a) 08/1/2024

Lab-1

1) Create a class Book that contains four members 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 details of book.

```
import java.util.Scanner;
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, price, numpages;
        name = "Book name: " + this.name + "\n";
        author = "Author name: " + this.author + "\n";
        Price = "Price: " + this.price + "\n";
        numpages = "No of pages: " + this.numpages + "\n";
        return name + author + price + numpages;
    }
}
class maintain
{
    public static void main(String args[])
    {
        Scanner sc = new Scanner(System.in);
        int n;
        String name;
        String author;
```

```
int price ;
int numpages;
System.out.println("Enter the no of book:");
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 the book:");
  name = s.next();
  System.out.println("Enter the Author name:");
  author = S.next();
  System.out.println("Enter Authorname:");
  author = s.next();
  System.out.println("Enter Price:");
  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 no of book:

1

Book1:

Enter name of the book:

OOPS

Enter author name:

Sumita Arora

Enter price:

300

Enter no of pages:

500

Book 1:

Book name: OOPS

Author name: Sumit Arora

Price: 300

No of pages: 500

05/01/2024

(a) Write a Java program to create a class student with 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.

```
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.println("Enter marks for 6 subjects:");
        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;
        }
        return (double) totalMarks / 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 Dhruv
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
```

```
System.out.print("Enter the number of students : ");
int numStudents = scanner.nextInt();
Student[] students = new Student[numStudents];
for(int i=0; i<numStudents; i++){
    students[i] = new Student();
System.out.println("\nEnter details for student "+(i+1)+" : ");
students[i].acceptDetails();
}
System.out.println("\nDetails of all students:");
for(Student student : students){
    student.displayDetails();
    System.out.println();
}
}
}
```

Output.

```
Enter the number of students : 1
Enter details for student 1:
Enter USN : 1BM22CS083
Enter name : Dhruv
Enter marks for 6 subjects:
Subject 1: 100
Subject 2: 100
Subject 3: 100
Subject 4: 100
Subject 5: 100
Subject 6: 100
Details of all students:
Student details:
USN : 1BM22CS083
Name : Dhruv
Percentage : 100.0%
```

SSD 8/1/2024

08/01/2024

* Quadratic equations roots

```
import java.lang.Math;
import java.util.Scanner;
class Quadratic2 {
    Scanner S = new Scanner (System.in)
    System.out.println ("Enter Equation of form ax^2+bx+c");
        this.a = S.nextInt();
        b = S.nextInt();
        c = S.nextInt();
}
Void CalculateRoot() {
    int d = b*b - 4*a*c
    if(d==0){
        double root1 = -b/(2*a)
        System.out.println("Roots are"+root1+"and"+root2);
    }
else if (d>0){
    double root1 = (-b + Math.sqrt(d))/(2*a);
    double root2 = (-b - Math.sqrt(d))/(2*a);
    System.out.println("Roots are "+root1+"and"+root2);
}
    else{
        double rp1 = -b/(2*a);
        double rp2 = Math.sqrt(d)/(2*a)
        System.out.println("Roots are "+rp1+" i "+rp2+"and"+rp1+
                "-i"+rp2);
    }
}
class Run{
    public static void main (String args[]){
        Quadratic2 q = new Quadratic2();
```

q. Calculate Root ( ) ;

}

}

Output

Enter constant in form ax+by+c 1 -7 10

Roots are 5.0 and 2.0

Enter constant in form ax+by+c 1 2 1

Roots are -1.0 and -1.0.

(8) 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 & 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 * dimension2;
        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 * dimension1 * dimension1;
        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

(9) Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account and the other current account. The savings account provides compound interest and withdrawal 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 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. Derive classes Cur-acct and Sav-acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks:

a) Accept deposit from customer and update the balance.

b) Display balance.

c) Compute & deposit interest

d) Permit withdrawal and update the balance

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

```
abstract class
BankAccount {
        private String accountNumber;
        private double balance;
        pubic BankAccount (String accountNumber, double balance ){
        this.accountNumber = accountNumber;
        this.balance = balance;
        }
public String getAccountNumber ( ){
        return accountNumber;
}
        public double getBalance ( ){
                return balance;
        }
protected void setBalance (double balance ){
        this.balance = balance;
        }
```

```
public abstract void deposit (double amount);
public abstract void withdraw (double amount);
}

class SavingsAccount extends BankAccount {
  public SavingsAccount (String accountNumber, double balance) {
    super (accountNumber, balance);
  }

  public void deposit (double amount)
  {
    setBalance (getBalance() + amount);
    System.out.println ("Deposit of $" + amount + "
                        Successful. Current balance: $" +
                        getBalance());
  }

  public void withdraw (double amount)
  { if (getBalance() >= amount) {
    setBalance (getBalance() - amount);
    System.out.println ("Withdrawal of $" + amount +
                        " successful. Current balance: $" +
                        getBalance());
  }
else {
    System.out.println ("Insufficient funds withdrawal failed.");
  }
}

class CurrentAccount extends BankAccount {
  public CurrentAccount (String accountNumber,
                         double balance) {
    super(accountNumber, balance);
  }
  public void deposit (double amount)
  {
    setBalance (getBalance() + amount);
```

```
System.out.println("Deposit of $" + amount + " successful. Current
                    balance: $" +
                    getBalance());
    }

    public void withdraw(double amount) {
        if (getBalance() >= amount) {
            setBalance(getBalance() - amount);
            System.out.println("Withdrawal of $" + amount + " successful.
                    Current balance: $" + getBalance());
        }
        else {
            System.out.println("Insufficient funds. Withdrawal failed.");
        }
    }
}

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

        double ibal, damt, wamt;
        ibal = 1000.00;
        SavingsAccount savingsAccount = new SavingsAccount("SA001", ibal);
        System.out.println("Savings A/c: Initial Balance: $" + ibal);
        damt = 500.00;
        savingsAccount.deposit(damt);
        wamt = 250.00;
        savingsAccount.withdraw(wamt);
        wamt = 1600.00;
        System.out.println("\nTrying to withdraw: $" + wamt);
        SavingsAccount.withdraw(wamt);
        System.out.println();
        ibal = 5000.00;
        CurrentAccount currentAccount = new CurrentAccount("CA001", ibal);
        System.out.println("Current A/c: Initial Balance: $" + ibal);
        damt = 2500.00;
        currentAccount.deposit(10000.0);
        wamt = 12500.00;
        currentAccount.withdraw(30000.0);
        wamt = 6000.00;
```

System.out.println("\n Try to withdraw: $" + amount);
savingsAccount.withdraw(amount);
}
}

Output

Savings A/c : Initial Balance : $1000.0
Deposit of $500.0 successful. Current balance: $1500.0
Withdrawal of $250.0 successful. Current balance: $1250.0.

Try to withdraw: $1600.0
Insufficient funds.
Withdrawal failed

Current A/c : Initial Balance : $5000.0

Deposit of $1000.0
successful. Current balance: $6000.0
Withdrawal of $3000.0 successful.
Current balance: $3000.0.

Try to withdraw: $6000.0
Insufficient funds.
Withdrawal failed.

22/1/24.

Lab 2

1> Package Problem.

```
Package CIE ;
public class student {
        public String usn ;
        public string name ;
        public int sem ;

        public student (string usn, String name, int sem){
            this.usn = usn ;
            this.name = name;
            this.sem = 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 [ ] internal Marks;
public Internals (string usn, String name, int sem,
                        int [ ] internal Marks ){
        super (usn, name, sem);
    this.internal Marks = internal Marks;
}
}
```

```
Package SEE;
import CIE.student;
public class External extends Student {
    public int [] externalMarks;

public External (String usn, String name, int sem,
                 int [] externalMarks){
        super (usn, name, sem);
    this.externalMarks = externalMarks;
}
}
```

```
import java.util.Scanner;
import CIE.Internals;
import SEE.External;
public class Main {
    public static void main (String[] args){
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter the number of students:");
    int n = scanner.nextInt();
    Scanner.nextLine();
    Internals [] internalsData = new Internals [n];
    External [] externalsData = new External [n];

for(int i=0; i<n; i++) {
        System.out.print("Enter USN:");
        String usn = Scanner.nextLine();
        System.out.print("Enter name:");
        String name = Scanner.nextLine();
        System.out.print("Enter Semester:");
```

```
int sem = scanner.nextInt();
scanner.nextLine();
System.out.println("Enter Internal Marks of 5
                    subjects:");
int[] internalMarks = new int[5];
for(int j=0; j<5; j++){
    System.out.print("subject"+(j+1)+":");
    internalMarks[j] = scanner.nextInt();
}
System.out.println("Enter External Marks for 5
                    subjects:");
int[] externalMarks = new int[5];
for(int j=0; j<5; j++){
    System.out.print("subject"+(j+1)+":");
    externalMarks[j] = scanner.nextInt();
}
internalsData[i] = new Internals(usn, name, sem, internal
                                 Marks);
externalData[i] = new External(usn, name, sem, external
                               Marks);
}
for(int i=0; i<n; i++){
    System.out.println("\nStudent"+(i+1)+":");
    System.out.println("Internal Marks:" +
        arrayToString(internalsData[i].
        internalMarks));
System.out.println("External Marks:" + arrayToString(
        externalsData[i].externalMarks));
System.out.println("Total Marks(CIE+SEE):" +
        calculateTotalMarks(internalsData[i].
        internalMarks,
```

```
        externalsData[i].externalMarks));
    }
    scanner.close();
}

private static int calculateTotalMarks(int[] internalMarks,
                                       int[] externalMarks) {
    int total = 0;
    for (int i=0; i<internalMarks.length; i++) {
        total += internalMarks[i] +
                 externalMarks[i];
    }
    return total;
}

private static String arrayToString(int[] array) {
    StringBuilder sb = new StringBuilder();
    sb.append("[");
    for (int i=0; i<array.length; i++) {
        sb.append(array[i]);
        if (i != array.length-1) {
            sb.append(", ");
        }
    }
    sb.append("]");
    return sb.toString();
}
```

a) Output

Enter the number of students: 1

Enter details for student 1:

Enter USN: 1BM22CS088

Enter Name: Dhruv Rampuria

Enter Sem & Sec: 3

Enter Internal Marks for 5 subjects:

Subject 1: 50

Subject 2: 50

Subject 3: 50

Subject 4: 50

Subject 5: 50

Enter External Marks for 5 subjects:

Subject 1: 50

Subject 2: 50

Subject 3: 50

Subject 4: 50

Subject 5: 50.

Student 1:

Internal Marks: [50, 50, 50, 50, 50]

External Marks: [50, 50, 50, 50, 50]

Total Marks (CIE + SEE): 500

5) Write a program that demonstrates handling of exceptions in inheritance trees. 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 input age < 0. In Son class, implement a constructor that takes both father & son's age & throws an exception if son's age is >= father's age.

```
import java.util.Scanner;
class WrongAgeException extends Exception {
    WrongAgeException(String message) {
        super(message);
    }
}
class Father {
    private int age;
    public Father(int age) throws WrongAgeException {
        if (age < 0) {
            throw new WrongAgeException("Father's age cannot be negative");
        }
        this.age = age;
    }
    public int getAge() {
        return age;
    }
}
```

```
class Son extends Father {
    private int sonAge;

public Son(int fatherAge, int sonAge) throws WrongAgeException {
    super(fatherAge);
    if (sonAge >= fatherAge) {
throw new WrongAgeException("Son's age should be
 less than Father's age");
    }
      this.sonAge = sonAge;
}
public int getSonAge() {
    return sonAge;
    }
}
public class InheritanceExceptionDemo {
  public static void main(String[] args) {
    try {
    Scanner s = new Scanner(System.in);
   System.out.print("Enter the age of Father:");
   int fatherAge = s.nextInt();
   System.out.print("Enter the age of Son:");
   int sonAge = s.nextInt();
   Father father = new Father(fatherAge);
   System.out.println("Father's age:" + father.getAge());
    Son son = new Son(fatherAge, sonAge);
    System.out.println("Son's age:" + son.getSonAge());
} catch (WrongAgeException e) {
   System.out.println("Exception:" + e.getMessage());
} catch(Exception e) {
      System.out.println("Exception occured:" + e.getMessage());
}
}
```

Output:

Enter the age of Father: -1
Enter the age of Son: 2.
Exception: Father's age cannot be negative

Enter the age of Father: 20
Enter the age of Son: 23

Father's age: 20
Exception: Son's age should be less than Father's age.

(8) Write a program which creates two threads, one thread displaying "BMS College of Engineering" once every ten seconds and another displaying "CSE" once every 2 seconds.

```
class DisplayThread extends Thread {
    private String message;
    private int intervalMillis;
    public DisplayThread(String message, int intervalMillis) {
        this.message = message;
        this.intervalMillis = intervalMillis;
    }
    public void run() {
        while (true) {
            try {
                System.out.println(message);
                Thread.sleep(intervalMillis);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
```

```
}
}
}
```

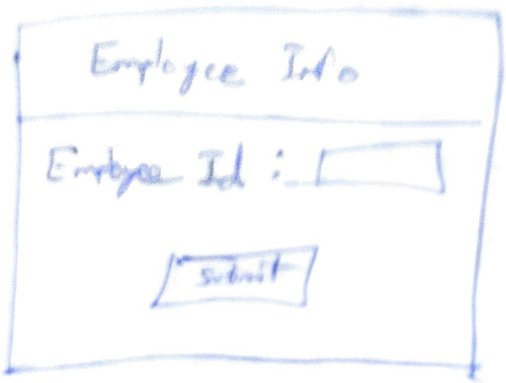
```
public class DisplayProgram {
public static void main (String [] args ){
 System.out
   DisplayThread Thread1 = new DisplayThread("BMS
   College of Engineering ",10000); // 10 seconds
  DisplayThread Thread2 = new DisplayThread("CSE",
  2000);

  Thread1.start();
  Thread2.start();

  }
}
```

Output:

BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE.

(9) Q. Creating Table, button & Text field.

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

public class awtex extends WindowAdapter {

    Frame f;
    awtex() {

        f = new Frame;
        f.addWindowListner(this);
        Label l = new Label("Employee id");
        Button b = new Button("Submit");
        TextField t = new TextField();

t.setBounds(20, 80, 80, 30);
t.setBounds(20, 100, 80, 30);
t.setBounds(100, 100, 80, 30);

f.add(b);
    f.add(l);
    f.add(t);

f.setSize(400, 300);
f.setTitle("Emp.Info");
f.setLayout(null);
f.setVisible(true);
}
```


Employee Info
Employee Id :
Submit

(9) Create a button & add 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");
    }
```

```
pulic void WindowClosing ( WindowEvent ) {
        system.exit(0);
}
pulic static void main (String args[]){
        new EventHandling;
}
}
```

OR

SSD
2/3/24