System.out.println("Insufficient balance");
    }

    public void transfer(String reason, double amount) {
        if (amount > 0 && balance >= amount) {
            balance -= amount;
            String msg = "Transfer of Rs. " + amount + " Success
" + " Reason :" + reason ;
            transactions.add(msg);
            System.out.println(msg);
        }
    }

```

```

        } else if (amount <= 0)
            System.out.println("Enter positive values only");
        else
            System.out.println("Insufficient balance");
    }

    public void addmoney(double amount) {
        if (amount > 0) {
            balance += amount;
            String msg = "Updated Balance of Receiver: Rs. " +
balance;

            transactions.add(msg);
            System.out.println(msg);
        } else {
            System.out.println("Enter valid amount");
        }
    }

    public void showTransactions() {
        System.out.println("\n====Transaction History==== \n
Account Number : "+ accno);
        for (String t : transactions) {
            System.out.println(t);
        }
    }

    public double getBalance() {
        return balance;
    }

    public String getAccno() {
        return accno;
    }

    public String getAccinfo() {
        return "Account Number: " + accno + ", Account Holder
Name: " + acchname + ", Balance: Rs. " + balance;
    }
}

static class Bank {
    public List<Account> accounts;

    public Bank() {
        this.accounts = new ArrayList<>();
    }
}

```

```

    }

    public void addAccount(Account account) {
        accounts.add(account);
        System.out.println("Adding account to the Bank");
    }

    public Account findAccount(String accno) {
        for (Account account : accounts) {
            if (account.getAccno().equals(accno)) {
                return account;
            }
        }
        return null;
    }
}

public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    Bank bank = new Bank();
    int choice;

    do {
        System.out.println("\n===== Bank ===== \n 1. Create
Account \n 2. Deposit \n 3. Withdraw \n 4. View Account \n 5. Perform
Transaction \n 6. View Transactions \n 7.Exit \n Enter your Choice :
");

        choice = sc.nextInt();
        sc.nextLine();

        switch (choice) {
            case 1:
                System.out.print("Enter Account Number: ");
                String accNo = sc.nextLine();
                System.out.print("Enter Account Holder Name: ");
                String accName = sc.nextLine();
                System.out.print("Enter Initial Balance: ");
                double initBal = sc.nextDouble();
                sc.nextLine();
                Account acc = new Account(accNo, accName,
initBal);

                bank.addAccount(acc);
                break;

            case 2:

```

```

        System.out.print("Enter Account Number: ");
        String depAccNo = sc.nextLine();
        Account depAcc = bank.findAccount(depAccNo);
        if (depAcc != null) {
            System.out.print("Enter amount to deposit:
");

            double amount = sc.nextDouble();
            sc.nextLine();
            depAcc.deposit(amount);
        } else {
            System.out.println("Account not found.");
        }
        break;

    case 3:
        System.out.print("Enter Account Number: ");
        String withAccNo = sc.nextLine();
        Account withdrawAcc =
bank.findAccount(withAccNo);
        if (withdrawAcc != null) {
            System.out.print("Enter amount to withdraw:
");

            double amt = sc.nextDouble();
            sc.nextLine();
            withdrawAcc.withdraw(amt);
        } else {
            System.out.println("Account not found.");
        }
        break;

    case 4:
        System.out.print("Enter Account Number: ");
        String infoAccNo = sc.nextLine();
        Account infoAcc = bank.findAccount(infoAccNo);
        if (infoAcc != null) {
            System.out.println("Account Found!");
            System.out.println(infoAcc.getAccinfo());
        } else {
            System.out.println("Account not found.");
        }
        break;

    case 5:
        System.out.print("Enter Sender Account Number:
");

```

```

        String senderAccNo = sc.nextLine();
        Account senderAcc =
bank.findAccount(senderAccNo);
        if (senderAcc != null) {
            System.out.print("Enter Receiver Account
Number: ");

            String receiverAccNo = sc.nextLine();
            Account receiverAcc =
bank.findAccount(receiverAccNo);
            if (receiverAcc != null) {
                System.out.print("Enter transaction
details: ");

                double amt = sc.nextDouble();
                sc.nextLine();
                String note = sc.nextLine();
                senderAcc.transfer(amt, note);
                receiverAcc.addmoney(amt);
            } else {
                System.out.println("Receiver Account not
found.");
            }
        } else {
            System.out.println("Sender Account not
found.");
        }
        break;

case 6:
    System.out.println("Enter Account Number: ");
    String taccno = sc.nextLine();
    Account tacc = bank.findAccount(taccno);
    if (tacc != null) {
        tacc.showTransactions();
    }
    else {
        System.out.println("Account not found...");
    }
    break;

case 7:
    System.out.println("*** Thanking you! ***");
    break;

default:

```

```

        System.out.println("Enter a valid option only
        ...");
    }

    } while (choice != 7);

    sc.close();
}
}

```

The screenshot shows a Java IDE with two windows. The left window is the console, showing the output of the BankingTrans.java program. The right window is the BankingTrans.java file, showing the source code.

**BankingTrans.java Source Code:**

```

import java.util.*;

public class BankingTrans {

    static class Account {
        String accno;
        String acchname;
        double balance;
        LinkedList<String> transactions;

        public Account(String accno, String acchname, double balance) {
            this.accno = accno;
            this.acchname = acchname;
            this.balance = balance;
            this.transactions = new LinkedList<>();
            transactions.add("Opening Balance : Rs. " + balance);
        }

        public void deposit(double amount) {
            if (amount > 0) {
                balance += amount;
                String msg = "Deposited Rs. " + amount + "\n Available Balance Rs. " +
                    balance;
                transactions.add(msg);
                System.out.println(msg);
            } else {
                System.out.println("Enter valid amount");
            }
        }

        public void withdraw(double amount) {
            if (amount > 0 && balance >= amount) {
                balance -= amount;
                String msg = "Withdrawal of Rs. " + amount + " Success"+ "\n Current
                    Balance Rs. " + balance;
                transactions.add(msg);
                System.out.println(msg);
            } else if (amount <= 0) {
                System.out.println("Enter positive values only");
            } else {
                System.out.println("Insufficient balance");
            }
        }
    }
}

```

**Console Output:**

```

bat@matrix:~/Desktop/Java /Day3$ java BankingTrans
===== Bank =====
1. Create Account
2. Deposit
3. Withdraw
4. View Account
5. Perform Transaction
6. View Transactions
7.Exit
Enter your Choice :
1
Enter Account Number: 123
Enter Account Holder Name: asd
Enter Initial Balance: 235468
Adding account to the Bank

===== Bank =====
1. Create Account
2. Deposit
3. Withdraw
4. View Account
5. Perform Transaction
6. View Transactions
7.Exit
Enter your Choice :
1
Enter Account Number: 234
Enter Account Holder Name: areg
Enter Initial Balance: 23565734
Adding account to the Bank

===== Bank =====
1. Create Account
2. Deposit
3. Withdraw
4. View Account
5. Perform Transaction
6. View Transactions
7.Exit
Enter your Choice :
5
Enter Sender Account Number: 123
Enter Receiver Account Number: 234
Enter transaction details: 1234
asdfg
Transfer of Rs. 1234.0 Success Reason :asdfg
Updated Balance of Receiver: Rs. 2.3566968E7

===== Bank =====
1. Create Account

```

Output:

```

===== Bank =====
1. Create Account
2. Deposit
3. Withdraw
4. View Account
5. Perform Transaction
6. View Transactions
7.Exit
Enter your Choice :
1
Enter Account Number: 123

```



Enter Account Holder Name: asd  
Enter Initial Balance: 235468  
Adding account to the Bank

===== Bank =====

1. Create Account
2. Deposit
3. Withdraw
4. View Account
5. Perform Transaction
6. View Transactions
- 7.Exit

Enter your Choice :

1

Enter Account Number: 234  
Enter Account Holder Name: areg  
Enter Initial Balance: 23565734  
Adding account to the Bank

===== Bank =====

1. Create Account
2. Deposit
3. Withdraw
4. View Account
5. Perform Transaction
6. View Transactions
- 7.Exit

Enter your Choice :

5

Enter Sender Account Number: 123  
Enter Receiver Account Number: 234  
Enter transaction details: 1234  
asdfg  
Transfer of Rs. 1234.0 Success Reason :asdfg  
Updated Balance of Receiver: Rs. 2.3566968E7

===== Bank =====

1. Create Account
2. Deposit
3. Withdraw
4. View Account
5. Perform Transaction
6. View Transactions
- 7.Exit

Enter your Choice :

5

Enter Sender Account Number: 1233

Sender Account not found.

===== Bank =====

1. Create Account
2. Deposit
3. Withdraw
4. View Account
5. Perform Transaction
6. View Transactions
- 7.Exit

Enter your Choice :

5

Enter Sender Account Number: 123

Enter Receiver Account Number: 234

Enter transaction details: asdaf

Exception in thread "main" java.util.InputMismatchException  
at java.base/java.util.Scanner.throwFor(Scanner.java:947)  
at java.base/java.util.Scanner.next(Scanner.java:1602)  
at java.base/java.util.Scanner.nextDouble(Scanner.java:2573)  
at BankingTrans.main(BankingTrans.java:192)