Date 8/11/24
Page

OOI - 3

1. Quadratic equation program

```
class 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");
  SOP ("enter a");
a = S.nextInt();
SOP ("enter b");
  b = S.nextInt();
  SOP ("enter c:");
  c = s.nextInt();
  }

  void discriminant() {
    d = (b*b) - (4*a*c);
    }
void calculate roots() {
    if (d>0)
{
    SOP("roots are real and unequal");
root1 = (-b + Math.sqrt(d))/(2*a);
root2 = (-b - Math.sqrt(d))/(2*a);
SOP ("first root is:" + root1);
SOP ("second root is:" + root2);
  }
```

```
else if (d == 0)
{
SOP ("Roots are real and equal");
root 1 = (-b + Math.sqrt(d)) / (2*a);
SOP ("Root: " + root 1);
}
else
{
SOP ("No real solution. Roots are imaginary");
double real = -b / (2*a);
double imagenary = Math.sqrt(-d) / (2*a);
SOP ("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.calculate roots ();
}
}
```

output:

output for code 1.

Quadratic equation is in the form : ax^2+bx+c
enter a : 1
enter b : 1
enter c : 1
No real solution roots are imaginary

Quadratic equation is in the form : ax^2+bx+c
enter a : 1
enter b : -2
enter c : 1
Roots are real and equal
root : 1.0

Quadratic equation is in the form : ax^2+bx+c.
enter a : 1
enter b : 5
enter c : 2
Roots are real and unequal
first root is : -0.4384471871911697
Second root is : -4.561552812808831

Quadratic equation is in the form : ax^2+bx+c
enter a : -1
enter b : -2
enter c : -5.
No real solution. Roots are imaginary
The equation has two complex roots : -1.0 + -2.0i and -1.0 + -2.0i

2. SGPA.

```
import java.util.Scanner;

class student {
    string usn;
    string name;
    int [] credits = new int [8];
    int [] marks = new int [8];

    public void accept details () {
        Scanner scanner = new Scanner (System.in);

        SOP(s: "Enter USN : ");
        usn = scanner.nextline();

        SOP (s: "Enter name :");
        name = scanner.nextline();

        SOP (" Enter details for each subject : \n");

        for (int i = 0 ; i < credits.length; i++) {
            SOP (" enter credits for sub " + (i+1) + "
                : ");
            marks [i] = scanner.nextInt ();
        }
        Scanner.close ();
    }

    /* method to cale SGPA */
    public double calculate SGPA () {
        int total credits = 0;
        int weighted sum = 0;
        double sgpa;
```

```
for (int i=0 ; i < credits.length ; i++)
    {
    total credits += credits[i];
    int grade points;

    grade points = (marks[i]/10) + 1;
if (grade points == 11) {
        gradepoints = 10;
    }
    else if (gradepoints <= 4){
        grade points = 0;
    }
    weighted sum += grade points * credits[i];
    }
ans = (double) weighted sum / (double)
total credits;
        return ans;
    }
}
public class sgpa {
public static void main (String[] args) {
Scanner scanner = new
Scanner (System.in);
    student student = new student();
    student.. accept details();
    SOP ("\n student details :");
    SOP ("USN:" + student.usn);
    SOP ("Name:" + student.name);
    /* calc and print SGPA */
double sgpa = student.calculate SGPA();
SOP ("SGPA :" + sgpa);
    scanner.close();
}
}
```

Output - 2

Enter USN : 1bm22cs059
Enter name : Jagdeesh
Enter details for each subject :

enter credits for sub 1 : 4
— marks " — : 85

enter credits for sub 2 : 4
enter marks for sub 2 : 95

enter credits for sub 3 : 3
enter marks for sub 3 : 98

enter credits for sub 4 : 3
enter marks for sub 4 : 99

enter credits for sub 5 : 3
enter marks for sub 5 : 35

enter credits for sub : 6 . 1 :
enter marks for sub : 6 96

enter credits for sub : 7 1
enter marks for sub : 7 96

enter credits for sub 8 : 1
enter marks for sub 8 : 100

student details :
USN : 1bm22cs059
name : Jagdeesh

SGPA : 8.3 //

10/24

3 java

```
import util.scanner;
class book{
  string name;
  string author;
  double price;
  int num_pages;

Book(string name, string author, double price, int num_pg){
  this.name = name;
  this.auth = auth;
  this.price = price;
  this.num_pages = num_pages;
}
void setdetails(){
  name = name;
  auth = auth;
  price = price;
  num_pages = num_pgs;
}
void get details(){
  Scanner s = new Scanner(System.in);
  SOP(s: "Enter book name: ");
  name = s.Next line();
  SOP(s: "enter auth name:");
  auth = s.nextline();
  SOP(s: "enter price:");
  price = s.next double();
  SOP(s: "Enter number of pages:");
  num_pages = s.next Int();
  SOP(x: "_ _ _ _ _ _ _ _ _ _");
}
```

```
public String toString() {
return ("Book Name :" + name + "\n Author name
: " + author + "\n Price : " + price + "\n Number of pages
: " + num_pages);
}
}
class BookMain {
public static void main (String args[]){
Scanner s = new Scanner (System.in);
int n, i;
SOP (x: "Enter number of Books");
n = s.nextInt();
Book[] books = new Book[n];
for (i = 0; i < n; i++) {
SOP ("Enter details of book " + (i+1));
books[i] = new Book( name: "", author: "", price: 0.0,
num_pages: 0);
books[i] = getdetails();
}
for (i = 0; i < n; i++) {
SOP ("Details of Book" + (i+1));
SOP (books[i]);
}
}
}
```

classmate
Date 22/1/24
Page

OOJ LAB Book

1 output:

Enter Number of books
2
Enter details of book 1
Enter book name : Wings of Fire
enter author : Colleen Hoover
enter price : 250
enter number of pages : 200

Enter details of book 2
Enter book name : Twisted Love
enter author : Ana Huang
enter price : 380
enter number of Pages : 400

Lab-4

1. Develop a J.P to create an abstract class named shape that contains 2 integers and an empty method named print area(). 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 print Area() 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);
    }
@Override
public void printArea() {
int area = dimension1 * dimension2;
    SOP("Area of Rectangle:"+area);
}
}

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

@Override
public void printArea(){
double area = 0.5 * dimension1 * dimension2;
SOP("Area of Triangle: "+area);
}
}
class Circle extends Shape{
public Circle(int radius){
super(radius, 0);
}
@Override
public void printArea(){
double area = Math.PI * dimension1 * dim2;
SOP("Area of circle: "+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:

Area of Rectangle: 50
Area of Triangle: 24.0
Area of circle: 50.265482457436691

Lab - Program-5

2. Develop a JCP to create a class bank that maintains two kinds of acc for its customers, one called savings & one more called current acc. The savings acc provides compound interest and withdrawal facilities but no cheque book facility. The current acc provides cheque book facility but no interest. Current acc holders should also maintain min bal and if the bal falls below this level a service charge is imposed.

Create a class acc that stores customer name, acc number and type of acc. Form this derive the classes Cur-acc and Sav-acc 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 bal

b) display the bal

c) permit withdrawal and update the balance. Check for the min bal, impose penalty if necessary and update the balance.

Code:

```
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;
    this.balance = balance;
}

public void deposit (double amount) {
    balance += amount;
    SOPln("deposit successful. Updated balance: " + balance);
}
public void displayBalance () {
    SOPln ("Balance for account " + accountNumber
    + ": " + balance);
public void depositInterest (double rate) {
    if ("savings".equals (accountType)) {
        double interest = balance * (rate / 100);
        balance += interest;
        SOPln("interest deposited. Updated balance: "
        + balance);
    } else {
        SOPln ("No interest for current account.");
    }
}

public void withdraw (double amount) {
    if (amount <= balance) {
        balance -= amount;
        SOPln("withdrawal successful. Updated balance: "
        + balance);
    } else {
        SOPln("Insufficient funds for withdrawal.");
    }
}
```

class SavAcct extends Account {
public SavAcct (String customerName, long accountNumber, double balance) {
super (customerName, accountNumber, "Savings", balance);
}
}

Class CurrAcct extends Account {
double minBalance;
double serviceCharge;
public CurrAcct (String customerName, long accountNumber, double balance, double minBalance, double serviceCharge) {
super (customerName, accountNumber, "Current", balance);
this.minBalance = minBalance;
this.serviceCharge = serviceCharge;
}
@Override
public void withdraw (double amount) {
if (amount <= balance - minBalance) {
balance -= amount;
S.O.P ("Withdrawal successful. Updated balance : " + balance);
} else {
S.O.P ("Insufficient funds for withdrawal. Minimum balance requirement not met.");
}
}
public void checkMinimumBalance () {
if (balance < minBalance) {
balance -= serviceCharge;
SOP ("Service charge imposed for falling below minimum balance. Updated balance: " + balance);
}
}

```
public class Bank{
public static void main(String[] args){
Scanner scanner = new Scanner(System.in);
SavAcct savingsAccount = new SavAcct("John Doe",
                                     123456789, 1000.0);

CurAcct currentAccount = new CurAcct("John Doe",
                                     987654321, 2000.0,
                                     500.0, 50.0);

SOP ln("Savings account operations:");
savingsAccount.displayBalance();
SavingAcount.deposit(500.0);
SavingAcount.depositInterest(5.0);
SavingAcount.withdraw(200.0);

SOP ln("\n Current Account operations:");
currentAcount.displayBalance();
CurrentAcount.deposit(1000.0);
CurrentAcount.withdraw(800.0);
CurrentAcount.checkMinimumBalance();

Scanner.close();
}
}
```

Output:

```
Savings Account operations:
Balance for account 123456789 : 1000.0
deposit successful. updated balance : 1500.0
interest deposited. updated Balance : 1575.0
withdrawal successful. updated balance : 1375.0
```

Current Account operations:

[Balance] for account 1987654321: 2000.0

deposit successful. Updated balance: 3000.0

withdrawal successful. Updated balance: 2000.0

classmate
Date 29/1/24
Page

Package

```
Package CIE;
public class student{
    String USN;
    String name;
    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[] 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[] SeeMarks;
    public external(String USN, String name, int
                    sem, int[] SeeMarks){
    super(USN, name, sem)
    this.seeMarks = SeeMarks;
  }
}
```

```
import CIE.Prelims;
import SEE.external;
import java.util.Scanner;
public class final marks {
    public static void main(String[] args){
        Scanner scanner = new Scanner(System.in);
        System.out.println("enter no of students");
        int n = scanner.nextInt();
        internals[] cie students = new Internal[n];
        external[] see students = new external[n];
        for(int i=0; i<n; i++){
            SOP("enter CIE details" + (i+1));
            SOP("USN:");
            String USN = scanner.next();
            SOP("Name:");
            String name = scanner.next;
            SOP("sem:");
            int sem = scanner.nextInt();
            int[] cie Marks = new int[5];
            SOP("enter cie marks for 5");
            for(int j=0; j<5; j++){
                cie marks[j] = scanner.nextInt();
            }
            cie students[i] = new internals(USN, name, sem, cie);
        }
        for(int i=0; i<n; i++){
            SOP("Enter SEE details" + (i+1));
            SOP("USN:");
            String USN = scanner.next();
            SOP("Name:");
            String name = scanner.next();
            SOP("Sem:");
            int sem = scanner.nextInt();
```

int[] SEE marks = new int[5];
SOP("enter SEE marks for 5 cie");
for(int j=0; j<5; j++) {
see marks[j] = Scanner next Int();
}
see student[i] = new external(USN, name, sem, see marks)
}
see. SOP("info of marks of students");
for(int i=0; i<n; i++) {
SOP("In details of student" + (i+1));
SOP("USN, " + cie student[i].USN);
SOP("name " + "student[i].name");
SOP("sem, " + "cie student[i].sem");
SOP("CIE marks:");
for(int j=0; j<5; j++) {
SOP(cie students[i].internal marks[j]);
}
SOP("In SEE marks:");
for(int j=0; j<5; j++) {
SOP(see student[i] see marks[j] + " ");
}
}

output:

Enter number of student: 1
Enter details for CIE of student 1
USN: 1BM22CS059
Name: avani
Sem: 2
Enter CIE marks for 5 courses:
19
38
40
39
40

enter details for SEE of student 1
USN: 1BM22CS059
Name: avani
Sem: 4
enter see marks for 5 course:
41
43
45
50
60

Final marks of students:
details of student 1
USN: 1BM22CS059
CIE marks:
38 39 40 39 40
SEE marks:
41 45 43 50 60

29/11/24

Lab-7

Q. Write a program which creates two threads one thread displaying "BMS college of Engineering" once every 10 seconds and another displaying "CSE once every 2 secs".

```
class BMS implements Runnable {
    public void run() {
    while (true) {
        try {
            SOP("BMS college of Engg");
            Thread.sleep(10000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
}

class CSE implements Runnable {
    public void run() {
        while (true) {
        try {
            SOP("CSE");
            Thread.sleep(2000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
}

public class Main {
    PSVM (String[] args) {
Thread t1 = new Thread(new BMS());
Thread t2 = new Thread(new CSE());
    t1.start(); t2.start(); } }
```

O/P-

BMS college of Engineering
CSE
CSE
CSE
CSE
CSE
BMS college of Engeneering
CSE
CSE
CSE
CSE
BMS college of Engeneering
CSE
CSE
CSE
CSE
CSE
BMS college of Engineering

Lab-3

Q. WAP that demonstrate handling of exceptions in Inheritance Tree. Create a base class called "Father" and a derived class called "Son" which extends base class. In father class implement a constructor which take the age and throws the exception WrongAge() when the input age < 0. In Son class, implement a constructor that calls both father and son's age and throws an exception if son's age is >= father's age.

```
import java.util.Scanner;
class WrongAge extends Exception
{
public WrongAge(String message)
{
super(message);
}
}
class Father
{
int fatherAge;
Father() throws WrongAge
{
Scanner s = new Scanner(System.in);
SOP("enter fathers age:")
fatherAge = So.nextInt();
if (FatherAge < 0)
throw new WrongAge("Age cannot be negative");
}
void display()
{
SOP("father age is:"+fatherAge);
}
```

class Son extends Father
{
int son Age;
Son() throws Wrong Age
{
super();
Scanner s = new Scanner(System.in);
System.out.println("enter son age:");
Son Age = s.nextInt();
if (son Age > Father Age)
{
throw new Wrong Age("Son age cannot be greater than Father's age");
}
else if (Son Age == Father Age)
{
throw new Wrong Age("Son's age cannot be equal to Father's age");
}
else if (son Age < 0)
{
throw new Wrong Age("Age cannot be negative");
}
void display()
{
super.display();
SOP("Son age is:" + son Age);
}
public class main (String [] args)
{
try {
Son s = new Son();
s.display();
}

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

Output :

Enter Fathers age : 40
Enter Sons age : 18
Fathers age is : 40
Sons age : 18

Enter Fathers age : 30
Enter Sons age : 30
Sons age cannot be equal to Fathers age

Enter Fathers age : -20
Age cannot be negative

26.02.24

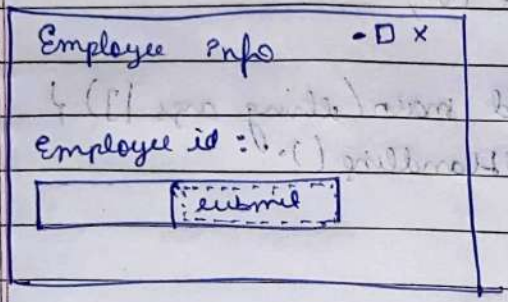
Lab - 9

Q1. Create label, button and text field in a frame using AWT

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

public class AWTExample extends WindowAdapter{
    Frame f;
    Label l;
    TextField t;
    AWTExample(){
        f = new Frame();
        f.addWindowListener(this);
        l = new Label("employee id:");
        Button b = new Button("submit");
        t = new TextField();
        l.setBound(20, 80, 80, 30);
        t.setBounds(20, 100, 80, 30);
        b.setBound(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);
    }
```

```
PS void windowClosing (WindowEvent e) {
    System.exit(0);
}
P S void main (String [] args) {
    AWTExample obj = new AWTExample();
}
```

O/P



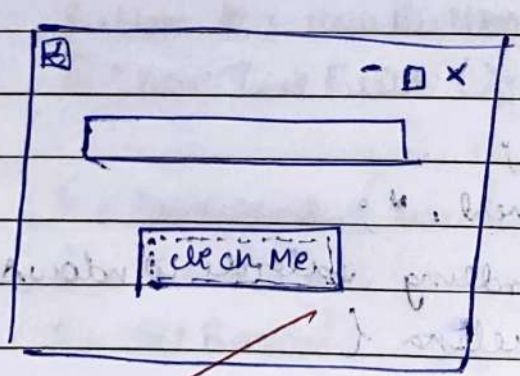
Lab-10

Q2. 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;
    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(l);
f.setSize(300,300);
f.setLayout(null);
f.setVisible(true);
}
public void actionPerformed(ActionEvent e)
{ l.setText("Welcome");
}
public void WindowClosing(WindowEvent e){
System.exit(0);
}
public static void main(String args[]){
new EventHandling();
}
}

output:



26.02.24