WAP to overload the method print that prints sum of n natural numbers in a given range when 2 parameters are passed.

→
```
class overload
{
  void print (int n)
  {
    int sum=0;
    for(int i=1; i<=n; i++)
  {
    sum=sum+i;
  }
  System.out.println("Sum of "+n+" natural no is");
}
void print (int m, int n)
{
System.out.printf("Prime numbers in the range are");
for(int i=m; i<=n; i++)
{
  int flag=0;
  for(int j=2; j<=i/2; j++)
{
  if(i%j==0)
{
  flag=1;
  break;
}
}
  if(flag==0)
    System.out.println(i);
}
}
```

Date: / /

```
void display ()
{
  System.out.print ("name "+""+" phone no."+" "+
  +total");

  System.out.println(c_nam+""+c_ph+" "+total);
  System.out.println()

}
}
class cDemo
{
  public static void main (String[] args)
{
  Grocery g1 = new grocery ("Rama", "8060308
  Grocery g2 = new grocery ("Shama", "7689325
  Grocery g3 = new grocery ("Bham", "ac32587

g1.calc(2,2,1);
g1.display();

g2.calc(3,5,2);
g2.display();

g3.calc(1,1,0.5);
g3.display();

}
}
```

```
class overload Demo{

Public static void main (string [] args) {
ovaload o= new overload();
o.print(5);
o.print(7,13);
}
}
```

2. Write a java program to create a class grocery that uses the variables c_name & c_phone. Create a method to accept 3 parameters to specify quantity of dal, pulses, Sugar return total price, Display name, ph no and bill

```
class grocery {

String c_name;
String c_ph;
double total;
grocery(String c_name, String c_ph)
{
this.c_name = c_name;
this.c_ph = c_ph;
}

Void calc (double q_dal, double q_pulses, doubl
{
total = q_dal * 100 + q_pulses * 80 + q_Sugar * 60;
}
```

```
import java.util.Scanner

class Account
{
    String customerName;
    long accountNumber;
    String accountType;
    double balance;

    Public Account(String customerName,
    long accountNumber, String accountType, double balance){

    this.customerName = customerName;
    this.accountNumber = accountNumber;
    this.accountType = accountType;

}
 Public void deposit(double amt)
{
    balance += amount;
    System.out.println("Deposit of
    $" + amount + " successful");
    displayBalance();

}
Public void withdraw(double amt)
{
  if(amount <= balance)
{ balance -= amount
   System.out.println("successful");
}
displayBalance();
}
```

Program (1)

Father, son age

```
class son extends Father {
Private int SonAge;

Son(int fatherAge, int SonAge) throws WrongAge {
super(fatherAge);

if (SonAge >= FatherAge) {
throw new WrongAge();
}
this.sonAge = SonAge;
}
int getSonAge = SonAge;
}
}

Public class Main {
Public static void main (String[] args) {
try {

Father father = new Father(50);
System.out.println("Fathers age:" + Father.
Son son = new Son(father
system.out.println("Sons Age" + Son.getSon
}

catch (WrongAge e) {
System.out.println(e.getMessage());
}
}
}
```

```
class savacct extends Account
{
  private double interest rate = 0.05;
  public savacct (String customerName,
    long accountNumber, double balance)
  {
    super (customer Name, account Number);
  }

  public void compute interest ()
  {
    double interest = balance * interestRate;
    balance += interest;
    System.out.println ("Interest of $" + interest +
    display Balance ()
  }
  }
Public class two {
Public static void main (String[] args)
{
  Scanner Scanner = new Scanner (System.in
Sav acct Savings Account = new savacct
("John Doe", 123456789, 5000);
Saving Account . display balance();
Saving Account . deposit (1000);
Saving Account . compute Yields ();
Savings Account . withdraw (2000);
```

19.01.24

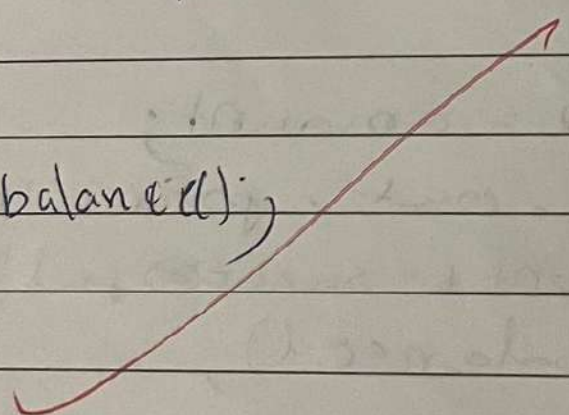
```
class curAcct extends Account
{
private double min balance = 1000
private double service charge = 50;

public curAcct (String cust name,
long acc no, doubk balance)

{
   super (cust name, acc number, "Current

}
 @ override
  Public void withdraw (doubk amount)
  {
   if (amount <= balance - minimom balance)
{
    balance -= amount;
 System.out.println(" withdrawl of " + amount);
}

display balance();
}
}
```

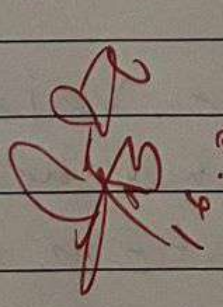


```
class DisplayThread extends Thread{
  private String message;

  Public DisplayThread(String message
  int interval)
{
  this.message = message;
  this.interval = interval;
}

  Public void run() {
    while(true) {
      try {
        System.out.println(message);
        Thread.sleep(interval);
      } catch(InterruptedException e){
          e.printStackTrace();
      }
    }
  }
}

  Public class Main{
  Public static void main(String[]
{
    DisplayThread thread1 = new
    DisplayThread("BMS", 10000)
    DisplayThread thread2 = new Display
    Thread("CSE", 2000);
    thread1.start()
    thread2.start()
```



Q Develop a Java program to create a class Bank that maintains two kinds of accounts for its customers, one called savings account and the other called current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed. Create a class account that stores customer name, account number and type of account. From this derive the class cur-acct and sav-acct to make them more specific to their requirements. Include the following methods in order to achieve the following tasks: Accept deposit from customer and update the balance. Display the balance. Compute the deposit interest. Permit the withdrawal and update balance

PROGRAM

```
import java.util.Scanner;

class Account
{
    String customerName;
    int accountNumber;
    String accountType;
    double balance;

    Account (String Name, int accNo,
      String Acctype, double initialBalance)
    {
        customerName = name;
        accountNumber = accNo;
        accountType = accType;
        balance = initialBalance;
    }

    void deposit (double amount)
    {
        balance += amount;
        System.out.println ("Deposit
        of $" + amount + " successful.");
    }
```

```
    void displayBalance()
    {
        System.out.println("Balance:
                                ₹ " + balance);
    }
}

class CurAcct extends Account
{
    double minBalance;
    double serviceCharge;

    CurAcct(String Name, int accno,
     string AccType, double initial
     Balance, double minBal,
     double charge)
    {
        super(name, accNo, accType,
                    InitialBalance);
        minBalance = minBal;
        serviceCharge = charge;
    }

    void withdraw(double amount)
```

```
{
    if (balance - amount >=
                    minBalance)
    {
        balance -= amount;
        System.out.println
        ("withdrawal of $"+
          amount + " successful");
    }
    else
    {
        System.out.println(
        "Insufficient funds.
          withdrawal failed.");
    }
}

void deductServiceCharge()
{
    if (balance < minBalance)
    {
        balance -= serviceCharge;
        System.out.println("service
        charge of $" + serviceCharge +
        "applied due to balance below min");
```

```
        }
    }
}

class SavAcct extends Acccount
{
    double interestRate;

    savAcct (String Name, intaccNo,
      String accType, double initial
      Balance, double interest)
    {
        super (name, accNo, accType,
            initialBalance);
        interestRate = interest;
    }

    void calculateInterest()
    {
        double interest = balance *
                    interestRate /1000;
        balance += interest;
        system.out.println ("Interest
          of $" + interest + " added");
    }
```

```
    void withdraw (double amount)
    {
        if (balance - amount >=0)
        {
            balance - = amount;
            system.out.println("
             withdrawal of $" +
              amount + " successful");
        }
        else
        {
            System.out.println("
             Insufficient funds.
             withdrawal failed");
        }
    }
}

class Bank
{
  public static void main (String [] args)
  {
    Scanner scanner = new Scanner (system
```

```
SavAcct savings = new SavAcct ("John
Doe", 123456, "savings", 1000, 5);

CurAcct current = new CurAcct ("Jane
Doe", 654321, "current", 2000, 500, 10);

System.out.println ("welcome to our
Bank!");

while (true)
{
    System.out.println ("\n1. Deposit \n2.
    withdraw \n3. Display Balance
    \n4. Exit");
    System.out.println ("Enter your
    choice");
    int choice = scanner.nextInt();

    switch (choice)
    {
      case 1:
        system.out.println ("Enter
        amount to deposit: ");
        double depositAmount = scanner
              .nextDouble ();
```

```
System.out.println("Select acco
(1 for savings), (2 for current
int accountChoice = scanner.
    nextInt();
if (accountChoice == 1)
  savings.deposit(depositAmount
else if (accountChoice==2)
  current.deposit(depositAmount
break;

case 2:
System.out.println("Enter amou
  to withdraw:");
double withdrawAmount =
  scanner.nextDouble();
System.out.println("select
  account (1 for savings,
    2 for current): ");
accountChoice = scanner.nextInt
if (accountChoice == 1)
  savings.withdraw(withdraw
                    Amount);
else if (accountChoice == 2)
{
```

```
            current.withdraw(withdraw
                Amount);
            current.deductServiceCharge();
        }
        break;
    case 3:
        System.out.println("Select
            account (1 for savings,
            2 for current):");
        accountChoice = scanner.nextInt();
        if (accountChoice == 1)
            savings.displayBalance();
        else if (accountChoice == 2)
            current.displayBalance();
        break;

    case 4:
        System.out.println("Thank you
            for banking with us!");
        System.exit(0);

    default:
        System.out.println("Invalid
            choice. Please try again");
```

```
            }
        }
    }
}
```

OUTPUT

```
Welcome to our bank!
1. Deposit
2. Withdraw
3. Display Balance
4. Exit
Enter your choice: 1
Enter amount to deposit: 1500
Select account (1 for savings, 2
  for current): 1
Deposit of $1500.0 successful.

1. Deposit
2. Withdraw
3. Display Balance
4. Exit
Enter your choice: 1
```

Enter amount to deposit: 2000
select Account (1 for savings, 2 for current): 2
Deposit of $2000.0 successful.

Enter your choice: 2
Enter amount to withdraw: 500
select Account (1 for savings, 2 for current): 1
withdrawal of $500.0 successful.

Enter your choice: 2
Enter amount to withdraw: 1500
select Account (1 for savings, 2 for current): 2
withdrawal of $1500.0 successful

Enter your choice: 3
select Account (1 for savings, 2 for current): 1
Balance: $2000.0

Enter your choice: 3
select Account (1 for savings, 2 for

current): 2
Balance: $2500.0

Enter your choice: 4
Thank you for banking with us

23-2-24

Program ①

```
import java.awt.*;
import java.awt.event
Public class EventHandling extends
Window Adapter implements ActionLis
Frame F;
textfield Tf;
EventHandling() {
F=new Frame();
F.addWindowListner(this);
tf=new textfield();
tf.setbounds(100,120,80,30);
b.addActionlistner(this);
F.add(b);
F.add(tf);
F.setSize(300,300);
F.setlayout(null);
F.setVisible(true)
}
Public void actionperformed(Action
{
    tf.settext("welcome");
}
Public void windowClosing
(WindowEvent e) {
System.exit(0)}
}
```

```
Public static void main (String args[])
{
    new Event handling ();
}
}
```