

1) Write a program to overload the method print that prints sum of n natural numbers when one variable is passed, and prints the prime 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  
        numbers is " + sum);  
    }  
    void print(int m, int n) {  
        System.out.println("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);  
        }  
    }  
}  
  
class Overload Demo {  
    public static void main (String[] arg) {  
        Overload o = new Overload();  
    }  
}
```

PAGE NO. 5

```

        O.print(s);
        O.print(7,13);
    }
}

```

OUTPUT : Sum of 5 natural numbers is 15
 prime numbers in the range are
 7
 11
 13

2. Write a Java program to create a class Grocery that has the variable C-name and C-phone. Create a method to accept 3 parameters to specify quantity of dal, quantity of pulses and quantity of sugar. The method to return the total price. Display the name, phone and total bill of 3 customers.

```

-> 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,
              double q-sugar) {
        total = q-dal * 100 + q-pulses * 80 + q-sugar * 50;
    }

    void display ()
    {

```

```

system.out.println("Name " + " " + "phone number " +
    " " + "Total");
system.out.println(c-name + " " + c-ph + " " + total);
system.out.println()
}
}

class GroDemo {
    public static void main (String[] args) {
        Grocery g1 = new Grocery ("Rama", "8060302010");
        Grocery g2 = new Grocery ("Sharma", "7689632510");
        Grocery g3 = new Grocery ("Bhama", "9632587412");
        g1.calc (2, 2, 1);
        g1.display ();
        g2.calc (3, 5, 2);
        g2.display ();
        g3.calc (1, 1, 0.5);
        g3.display ();
    }
}

```

OUTPUT :

Name	phone number	Total
Rama	8060302010	410.0
Sharma	7689632510	800.0
Bhama	9632587412	205.0

3. Write a Java program to calculate roots of a quadratic equation. Use appropriate methods to take input and calculate the roots.

```
→ import java.util.Scanner;
class Quad {
    int a, b, c;
    double root1, root2, d;
    Scanner s = new Scanner(System.in);

    void input() {
        System.out.println("Quadratic equation is  
in the form :  $ax^2 + bx + c$ ");
        System.out.print("Enter a:");
        a = s.nextInt();
        System.out.print("Enter b:");
        b = s.nextInt();
        System.out.print("Enter c:");
        c = s.nextInt();
    }

    void discriminant() {
        d = (b * b) - (4 * a * c);
    }

    void calculateRoots() {
        if (d > 0)
        {
            System.out.println("Roots are real & unequal");
            root1 = (-b + Math.sqrt(d)) / (2 * a);
            root2 = (-b - Math.sqrt(d)) / (2 * a);
            System.out.println("First root is " + root1);
            System.out.println("Second root is " + root2);
        }
        else if (d == 0)
        {
            // Roots are real & equal
        }
    }
}
```

Handwritten notes:
Roots are real & equal
m-12 m

```

system.out.println("Roots are real &
    equal");
    root1 = (-b + Math.sqrt(d)) / (2*a);
    system.out.println("Root : " + root1);
}
else
{
    system.out.println("No real solution. Roots
        are imaginary");
    double real = -b / (2*a);
    double real imaginary = Math.sqrt(-d) / (2*a);
    system.out.println("The equation has two
        complex roots : " + real + " + " + imaginary + "
        i and " + real + " - " + imaginary + " i");
}
}
}

```

class Main {

public static void main (String [] args) {

Quad q = new Quad();

q.input();

q.discriminant();

q.calculateRoots();

}
 }

Lab-2

Page No. _____
Date 12/11/24

3.

Q1

Create a class book that contains four member: name, author, price & num-pages. Include a constructor to set the values for the members, Include methods to set and get the details of the objects. Include a toString() method that could display the complete detail of the book. Develop a Java program to create a book object.

```
→ import java.util.Scanner;

class book
{
    String name;
    String author;
    int price;
    int numPage;

    Book() {}

    Book(String name, String author, int price,
        int numPages)
    {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
    }

    public String toString()
    {
        String name, author, price, numPages;
        name = "Book name" + this.name + "\n";
        author = "Author name" + this.author + "\n";
    }
}
```

```

        price = "Price" + this.price + "\n";
        numPage = "number of pages" + this.numPage + "\n";

        return name + author + price + numPage;
    }
}

```

class Main

```

{
    public static void main(String args[])
    {
        Scanner s = new Scanner(System.in);
        int n;
        String name;
        String author;
        int price;
        int numPage;
        System.out.println("Enter the number of books");
        n = s.nextInt();
        Books b[]
        b = new Books[n];
        for (int i = 0; i < n; i++)
        {
            System.out.println("Book" + (i+1) + ":");
            System.out.println("Enter name of book:");
            name = s.next();
            System.out.println("Enter author:");
            author = s.next();
            System.out.println("Enter price:");
            price = s.nextInt();
            System.out.println("Enter no. of pages:");
            numPage = s.nextInt();
        }
    }
}

```



```

b[i] = new books (name, author, price, number of pages);
}
for (int i = 0; i < n; i++)
    System.out.println("Book " + (i+1) + " : " +
        b[i]);
}
}

```

OUTPUT :

Enter the number of book : 2

Book 1 :

Enter the name of the book : Jungle-Book

Enter the author of the book : Rudyard-Kipling

Enter the number of pages of the book : 700

Enter the price of the book : 1200

Book 2 :

Enter the name of the book : Frankenstein

Enter the ^{author} name of the book : Mary Shelley

Enter the price of the book : 723

Enter the number of pages of the book : 1500

Book 1 :

Book name : Jungle-Book

author : Rudyard-Kipling

Price : 1200

number of pages : 700

Book 2 : ~~Frankenstein~~

Book name : ~~Mary Shelley~~ Frankenstein

author : Mary Shelley

price : 723

number of pages : 1500

Q3) Write a Java program to create a class student with members USN, name, marks (6 subjects). Include method to accept student details and marks, Also include a method to calculate the percentage and display appropriate details.
(Array of student object to be created).

-> Import java.util.Scanner

```
class student {
```

```
    String USN;
```

```
    String name;
```

```
    int [] marks = new int [6];
```

```
    void acceptDetails () {
```

```
        Scanner scanner = new Scanner (System.in);
```

```
        System.out.print ("Enter USN");
```

```
        USN = scanner.next();
```

```
        System.out.print ("Enter Name");
```

```
        name = scanner.next();
```

```
        System.out.println ("Enter marks for 6 subject");
```

```
        for (int i=0; i<6; i++) {
```

```
            System.out.print ("Subject " + (i+1) + ": ");
```

```
            marks[i] = scanner.nextInt();
```

```
        }
```

```
    }
```

```
    double calculatePercentage () {
```

```
        int totalMarks = 0;
```

```
        for (int mark : marks) {
```

```
            totalMarks += mark;
```

~~for (int marks = 0;~~
return (double) total marks / 6 ;

```
}  
void displayDetails() {  
    System.out.println("Student details");  
    System.out.println("USN" + USN);  
    System.out.println("Name" + name);  
    System.out.println("Percentage" +  
        calculatePercentage() + "%");  
}
```

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

```
        System.out.print("Enter the number of student  
        int numStudents = scanner.nextInt();
```

```
        Student[] student = new Student[numStudents];  
        for (int i = 0; i < numStudents; i++) {  
            student[i] = new Students();  
            System.out.println("\nEnter detail  
            for student" + (i+1) + " :");  
            student[i].acceptDetails();  
        }
```

```
        System.out.println("\nDetails of all student  
        for (Student student : students) {  
            student.displayDetails();  
            System.out.println();  
        }
```

```
    }  
}
```

12.01.21

- 1) Develop a Java Program to create an abstract class named `Shape` that contains two integers and an empty method named `printArea()`. Provide three classes named `Rectangle`, `Triangle` and `Circle` such that each one of the classes extends the class `Shape`. Each one of the classes contain the method `printArea()` that prints the area of the given shape.

```
→ abstract class Shape {  
    protected int dimension1;  
    protected int dimension2;
```

```
public Shape (int dimension1, int dimension2)  
{  
    this.dimension1 = dimension1;  
    this.dimension2 = dimension2;  
}
```

```
    public abstract void printArea();  
}
```

```
class Rectangle extends Shape {  
    public Rectangle (int length, int width) {  
        super (length, width);  
    }  
}
```

② override

```
    public void printArea() {  
        int area = dimension1 * dimension2;  
        System.out.println ("Rectangle Area : " + area);  
    }  
}
```

```
class Triangle extends Shape {  
    public Triangle (int base, int height) {  
        super (base, height);  
    }  
}
```

@ override

```
public void printArea() {
    double area = 0.5 * dimension1 * dimension2;
    System.out.println("Triangle area: " + area);
}
```

```
class Circle extends Shape {
    public Circle(int radius) {
        super(radius, 0);
    }
}
```

@ override

```
public void printArea() {
    double area = Math.PI * dimension1 * dimension1;
    System.out.println("Circle Area: " + area);
}
```

```
public class Main {
    public static void main (String[] args) {
        Rectangle rectangle = new Rectangle(5, 10);
        Triangle triangle = new Triangle(8, 6);
        Circle circle = new Circle(4);
```

```
        rectangle.printArea();
        triangle.printArea();
        circle.printArea();
    }
}
```

OUTPUT :

Rectangle Area : 50

Triangle Area : 24

Circle Area : 50.24

23 Develop a Java program to create a class Bank that maintain two kinds of account of its customers, one called savings account and another current account. The saving account provides compound interest and with drawal facilities but no cheque book facility. The current account provides 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 classes cur-acc and sav-acc to make them more specific to their requirements. Include the necessary method in order to achieve the following tasks:

- Accepts deposits from customer and update the balance.
- Display the balance
- Compute and deposits interest
- Permit withdrawal and update the balance.

Check for the minimum balance, impose penalty if necessary and update the balance

→ - import java.util.Scanner
class Account {

protected String customerName;
protected long accountNumber;
protected String accountType;
protected double balance;

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

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

}

public void displayBalance() {

System.out.println("Account Number : " + accountNumber);
System.out.println("Customer Name : " + customerName);
System.out.println("Account Type : " + accountType);
System.out.println("Balance : " + balance);
}

public void deposit (double amount) {

balance += amount;
System.out.println("Deposit of ₹ " + amount + "
Successful.");
displayBalance();
}

public void withdraw (double amount) {

if (amount <= balance) {
balance -= amount;
System.out.println("Withdrawal of ₹ " +
amount + " Successful");
} else {
System.out.println("Insufficient fund");
}

display Balance();

}

}

class CurrAcct extends Account {

private double minimumBalance = 1000;

private double service charge = 50;

public CurrAcct (String CustomerName, long
accountNumber, double balance) {

super (CustomerName, accountNumber, "current",
balance);

}

@ override

public void withdraw (double amount) {

if (amount <= balance - minimumBalance) {

balance -= amount;

System.out.println ("Withdrawal of \$" + amount +
" successful");

} else {

System.out.println ("Insufficient funds.
Service charge of \$" + service charge + " imposed");

balance -= service charge;

}

}

display balance();

}

}

class SavAcct extends Account {

private double interestRate = 0.05;

public void computeInterest () {

double interest = balance * interestRate;

balance += interest;

3. system.out.println("Interest of \$" + interest);
displayBalance();
}
}

→ public class Two {

public class Static void main(String[] args)
Scanner scanner = new Scanner(System.in);
SavingsAccount savAcct = new SavingsAccount
("John Doe", 123456789, 5000);
SavingsAccount.displayBalance();
SavingsAccount.deposit(1000);
SavingsAccount.calculateInterest();
SavingsAccount.withdraw(2000);

CurrentAccount currentAcct = new CurrentAccount("Jane
Doe", 987654321, 1500);
CurrentAccount.displayBalance();
CurrentAccount.deposit(5000);
CurrentAccount.withdraw(2000);

Scanner.close();

}

OUTPUT: Customer Name: John Doe
Account Type: Savings
Balance: \$6300.0
Withdrawal of \$2000 is:
Account Number: 123456789
Account Type: Savings
Balance: \$4300.0
Account Number: 987654321
Customer Name: Jane Doe.

Account type : current

Balance \$1500.00.

~~\$1500.00~~
19.01.21

* Create a package CIE which has two classes - student and internal. The student class has members like USN, name, sem. The class internal derived from student has an array that stores the internal marks scored in five courses of the current semester of the student. Create another package SEE which has the class external which is a derived class of student. This class has an array that stores the SEE marks scored in five courses of the current semester of the student. Import the two packages in a file that declares the final marks of a student in all five courses.

```
→ Package CIE;
import java.util.Scanner;
public class student {
```

```
    protected String usn = new String();
    protected String name = new String();
    protected int sem;
```

```
    public void inputStudentDetails() {
        Scanner S = new Scanner(System.in);
        System.out.println("give USN");
        usn = S.nextLine();
        System.out.println("give name");
        name = S.nextLine();
    }
}
```



```
system.out.println("give sem");  
sem = s.nextInt();  
}
```

```
public void displayStudentDetails () {
```

```
    system.out.println("The user is : ", user);  
    system.out.println("The name is : ", name);  
    system.out.println("The sem is : ", sem);  
}
```

```
Package CIE;
```

```
import java.util.Scanner;
```

```
protected int marks[] = new int[5];  
public void input (CIE marks []) {  
    Scanner s = new Scanner(System.in);  
    for (int i = 0; i < 5; i++) {  
        System.out.println("Enter marks for  
        course " + (i+1) + " : ");  
        marks[i] = s.nextInt();  
    }  
}
```

```
Package SEF;
```

```
import CIE.Internals;
```

```
import java.util.Scanner;
```

```
public class SEF extends  
    Internals {
```

```
    protected int marks[];
```

```
    protected int finalmarks[];
```

```
}
```

```
public SEF () {
```

marks = new int[5];
final marks = new int[5];

```
}  
public void input SEE marks() {  
    Scanner s = new Scanner(System.in);  
    for (int i = 0; i < 5; i++) {  
        System.out.println("Subject " + (i + 1) + " marks");  
        marks[i] = s.nextInt();  
    }  
}
```

```
}  
public void calc.Finalmarks() {  
    for (i = 0; i < 5; i++)  
        final marks[i] =  $\frac{\text{marks}[i] + \text{super marks}}{2}$   
}
```

```
package SEE;  
public class External extends Internal {  
    public External();  
    public class main {  
        public static void main (String arg[])
```

```
    {  
        for (i = 0; i < num of student; i++) {  
            final marks[i] = new External();  
            final marks[i] = new External();  
            final marks[i] = input CIE marks();  
        }  
    }
```

```
    System.out.println("display data")  
    for (int i = 0; i < num of students; i++) {  
        final marks[i] = calc.Finalmarks();  
    }  
}
```


Q1 Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called "Father" and derived class called "son" which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge(), when the input age ≤ 0 . In son class, implement a constructor that takes both father & son's age and throws an exception if son's age is $> \text{father's age}$.

```
class Father {
    private int age;
    public Father (int age) throws WrongAge
        Exception {
        if (age < 0) {
            throw new WrongAge Exception("Age can't be
            negative");
        }
        this.age = age;
    }
    public int getAge() {
        return age;
    }
}

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

class son extends Father {
    private int son Age;
```

```

public son (int fatherAge, int sonAge) throws
WrongAgeException, SonAgeException {
    super (fatherAge);
    if (sonAge >= fatherAge) {
        throw new SonAgeException ("Son
        can't be greater father's age");
    }
}

```

```

    this.sonAge = sonAge;
}

public int getsonAge() {
    return sonAge;
}

```

```

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

```

```

public class Main {
    public static void main (String[] args) {
        try {
            Father father = new Father (45);
            System.out.println ("Father's age: " +
            father.getAge());
        }
    }
}

```

```

        Son son = new son (45, 20);
        System.out.println ("son's age" + son
        getsonAge());
    }
}

```

```

        SonAgeException e;
        catch (WrongAgeException e) {
            System.out.println ("Error" + e.getMessage());
        }
    }
}

```


Q1 Write a program which creates two threads one thread displaying "BMS College of Engineering" once every ten seconds & another displaying "CSE" one every two seconds.

→ class DisplayThread extends Thread {
 private String message;
 private int interval;

 public DisplayThread (String message, int interval) {
 this.message = message;
 this.interval = interval;

 }
 @Override

 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[] args) {
 DisplayThread thread1 = new DisplayThread ("BMS college of engg", 10000);
 DisplayThread thread2 = new DisplayThread ("CSE", 2000);
 thread1.start ();

Thread 2, start();

}
}

Output: CSE
BMS collage of Engg.
CSE
CSE
BMS collage of Engg.
CSE
CSE
CSE
BMS collage of Engg.
CSE

~~CSE~~
~~BMS~~
12.02.2020

1. Creating label, Button & TextField in a Frame using AWT.

```
import java.awt.*;  
import java.awt.event.*;
```

```
public class AWTExample extends  
WindowAdapter {  
    Frame f;  
    AWTExample() {
```

```
        f = new Frame();  
        f = addWindowListener(this);  
        label l = new Label("Employee id:");  
        button b = new Button("submit");  
        TextField t = new TextField();
```

```
        l.setBounds(20, 80, 80, 30);  
        t.setBounds(20, 100, 80, 30);  
        b.setBounds(100, 100, 80, 30);
```

```
        f.add(b);  
        f.add(t);  
        f.add(l);  
        f.setSize(400, 300);  
        f.setTitle("Employee info");  
        f.setLayout(null);  
        f.setVisible(true);
```

```
    }  
    public void windowClosing(WindowEvent e) {  
        System.exit(0);  
    }
```

3

```

public static void main (String[] args) {
    AWTExample aut obj = new AWTExample();
}
}

```

2. Create a button and add a action listener for Mouse click.

```

-> import java.awt.*;
import java.awt.event.*;
public class EventHandling extends
WindowAdapter implements ActionListener {
    Frame f;
    TextField tf;
    Event Handling () {

```

```

        f = new Frame ();
        f.addWindowListener (this);
        tf = new TextField ();
        tf.setBounds (60, 50, 170, 20);
        Button b = new Button ("Click me");
        b.setBounds (100, 120, 80, 30);

```

```

        b.addActionListener (this);
        f.add(b); f.add(tf);
        f.setSize (300, 300);
        f.setLayout (null);
        f.setVisible (true);

```

```

    }

```

```

    public void actionPerformed (ActionEvent e) {
        tf.setText ("Welcome");
    }
}

```



```
public void windowClosing(WindowEvent e) {
    System.exit(0);
}
```

```
public static void main(String args[]) {
    new EventHandling();
}
}
```

1. Example 1.

```
import java.io.*;

public class ByteArrayInput {
    public static void main(String[] args)
        throws IOException {
        byte[] buf = {35, 36, 37, 38};
        ByteArrayInputStream byt = new ByteArrayInputStream(
            buf);
        int k = 0;
        while (k = byt.read() != -1) {
            char ch = (char) k;
            System.out.println("ASCII value of
            character is " + k + "; special character is " + ch);
        }
    }
}
```

2. Example 2.

```
public class FileEx {
    public static void main(String a[])
        throws IOException {
        FileInputStream fin = new FileInputStream(
            "Example.txt");
    }
}
```

3

```
int content;
system.out.println("Remaining bytes that  
be read : " + fin.available());
```

```
content = fin.read();
system.out.print((char) content + " ");
system.out.print(content + " ");
system.out.print("Remaining bytes that  
be read : " + fin.available());
```

```
system.out.println("Remaining bytes that  
can be read : " + fin.available());
}
}
```

Example 3.

```
import java.io.*;
public class ByteArray - eu {
    public static void main (String arg[])
        throws exception {
        FileOutputStream fout1 = new FileOutputStream
        ("Example - text");
        FileOutputStream fout2 = new FileOutputStream
        ("Example - text");
```

```
ByteArrayOutputStream bout = new ByteArrayOutputStream
        output stream ();
        bout.write(65);
        bout.writeTo(fout1);
        bout.writeTo(fout2);
        bout.flush();
        bout.close();
```



```

    system.out.println("Success...");
}
}

```

Example 4

→ import java.io.FileInputStream;
import java.io.IOException;

```

public class FileEx4 {
    public static void main (String a[]
        throws IOException {
        FileInputStream fin = new FileInputStream
        ("Example.txt");
        byte [] bytes = new byte [20];
        int i;
        char c;
        i = fin.read (bytes);
        System.out.println ("Number of byte read: " + i);
        System.out.println ("Byte read: ");

```

```

        for (byte b: bytes) {

```

```

            c = (char) b;

```

```

            System.out.print (c);

```

```

        }
    }
}

```

```

}

```

```

}

```

13) basis of class 35 : Special class 25
basis of class 36 : Special class 1
basis of class 37 : Special class 1
basis of class 78 : Special class 2

~~Q. 13~~
~~3.12.21~~