Lab-1

Date
Page

Sample programs

1)
```
class Hello {
    public static void main (string i[])
    {
    system.out.println ("Hello world");
    }
}
```

Output
Hello world.

2) Sum, difference, product & quotient

```
class Calc
{
    public static void main (string i[])
    {
        int a = 2; int b = 4;
        system.out.println ("Sum" + (a+b));
        system.out.println ("Difference" + (b-a));
        system.out.println ("Product" + (a*b));
        system.out.println ("Quotient" + (b/a));
    }
}
```

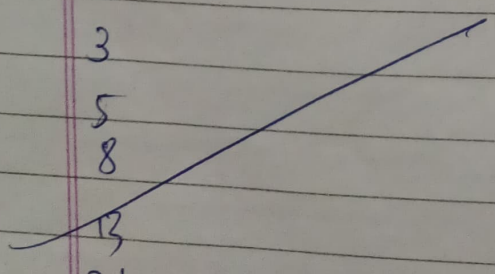
Output
Sum 6
Difference 2
Product 8
Quotient 2

3) Fibonacci series

```
class fib
{
    public static void main (string i[])
    {
        int a=0, b=1, i=0, c;
        system.out.println ("a" + "\n" + b);
        while (i<7)
        {
            system.out.println (a+b;
            c=a+b;
            a=b;
            b=c;
            i++;
        }
    }
}
```

Output

0
1
1
2
3
5
8
13
21



4)

4) Prime number

```
class Number
{
    public static void main (string i[])
    {
        int a = 9, b = 11, i = 0;
        for (int i = 2 ; i < a ; i++)
        {
            if (a % c == 0)
                c++;
        }
        if (c > 0)
            system.out.println ("9 is not prime");
        else
            system.out.println ("9 is prime");
        c = 0;
        for (int i = 2; i < b; i++)
        {
            if (b % i == 0)
                c++;
        }
        if (c > 0)
            system.out.println ("11 is not prime");
        else
            system.out.println ("11 is prime");
    }
}
```

Output

9 is not a prime number.
11 is a prime number.

1) Create a class Book that contains 4 members name, author, price, & num_pages. Include a constructor to set the values for the members. Include methods to set & get the details of the objects. Include a toString() method that could display the complete details of the book. Develop a java program to create n book objects.

→
```
import java.util.*;
class Books {
    String name;
    String author;
    int price;
    int numpages;
  Books() {}
  Books(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;
    int price, numpages;
    name = "Book name: " + this.name + "\n";
    author = "Author: " + this.author + "\n";
    price = "Price: " + this.price + "\n";
```

```
numpages = "No. of pages :" + this.numpages
                                + "\n";
return name + author + price + numpages;
    }
}
class Main
{
    public static void main (String args[])
    {
        Scanner s = new Scanner (System.in);
        int n;
        String name;
        String author;
        int price, numpages;
        System.out.println ("Enter no. 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 peice:");
            price = s.nextInt();
            System.out.println ("Enter no. of pages:");
            numpages = s.nextInt();
            b[i] = new Books (name, author, price,
                              numpages);
        }
```

```
for (int i=0; i<n; i++)
    System.out.println("Book " + (i+1) + "\n" + b[i]);
}
}
```

Output

Enter the number of books: 1
Book 1:
Enter name of book: Jungle-book
Enter author: Ruyard-Kipling
Enter price: 900
Enter number of pages: 500

Book 1:
Book name: Jungle-book
Author: Ruyard Kipling
Price: 900
Number of pages: 500

2) Write a java program to create a class Student with members USN, name, marks (6 subjects). Include methods 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.*;
class Student {
    private String usn;
    private String name;
    private int [] marks;
```

```
public Student (String usn, String name)
{
    this.usn = usn;
    this.name = name;
    this.marks = new int [6];
}
public void acceptDetails()
{
    Scanner scanner = new Scanner (System.in);
    System.out.println ("Enter USN: ");
    this.usn = scanner.nextLine();
    System.out.println ("Enter name: ");
    this.name = scanner.nextLine();

    for (int i=0; i<marks.length; i++){
        System.out.println ("Enter marks for
            Subject " + (i+1) + ": ");
        this.marks (i) = scanner.nextInt();
    }
}

public double calculatePercentage() {
    int totalMarks = 0;
    for (int mark : marks) {
        totalMarks += mark;
    }
    return (double) totalMarks / marks.length;
}

public void displayDetails () {
    System.out.println ("USN: " + this.usn);
    System.out.println ("Name: " + this.name);
```

```
        System.out.println("Marks:");
        for (int i=0; i< marks.length ; i++)
        {
            System.out.println("Subject "+(i+1)+
                    ":" + marks[i] + " ");
        }
        System.out.println();
        System.out.println("Percentage :"+
                calculatePercentage() + "%.");
    }
}

public class StudentArrayDemo{
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter number of
                students :");
        int numStudents = sc.nextInt();
        Student [] students = new Student
                [numStudents];
        for (int i=0; i< numStudents ; i++)
        {
            System.out.println("Enter details
                for student" + (i+1) + ":");
            students[i] = new Student("", "");
            students[i].acceptDetails();
        }

        System.out.println("Details of students");
        for (Student student : students) {
```

```
            student.displayDetails();
            System.out.println("\n.....");
            }
        }
    }
```

Output

Enter USN : 11
Enter number of students : 2
Name : ABC
Subject 1 : 90
Subject 2 : 90
Subject 3 : 90
Subject 4 : 90
Subject 5 : 90
Subject 6 : 90

Details of students
USN : 11
Name : ABC
Subject 1 : 90
Subject 2 : 90
Subject 3 : 90
Subject 4 : 90
Subject 5 : 90
Subject 6 : 90
Percentage : 90 %.

3) Quadratic equation:

```
import java.util.Scanner;
import java.lang.Math;
class quadratic
{
    public static void main (String xx[])
    {
        int a,b,c;
        System.out.println("Enter values of a,b,c
                            respectively");
        Scanner sl = new Scanner(System.in);
        a = sl.nextInt();
        b = sl.nextInt();
        c = sl.nextInt();
        double d = b*b - 4*a*c;
        Equation eq = new Equation(a,b,c);
        eq.quad();
}

class Equation
{
    int a,b,c;
    Equation(int x, int y, int z);
    {
        a=x;
        b=y;
        c=z;
    }
    quad()
    {
        if (a==0)
        System.out.println("Not a quadratic
                            equation");
```

```
else if (d>0)
{
    System.out.println("The equation has 2
                    different and real solutions");
    double r1 = (-b + Math.sqrt(d))/(2*a);
    double r2 = (-b + Math.sqrt(d))/(2*a);
    System.out.println("r1 = " +r1);
    System.out.println("r2 = " +r2);
}
else if (d==0)
{
    System.out.println("Equation has real
                    & equal solutions");
    double r1 = -b/(2*a);
    double r2 = -b/(2*a);
    System.out.println(r1);
    System.out.println(r2);
}
else(d<0)
{
    System.out.println("Nonreal solutions");
}
}
}
```

Output

Enter values of a, b, c respectively
1 5 6

8/2/2024

22/1/24

1) Develop a Java pgm to create an abstract class name named Shape that contains two integers & an empty method named printArea(). Provide three classes named Rectangle, Triangle & 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 given shape.

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

```
class Triangle extends Shape {
    public Triangle (int base, int height) {
        super (base, height);
    }
    public void printArea() {
        double area = 0.5 * dimension1 * dim2;
        System.out.println ("Area of Triangle:"
                            + area);
    }
}

class Circle extends Shape {
    public Circle (int radius) {
        super (radius, 0);
    }
    public void printArea() {
        double area = MATH.PI * dim1 * dim1;
        System.out.println ("Area of Circle:" +
                            area);
    }
}
public class Main {
    public static void main (String [] args) {
    Rectangle rectangle = new Rectangle (4,5);
    rectangle.printArea();

    Triangle triangle = new Triangle (3,6);
    triangle.printArea();

    Circle circle = new Circle (7);
    circle.printArea();
    }
}
```

Output

Area of rectangle : 20
Area of triangle : 9
Area of circle : 153.93804002589985

2) WAP to create a class Bank that maintains 2 kinds of account for its customers, one sav acc and another cur acc. Sav acc provides compound interest & withdrawal facilities but no cheque book facility. The cur acc provides cheque book but no interest. Cur acc holders should also maintain a min balance & if the balance falls below this level, a service charge is imposed.

Create a class Account that stores customer name, acc no. & type of acc. From this derive the classes Cur-acct & Sav-acct to make them more specific to their requirements. Include necessary methods in order to achieve following tasks.

i) Accept deposit from customer & update the balance.
ii) Display balance
iii) Compute & deposit interest.
iv) Permit withdrawal & update the balance. Check for min balance, impose penalty if necessary & update the balance.

→

```
class Bank {
    public static void main(String[] args)
    {
    SavAcc savacc = new SavAcc("John", "SA1001");
    CurAcc curacc = new CurAcc("Jane", "CA2002");
    savacc.deposit(5000);
    savacc.displayBalance();
    savacc.computeInterest();
    savacc.withdraw(2000);
    savacc.displayBalance();

    curacc.deposit(8000);
    curacc.displayBalance();
    curacc.withdraw(5000);
    curacc.displayBalance();
    }
}

class Account {
    protected String custName;
    protected String accNo;
    protected double balance;

    public Account(String custName, String
                                    accNo)
    {
        this.custName = custName;
        this.accNo = accNo;
        this.balance = 0;
    }
    public void deposit(double amount){
        balance += amount;
        System.out.println("Deposit of $" +
```

```
    public void displayBalance()
    {
        System.out.println("Acc no" + accNo +
                           "\nBalance: $" + balance);
    }
}

class Savings extends Account {
    public SavAcc(String custName, String
                                    accNo) {
        super(custName, accNo);
    }
    public void computeInterest()
    {
        double InterestRate = 0.05;
        double interest = balance * InterestRate;
        balance += interest;
        SOP("Interest of $" + interest + "computed
                & added to balance.");
    }
    public void withdraw(double amount)
    {
        if (balance >= amount) {
            balance -= amount;
            SOP("withdrawal of $" + amount +
                                "successful.");
        }
        else {
            SOP("Insufficient funds for withdrawal");
        }
    }
}
```

```
class CurAcc extends Account
{
    private double minBal = 1000;
    public CurAcc (String custName, String
                                    accNo)
    {
        super (custName, accNo)
    }
    public void withdraw (double amount)
    {
        if (balance - amount >= minBal)
        {
            balance -= amount;
            SOP("withdrawal of $" + amount
                    + "successfully");
            imposeServiceCharge();
        }
    }

private void imposeServiceCharge()
{
    double serviceCharg = 20;
    balance -= serviceCharg;
    SOP("Service charge of $" + serviceCharge
                    + "imposed");
}
}
```

880

22/1/24

```
class CurAcc extends Account
{
    private double minBal = 1000;
    public CurAcc (String custName, String
                                    accNo)
    {   super (custName, accNo)
    }
    public void withdraw (double amount)
    {
        if (balance - amount >= minBal()
            balance -= amount;
            SOP("withdrawal of $" + amount
                + successfully");
            imposeServiceCharge();
        }
}

private void imposeServiceCharge()
{
    double serviceCharg = 20;
    balance -= serviceCharg;
    SOP("Service charge of $" + serviceCharge
                + "imposed");
    }
}
```

22/11/24

29/1/24

1) Create a package CIE which has two classes - Student & Internals. The class Student has members like usn, name, sem. The class internals 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. 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 n students in all five courses

i) Create a folder CIE & save the programs Student.java & Internals.java within it.
ii) Create a folder SEE & save the program External.java within it.
iii) Save the Main program outside these 2 folders.
iv) ~~Compile~~ Compile Main.java & execute Main class.

→
```
package CIE;
public class Student {
    public String usn;
    public String name;
    protected int sem;
    public Student(String usn, String name,
                   int sem)
    {   this.usn = usn;
        this.name = name;
        this.sem = sem;
    }
}
```

```
package CIE;
public class Internals extends Student {
    public int[] internalMarks = new int[5];
    public Internals (String usn, String name,
                      int sem, int[] internalMarks)
{
        super(usn, name, sem);
        this.internalMarks = internalMarks;
    }
}
```

```
package SEE;
import CIE.Student;
public class External extends Student {
    public int[] seeMarks;
    public External (String usn, String name,
                     int sem, int[] seeMarks)
    {
        super(usn, name, sem);
        this.seeMarks = seeMarks;
    }
}
```

```
import CIE.Internals;
import SEE.Externals;
public class Main {
    public static void main (String[] args)
    {
        int[] internalMarks1 = {80, 75, 90, 85, 87};
        Internals st1 = new Internals ("1ABC123",
                "John Doe", 3, internalMarks1);
```

```
int [] seeMarks1 = {70, 85, 78, 92, 88};
External st2 = new External ("2X4456",
                "Jane Smith", 3, seeMarks1);

int [] finalMarks1 = new int [5];
for (int i=0 ; i<5 ; i++) {
    finalMarks1 [i] = st1.internalMarks [i]
                    + st2.seeMarks [i];
}
System.out.println ("Final marks for" +
        st1.name + "(USN:" + st1.usn + ")");
for (int i=0 ; i<5 ; i++) {
    System.out.println ("Course" + (i+1) +
            ":" + finalMarks1 [i]);
```

Output

"1BBC123", "John Doe", 3, [80, 75, 90, 85, 88]
"2X42456", "Jane Smith", 3, [70, 85, 78, 92, 88].

19/2/24

Date
Page

1) write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called "Father" & derived class called "Son" which extends the base class. In Father class, implement a constructor, which takes the age & throws exception (WrongAge) when the i/p age < 0. In Son class, implement a constructor that cases both father & son's age & throws an exception if son's age is >= father's age.

→
```
class WrongAgeException extends Exception {
    public WrongAgeException (String message) {
        super (message);
    }
}

class Father {
    int age;
    public Father (int age) throws WrongAgeException
    {
        if (age < 0) {
            throw new WrongAgeException ("Age
                cannot be negative");
        }
        this.age = age;
    }
}

class Son extends Father {
    int sonAge;
    public Son (int fatherAge, int sonAge)
        throws WrongAgeException {
        super (fatherAge);
```

19/2/24

Date
Page

1) Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called "Father" & derived class called "Son" which extends the base class. In Father class, implement a constructor, which takes the age & throws exception (WrongAge) when the i/p age < 0. In Son class, implement a constructor that cases both father & son's age & throws an exception if son's age is >= father's age.

→

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

class Father {
    int age;
    public Father (int age) throws WrongAgeException
    {
        if (age < 0) {
            throw new WrongAgeException ("Age
                cannot be negative");
        }
        this.age = age;
    }
}

class Son extends Father {
    int sonAge;
    public Son (int fatherAge, int sonAge)
        throws WrongAgeException {
        super (fatherAge);
```

```
        if (sonAge >= fatherAge) {
            throw new WrongAgeException("Son's
            age cannot be greater than father's");
        }
        this.sonAge = sonAge;
    }
}

public class Main {
    public static void main (String[] args)
    {
        try {
            int fatherAge = 40;
            int sonAge = 20;
            Son son = new Son (fatherAge, sonAge);
            SOP ("Father's age: " + fatherAge);
            SOP ("Son's age: " + son.sonAge);
        }
        catch (WrongAgeException e) {
            SOP ("Exception caught: " + e.getMessage());
        }
    }
}
```

Output

Father's age = 40

Son's age = 20

Date
Page

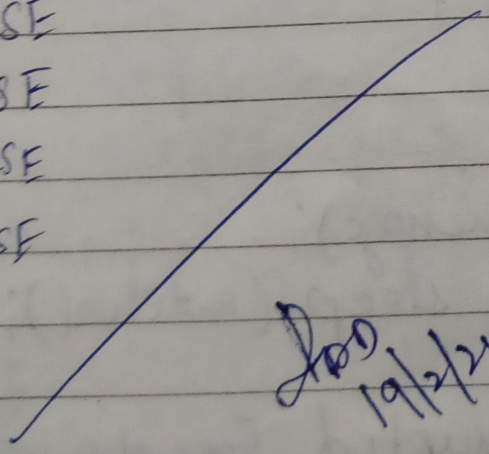
27 Write a program which creates two threads, one thread displaying "BMSCE" once every ten seconds & another displaying "CSE" once every two seconds.

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

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

    public void run() {
        while (true) {
            try {
                SOP (message);
                Thread.sleep (interval);
            }
            catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }

public class Main {
    public static void main (String [] args) {
        DisplayThread thread1 = new DisplayThread
            ("BMSCE", 10000);
```

```
DisplayThread = new DisplayThread ("CSE", 2000);

    thread1.start();
    thread2.start();
  }
}
```

Output

BMSEE
CSE
CSE
CSE
CSE
CSE
BMSCE
CSE
CSE
CSE
CSE
CSE



19/2/24

26/2/24

Date
Page

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

→
```
import java.awt.*;
import java.awt.event.*;
public class AWTExample extends WindowAd-
-apter {
    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(l);
        f.add(t);
        f.setSize(400,300);
        f.setTitle("Employee info");
        f.setLayout(null);
        f.setVisible(true);
    }
public void windowClosing(WindowEvent e){
        System.exit(0);
}
public static void main(String[] args){
    AWTExample awt_obj = new AWTExample();
}
}
```

2) Creating a button & 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;
    EventHandling () {
    {
        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 ();
        }
}
```

Date
Page

* Programs on IO

1)
```
import java.io.*;
public class ByteArrayInput {
    public static void main(String[] args) throws
                                        IOException {
        byte[] buf = {35, 36, 37, 38};
        ByteArrayInputStream byt = new ByteArray-
                                    -InputStream(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)
```
public class FileEx {
    public static void main(String a[]) throws
                                        IOException {
        FileInputStream fin = new FileInputStream
                                    ("Example.txt");
        int content;
        System.out.println("Remaining bytes
        that can be read:" + fin.available());
        content = fin.read();
        System.out.println((char(content) + " ");
        System.out.println(content + " ");
        SOP("Remaining bytes can be read:" +
                                    fin.available());
        SOP("Remaining bytes can be read:" +
                                    fin.available());
    }
}
```

4/2/24