B.M.S COLLEGE OF ENGINEERING BENGALURU
Autonomous Institute, Affiliated to VTU

Submitted in partial fulfillment of the requirements for record of

OBJECT ORIENTED JAVA PROGRAMMING

(23CS3PCOOJ)

Submitted by:

SHREYAS GOUDA M
1BM22CS270
SECTION:-3'E'
BATCH-3

Faculty incharge:

Dr Seema Patil

Department of Computer Science and Engineering
B.M.S College of Engineering
Bull Temple Road, Basavanagudi, Bangalore 560 019

19/12/2023

2) Develop a Java program to create a class student with members usn, name, an array of credits and array marks. Include methods to accept and display and to calculate SGPA of a student.

Program:

```
import java.util.Scanner;
class Subject {
    int subjectMarks;
    int credits;
    int grade;
}
class Students {
    String name;
    String usn;
    double SGPA;
    Scanner s;
    Student() {
    int i;
    Subject
    student() {
    {
        int i;
        subject = new Subject[9];
        for(i=0; i<9; i++){
            subject[i] = new Subject();
        s = new scanner(System.in);
    }
```

```
void getStudentDetails() {
        System.out.print("Enter the USN");
        usn = s.next();
        System.out.print("Enter the name");
        name = s.next();
    }
void getMarks(){
        for(int i=0; i<8; i++){
        System.out.println("Enter marks and credits for subjects" + (i+1) + ");
        System.out.print("Marks: ");
        subjects[i].subjectMarks = s.nextInt();
        Subjects.out.print("Credits: ");
        subjects[i].credits = s.nextInt();
    }
void computeSGPA(){
        if(Subject[i].grade>11)
            subject[i].grade = 10;
        if(subject[i].grade<=4)
            subject[i].grade = 0;
    }
}
void computeSGPA()
{
    int totalCredits = 0;
    for(int i=0; i<8; i++)
    {
        Sgpa += (subject[i].credits * subject[i].grade);
```

```
            total Credits += subject[i].credits;
        }
}
class main
{
    public static void main (String args[])
    {
        Student S1 = new Student();
        S1.getStudentDetails();
        S1.getMarks();
        S1.computeSGPA();
        System.out.println("Name:"+S1.name);
        System.out.println("USN:"+S1.usn);
        System.out.println("SGPA:"+S1.sgpa);
    }
}
```

Output

```
Enter the USN
1BM22CS270
Enter the name
Shreyas Gowda M
Enter marks for subject 1:
80
Enter Credits for subject 1:
4
Enter marks for subject 2:
95
Enter Credits for subject 2:
4
Enter marks for subjects 3:
85
Enter Credits for subject 3:
3
Enter marks for subject 4:
85
Credits for subject 4:
3
Enter marks for subject 5
78
Enter credits for subject 5
3
Enter marks for subject 6
81
Enter credits for subject 6
3
Enter marks for subject 7
99
Enter credits for subject 7
1
Enter marks for subject 8
100
Enter credits for subject 8
1
Name: Shreyas Gowda M
USN: 1BM22CS270
SGPA: 8.971256
```

10

14/12/23

26/12/23

3 Create a class Book which contains four members: name, author, price, num_pages. Include a constructor to set the values for the members, include To to String() method that could display the complete details of the book. Develop a java program to create n book objects

Program:

```
import java.util.Scanner;
class Books {
    String name;
    String author;
    int price;
    int numPages;
    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 bookdetails = "Book name: " + this.name
+ "\n" + "author name: " + this.author + "\n" +
"Price:" + this.price + "\n" + "Number of Pages:" +
this.numPage + "\n";
        return bookdetails;
    }
}
```

26/12/23

```
public class Main {
    public static void main (String args[]){
        Scanner s = new Scanner (System.in);
        System.out.print ("Enter number of books:");
        int n = s.nextInt();
        Books[] books = new Book[n];
        for (int i = 0; i<n; i++) {
            system.out.println("Enter details of book"+(i+1));
            System.out.println("Name: ");
            String name = s.next();
            System.out.println("author: ");
            String author = s.next();
            System.out.println("price: ");
            int price = s.nextint();
            books[i] = new Book(name, author, price, numPages);
        } system.out.println("Book details:");
        for(i=0; i<n; i++) {
            system.out.println("Book details"+(i+1)+"\n");
            System.out.println("Book" + books[i]);
        }
        s.close();
        System.out.println("Book details:\n");
        for (int i=0; i<n; i++)
            System.out.println("Book "+(i+1)+":\n"
+ books[i]);
    }
}
```

26/12/23

Enter the number of books : 3

Enter details for book 1 :
Name : Percy_Jackson
Author : Rick_Riordan
Price : 1500
Number of pages : 130

Enter detail for book 2 :
Name : Coffe can_ investing
Author : Dragan_Thai
Price : 560
Number of pages : 300

Enter details for book 3 :
Name : Harry_Potter
Author : J.K_Rowling
Price : 570.
Number of pages : 700

Book details :
Book 1 :
Name : Percy_Jackson
Author : Rick_Riordan
Price : 1500
Number of pages : 130

26/12/23

Book 2 :
Name : Coffe_Can_investing
Author : Dragan_Thai
Price : 560
Number of pages : 300
Book 3 :
Name : Harry_Potter
Author : J_K_Rowling
Price : 570
Number of pages : 700

Tue
26/12/23

02/01/24

Lab - 4

4) 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 contains the method printArea() that prints the area of the given shape.

Program:

```
import java.util.Scanner;
class InputScanner {
    Scanner s;
    InputScanner() {
        s = new Scanner(System.in);
    }
}

abstract class Shape extends InputScanner {
    double a, b;
    abstract void getInput();
    abstract void displayarea();
}

class Rectangle extends Shape {
    void getinput() {
        System.out.print("s: Enter length");
        a = s.nextdouble();
        System.out.print("s: Enter breadth");
        b = s.nextdouble();
    }
    void displayarea() {
        System.out.println("Area of rectangle = " + (a * b));
    }
```


02/01/24

```
class Right_triangle extends Shape {
    void getInput() {
        System.out.println("Enter base:");
        a = s.nextDouble();
        System.out.println("Enter height:");
        b = s.nextDouble();
    }
    void displayArea() {
        System.out.println("Area of triangle = " + (0.5 * a * b));
    }
}
class Circle extends Shape {
    void getInput() {
        System.out.println("Enter radius");
        a = s.nextdouble();
    }
    void displayArea() {
        System.out.println("The area of circle is = " + (3.14 * a * a));
    }
}
Public class MainArea {
    Public Static void main(String[] args) {
        Rectangle r = new Rectangle();
        Triangle t = new Triangle();
        Circle c = new Circle();
        r.getInput();
        r.displayarea();
        t.getInput();
        t.displayarea();
        c.getInput();
        c.displayarea();
```

02/01/24

```
    }
}
```

Output:
```
Enter length: 30
Enter breadth: 50
Area of rectangle = 1500.0
Enter base: 30
Enter height: 10
Area of Triangle = 150.00
Enter radius: 21
Area of circle = 1385.2692
```

02/01/24

09/01/24

Lab - 5

5) Develop a Java program to create a class Bank that maintains two kinds of account for it's customers, one called savings account and the current account. The savings account provides cheque book facility but no interest. Current account holders should also maintain minimum balance and service charge is imposed.

- Create a class Account that stores customer name, account number and type of account from this derive the class Current and Savings make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks:

- a) Accept deposit update the balance
- b) Display the balance
- c) Compute and deposit interest
- d) Permit withdrawal and update the balance Check for the minimum balance. impose penalty if necessary

09/01/24

Program:

```
import java.util.Scanner;

class input {
    Scanner sc = new Scanner(System.in);
}
class accounts extends input {
        String name;
        int accNo;
        double balance;
        public void getDetails() {
            System.out.print("Enter name: ");
            name = sc.nextLine();
            System.out.print("Enter account number: ");
            accNo = sc.nextInt();
        }
        public void deposit() {
            System.out.print("Enter deposit: ");
            double amt = sc.nextdouble;
            balance += amt;
            System.out.print("amount deposited");
        }
        public void withdraw() {
            System.out.println("Enter amount");
            double amt = sc.nextdouble();
            if (balance >= amt) {
                balance -= amt;
                System.out.print("successfull");
            }
            else
                System.out.println("Insufficient Balance");
        }
        public void display() {
            System.out.println("Name: " + name);
            System.out.println("Acc No: " + accNo);
            System.out.println("balance: " + balance);
        }
}
class Savings extends accounts {
        final double intrest rate = 0.04;
        public void compute Intrest() {
            double intrest = balance * intrest;
            balance += intrest;
            System.out.println("Credited");
        }
}
class Current extends accounts {
        final double minbalance = 500;
        final double penalty = 100;
        public void withdraw() {
            super.withdraw();
            checkMinBalance();
        }
        public void checkMinBalance() {
            if (balance < minbalance) {
                Balance -= penalty;
                System.out.println("Penalty applied");
```

```
    }
  }
}

class Bank1 extends Account {
  public static void main (String [] args) {
    Scanner sc = new Scanner(System.in);
    Savings ob1 = new Savings();
    Current ob2 = new Current();
    ob1.getdetails();
    ob2.getdetails();
    int choice;
    String acc;
    System.out.println("Menu");
    System.out.println("1. Deposit");
    System.out.println("2. withdraw");
    System.out.println("3. Displaybalance");
    System.out.println("4. Compute Intrest (Savings only)");
    System.out.println("5. exit");
    do {
      System.out.print("Enter your choice:");
      choice = sc.nextInt();
      System.out.print("Enter acc type:");
      acc = sc.next();
      switch (choice) {
        case 1:
          if (acc.equals("saving"))
            ob1.deposit(); else
            ob2.deposit();
          break;
        case 2:
          if (acc.equals("saving")) ob1.withdraw();
          else ob2.withdraw();
          break;
        case 3:
          if (acc.equals("saving")) ob1.display();
          else ob2.display();
        case 4:
          ob1.computeIntrest();
          break;
        case 5:
          System.out.println("Existing...");
          break;
        default:
          System.out.println("Invalid choice");
          Exit(0);
          break;
      }
    } while (choice != 5);
  }
```

output :

```
Enter name : Shreyas K
Enter account number : 12345
Enter name : Shreyanka  Enter account number : 1478
Menu :
1. Deposit
2. Withdraw
3. Display Balance
4 Compute Interest (Savings only)
5 Exit
Enter your choice : 1
Enter the acc type :
Savings
Enter amount to deposit : 1000
Amount deposited successfully.
Enter your choice : 2
Enter the acc type :
Current
Enter amount to withdraw : 500
Amount withdrawn successfully.
Enter your choice : 3
Enter the acc type :
Savings
Name : John
Account Number : 12345
Balance : 1000.0
Enter your choice : 4
Enter the acc type :
Savings
interest credited : 40.0
Enter your choice : 5
Exiting ....
```

Lab - 6

1) write a Java Program. to create a generic class Stack which hold 5 integers and 5 double values

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

class Stack<T>{
    private static final int MAX_SIZE = 5;
    private static final int MAX_SIZE = 5;
    private Object[] elements;
    private int top;
    public Stack(){
        if (top < MAX_SIZE - 1){
            elements[++top] = element;
            System.out.println("Pushed," + element);
        }
        else{
            System.out.println("Stack is full.
Cannot push" + element);
        }
}
    public T pop(){
        if(top >= 0){
            T popped Element = (T) elements
[top--];
            system.out.println("Stack is
Empty");
            return null;
        }
}
```

```
    public void display(){
        if (top >= 0){
            System.out.println("Stack elements:");
            for(int i=0; i<= top; i++){
                System.out.println(elements[i]
+ " ");
            }
            system.out.println();
        }
    }
}
public class main{
    public static void main(String[] args){
        Stack<Integer> integer Stack = new Stack<>();
        Stack<double> double Stack = new Stack<>();

        integer stack.push(1);
        integer stack.push(2);
        integer stack.push(3);

        double stack.push(1.11);
        double Stack.push(2.22);
        double Stack.push(3.333);

        System.out.println("integer Stack");
        integer stack.display();

        System.out.println("Double Stack");
        integer
        double stack.display();
```

integer Stack.pop(2);
double Stack.pop(3.3);
}
}

Output:
Pushed: 1
Pushed: 2
Pushed: 3
Pushed: 1.1
Pushed: 2.22
Pushed: 3.333
Integer Stack:
Stack elements: 1 2 3
double Stack:
Stack elements: 1.1 2.22 3.333
Popped: 2
Popped: 3.333

16/01/24

week - 6

Lab 7

6) Create a package CIE which has two classes - Student and 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 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.

Program

```
// Package CIE
packa
package CIE;
import java.util.Scanner;
public class Student {
    protected String usn;
    protected String name;
    protected int sem;
    public void inputStudentDetails() {
        Scanner S = new Scanner(System.in);
        System.out.print("Enter USN: ");
        usn = scanner.nextLine();
        System.out.print("Enter name: ");
        name = scanner.nextLine();
        System.out.print("Enter Semester: ");
        sem = scanner.nextInt();
$   }
```

```
public void displayStudentdetails() {
    System.out.println("USN: "+USN);
    System.out.println("Name: "+name);
    System.out.println("Semester: "+sem);
  }
}
public class Internal extends Student {
    protected int marks[] = new int[5];
    public void inputCIEmarks() {
        Scanner scanner = new Scanner(System.in);
        for(int i=0; i<5; i++){
            System.out.println("Enter Internal marks
for subject "+(i+1)+": ");
            marks[i] = scanner.nextInt();
        }
    }
    public void displayCIEMarks(){
        System.out.println("Internal Marks: ");
        for(int i=0; i<5; i++){
            System.out.println("Subject "+(i+1)
+": "+marks[i]);
        }
    }
}
```

```
package SEE;
import CIE.Internals;
import java.util.Scanner;
public class Externals extends CIE.Internals {
    protected int marks[];
    protected int finalMarks[];
    public Externals(){
        marks = new int[5];
        finalMarks = new int[5];
    }
    public void inputSEEmarks(){
        Scanner scanner = new Scanner(System.in);
        for(i=0; i<5; i++){
            System.out.print("Enter External marks
for subject "+(i+1)+": ");
            marks[i] = scanner.nextInt();
        }
    }
    public void calculateFinalMarks(){
        for(int i=0; i<5; i++){
            finalMarks[i] = marks[i]+
Sub1.mark[i]/2;
        }
    }
    public void displayFinalMarks(){
        displayStudentdetails();
        displayCIEMarks();
        System.out.println("Final Marks:");
        for(int i=0; i<5; i++){
```

```
            System.out.println("Subject " +
(i+1)+" : " + finalMarks[i]);
        }
    }
}
```

Main

```
public class javamain {
    public static void main(String args[]){
        int numofStudents = 2;
        External finalMarks[] = new External
[numofStudents];
        for(int i=0; i< numofStudents ;i++){
            finalMarks[i] = new External();
            finalMarks[i].inputStudentDetails();
            System.out.println("Enter CIE marks");
            finalMarks[i].inputCIEmarks();
            System.out.println("Enter SEE marks");
            finalMarks[i].inputSEEmarks();
        }
        System.out.println("Displaying data:\n");
        for(int i=0; i< numofStudents; i++){
            finalMarks[i].calculateFinalMarks();
            finalMarks[i].displayFinalMarks();
        }
    }
}
```

output

```
Enter number of Students: 1
Enter USN: 1BM22CS201
Enter name: JohnyBarsalow
Enter semester: 5
Enter CIE marks
Enter internals for subject 1: 45
Enter internals for subject 2: 50
Enter internals for subject 3: 55
Enter internals for subject 4: 60
Enter internals for subject 5: 48

Enter finals for subject 1: 90
Enter finals for subject 2: 95
Enter finals for subject 3: 85
Enter finals for subject 4: 62
Enter finals for subject 5: 73

USN: 1BM22CS201
Name: Johny Barsalow
Internal marks:
subject1: 45
subject2: 50
subject3: 55
subject4: 60
subject5: 48
Final marks:
subject1: 90
subject2: 97.5
subject3: 67.5
subject4: 62
subject5: 55.5
```

Week - 7

Lab - 8

8 write a program that demonstrates Handling of Exceptions in interface. Write a class called father and derived class called son. make a constructor which throws an exception if age is negative and a constructor son which throws an exception if son age is greater than father age

Program:

```
import java.util.Scanner
class Wrongage extends exception {
    public wrongage() {
        super("age error");
    }
    public Wrongage(String message) {
        super(message);
    }
}
class input scanner {
    protected Scanner scanner;
    public input scanner() {
        scanner = new scanner(System.in);
    }
    public int nextint() {
        return scanner.nextint();
    }
}
class father extends input scanner {
    public int father age;
    public father() throws wrongage {
        System.out.println("Enter fathers age");
        father Age = super.nextint();
        if (father age < 0) {
            throw new Wrongage("Age can not be negative");
        }
    }
}
class son extends Father {
    protected int sonage;
    public son() throws wrongage {
        super();
        system.out.println("Enter sons's age");
        sonAge = super.nextint();
        if (sonage >= father age || sonage < 0) {
            throw new wrongage("son's age can not be
greater or equals to faher or negatuve");
        }
    }
}
```

```
public class thisMain {
    public static void main (String [] args) {
        try {
            Son son1 = new Son ();
            System.out.println ("Father's age:" +
son1.fatherage );
            System.out.println ("Son's age:" + son1.son
-age );" }

        catch (Wrongage e){
            System.out.println ("Exception" +
e.getMessage() );"
        }
    }
}
```

output: Enter Father's age: 40
Enter Son's age: 25

Father's age: 40
Son's age: 25

Enter Father's age:
50
Enter Son's age:
55
Exception: Son's age cannot be greater than or Equal to father's age or negative.

Week - 8

Lab - 9

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

Program:

```
import java.util.Scanner
class display BMS implements Runnable {
    public void run() {
        while (true) {
            System.out.println("BMS College of Engi");
            try {
                Thread.sleep(10000);
            }
            Catch (InterruptedException e) {
                e.printStackTrace
            }
        }
    }
}

class displayCSE implements Runnable {
    public void run() {
        try {
            while (true) {
                System.out.println("BMS college of Engineering CSE");
                Thread.sleep(millis: 2000);
            }
        }
    }
}
```

```
public class Sleep_main {
    public static void main(String[] args) {
        Thread thread1 = new Thread(new displayBMS());
        Thread thread2 = new Thread(new displayCSE());
        thread1.start();
        thread2.start();
    }
}
```

Output:

```
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of engineering
CSE
CSE
CSE
CSE
CSE
```

Week - 10 / - Lab - 10

Q demonstrate deadlock and inter process communication

Program

```
// Producer Consumer problem
import java.util.Scanner;
class Q{
    int n;
    boolean value set = false;
    synchronized int get(){
        while( ! value set){
            try {
                System.out.println("Consumer waiting");
                wait();
            } cat (InterruptedException E){
                System.out.println(InterruptedException);
            }
        System.out.println("Got: "+n);
        valueset = false;
        System.out.println("Intimate producer");
        notify();
        return n;
    }
}
synchronized void put(int n){
    while(valueset)
        try{
            System.out.println("Producer waiting");
            wait();
        } catch(InterruptedException e){
            System.out.println("Interrupt Exception");
        }
    this.n = n;
    valueset = true;
    System.out.println("Put: "+n);
    System.out.println("Intimate Consumer");
    notify();
}
class produce implements runnable{
    Q q;
    Produce(Q q){
        this.q = q;
        new Thread(this, name: "Producer").start()
    }
    public void run(){
        int i = 0;
        while(i<15){
            q.put(i++);
        }
    }
}
```

```
class consumer implements Runnable{
    Q q;
    consumer (Q q)
    this.q=q;
    new Thread(this, "consumer").start();
    public void run(){
        int i=0;
        while(i<15){
            int r=q.get();
            System.out.println("consumed:"+r);
            i++;
        }
    }
}
class PCFixed{
    public static void main(String args[]){
        Q q=new Q();
        new producer(q);
        new consumer(q);
        System.out.println("Press control-c to stop");
    }
}
```

output:

```
put: 0
intimate consumer
producer waiting
Got: 0
intimate producer
put: 1
intimate Consumer
producer waiting
consumed: 0
0
Got: 1
Intimate Producer
consumed:
put: 2
```

Deadlock

```
class A{
    synchronized void foo(B b){
        String name = Thread.currentThread().getName();
        System.out.println(name + " entered A.foo");
        try{
            Thread.sleep(1000);
        } catch(Exception e){
            System.out.println("A interrupted");
        }
        System.out.println(name + " trying to call B.last()");
        b.last();
    }
    void last(){
        System.out.println("Inside A.last");
    }
}
class B{
    synchronized void bar(A a){
        String name = Thread.currentThread().getName();
        System.out.println(name + " entered B.bar");
        try{
            Thread.sleep(1000);
        } catch(Exception e){
            System.out.println("B interrupted");
        }
        System.out.println(name + " trying to call A.last()");
        a.last();
    }
    void last(){
        System.out.println("Inside A.last");
    }
}
class Deadlock implements Runnable{
    A a = new A();
    B b = new B();
    Deadlock(){
        Thread.currentThread().setName("MainThread");
        Thread t = new Thread(this);
        t.start();
        a.foo(b);
        System.out.println("Back in main thread");
    }
    public void run(){
        b.bar(a);
        System.out.println("Back in other thread");
    }
}
```

Output:
Main Thread entered A.foo
Racing Thread entered B.bar
Main Thread trying to call B.last()
Inside A.last
Back in main thread
Racing Thread trying to call A.last()
Inside A.last
Back in other thread

13.02.24

20/2/24

Lab - 9

Q WAP that creates a user interface to perform integer division. The user enters two numbers in the text fields, num1 & num2. The division of num1 and num2 is displayed in result field when the div. is of num1 & num2 were not an integer. the program would throw an arithmetic exception display the message dialog box.

Program:

```
class swingdemo{
import javax.swing;
import java.awt;
import java.awt.event;

class swingDemo{

    swingDemo(){
        JFrame jfm = new JFrame("Swing apps");
        jfm.setSize(275,180);
        jfm.setLayout(new FlowLayout());
        jfm.setDefaultCloseOperation(JFrame.
EXIT_ON_CLOSE);
        JTextField num1 = new JTextField(8);
        JTextField num2 = new JTextField(8);
        JLabel alab = new JLabel();
        JLabel alab = new JLabel();
```

```
ActionListener l = new ActionListener(){
    public void actionPerformed(ActionEvent){
        System.out.println("Action Event from a text");
    }
};
asf.addActionListener(l);
bsf.addActionListener(l);

button.addActionListener(new ActionListener){
    public void actionPerformed{
        try{
            int a = Integer.parseInt(ajt.getText());
            int b = Integer.parseInt(bjt.getText());
            alab.setText("\n A = "+a);
            blab.setText("\n ans = "+ans);
        }catch(NumberFormatException e)
        {
            alab.setText("");
            blab.setText("");
            ansLab.setText();
        }catch(ArithmeticException e){
            alab.setText("");
            blab.setText("");
            ans.setText("0 should be seen");
        }
    }
});
    }
    fin(setVisible(true);
}
```

```
public static void main(String args[]){
    SwingUtilities.invokeLater(new Runnable()
    {
        public void run(){
            new SwingDemo();
        }
    }
}
};
```

Output: Enter the divider and dividend
10 2
calculate A = 10 B = 2 Ans = 5

Shreyas Gowda M
1BM22CS270

Functions: JFrame :- The javax.swing.JFrame class is a type of container, which inherits the java.awt.JFrame works like the main window where components like labels, text fields are added

setSize(int width, int height) :- used to resize a frame using width & height

12/12/2023

1) Develop a Java program that prints all real solutions to quadratic equation $ax^2+bx+c=0$, read in a, b, c and use the quadratic formula. if b^2-4ac is negative, display a message stating that there are no real solutions.

Program :-

```
import java.util.Scanner;
class quadratic
{
    int a,b,c;
    double r1,r2,d;
    void getd()
    {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter a,b,c");
        a = s.nextInt();
        b = s.nextInt();
        c = s.nextInt();
    }
void compute()
{
    while(a==0)
    {
        System.out.println("Not a quadratic Equation");
        System.out.println("Enter non zero value");
        a = s.nextInt();
    }
```

```
d = b*b - 4*a*c;
if (d == 0)
{
    r1 = (-b / 2*a);
    System.out.println("roots are real and equal");
    System.out.println("Root 1 = Root 2 = " + r1);
}
else if (d > 0)
{
    r1 = (-b) + (Math.sqrt(d)) / (double)(2*a);
    r2 = ((-b) - (Math.sqrt(d)) / (double)(2*a);
    System.out.println("roots are real and distinct");
    System.out.println("R1 = " + r1 + " and R2 = " + r2);
}
else if (d < 0)
{
    System.out.println("roots are imaginary");
    r1 = (-b) / (2*a);
    r2 = Math.sqrt(-d) / (2*a);
    System.out.println("R1 = " + r1 + " + i " + r2);
    System.out.println("R2 = " + r1 + " - i " + r2);
}
}
class Quadratic Main
{
public static void main (String args[])
{
Quadratic q = new Quadratic();
q.getd();
q.Compute();
}
}
```

output

Enter the Conjugates of a, b, c
1
5
2
roots are real and distinct
r1 = -0.43944 and r2 = -4.5615

Enter the Conjugate of a, b, c
4
-4
1
Roots are real and equal
root 1 = root 2 = 1/2

Enter the Conjugate of a, b, c
1
1
1
The roots are imaginary
root1 = -0.5 + i 1.732 or
root2 = -0.5 - i 1.732

Shreyas Gowda 4G 1BM22CS270

10
12/12/23

Set Layout ( ) - method allows you to the layout of the containers. The Layout managers help to layout the components held by their containers.

Set default (close operation ( ) - methods is used to specify one of several options for the close button JFrame.

JLabel - The object of JLabel class is a component for placing text on a container. It is used to display a single read / write

JText Field - the object of a JText Field class is text component that allows the editions of a single line text. It inherits JText component class.

add (frame) - add new frame to existing ones

Action Listener - the java ActionListener is notified whenever you click on the button

set Text ( ) - this method substitutes new text for all part of text in text field.

setVisible ( ) is a method that has returns type boolean.

23.09.24

LAB: 1

Develop a Java program that prints all real solutions to the quadratic equation $ax^2+bx+c = 0$. Read a, b, c and use the quadratic formula. If the discriminate b^2-4ac is negative, display a stating that there are no real solutions

```
import java.util.Scanner;
class Quadratic
{
	int a, b, c;
	double r1, r2, d;
	void getd()
	{
		Scanner s = new Scanner(System.in);
		System.out.println("Enter the coefficients of a,b,c");
		a = s.nextInt();
		b = s.nextInt();
		c = s.nextInt();
	}
	void compute()
	{
		while(a==0)
		{
			System.out.println("Not a quadratic equation");
			System.out.println("Enter a non zero value for a:");
			Scanner s = new Scanner(System.in);
			a = s.nextInt();
		}
		d = b*b-4*a*c;
		if(d==0)
		{
			r1 = (-b)/(2*a);
			System.out.println("Roots are real and equal");
			System.out.println("Roo1 = Root2 = " + r1);
		}
		else if(d>0)
		{
			r1 = ((-b)+(Math.sqrt(d)))/(double)(2*a);
			r2 = ((-b)-(Math.sqrt(d)))/(double)(2*a);
			System.out.println("Roots are real and distinct");
			System.out.println("Roo1 = " + r1 + " Root2 = " + r2);
		}
		else if(d<0)
		{
			System.out.println("Roots are imaginary");
			r1 = (-b)/(2*a);
			r2 = Math.sqrt(-d)/(2*a);
			System.out.println("Root1 = " + r1 + " + i"+r2);
			System.out.println("Root1 = " + r1 + " - i"+r2);
		}
	}
}
```

```
class QuadraticMain
{
	public static void main(String args[])
	{
		Quadratic q = new Quadratic();
		q.getd();
		q.compute();
	}
}
```

LAB: 2

Develop a Java program to create a class Student with members usn, name, an array credits and an array mark. Include methods to accept and display details and a method to calculate SGPA of a student.

//Develop a java program to create a class Student with members usn,name, an array creadits and an array marks. Include methods to accept and display details and a method to calculate SGPA of a student

```
import java.util.Scanner;
class Student{
   String name;
   String usn;
   double SGPA;
   Subject subject[];
   Scanner s;

   Student(){
      int i;
      subject=new Subject[9];
      for(i=0;i<9;i++){
         subject[i]=new Subject();
         s=new Scanner(System.in);
      }

   }

   void getStudentDetails(){
      System.out.println("Enter your Name:");
      name=s.next();
      System.out.println("Enter your USN:");
      usn=s.next();
   }

   void getMarks(){
      for(int i=0;i<9;i++){
         System.out.println("Enter marks for subject "+(i+1)+":");
         subject[i].subjectMarks=s.nextInt();
         System.out.println("Enter credits for subject "+(i+1)+":");
         subject[i].credits=s.nextInt();

         subject[i].grade=(subject[i].subjectMarks/10)+1;

         if (subject[i].grade==11){
            subject[i].grade=10;
         }
         if (subject[i].grade<=4){
            subject[i].grade=0;
         }

      }
   }

   void computeSGPA(){
      int effectiveScore=0;
```

```
        for(int i=0;i<9;i++){
            effectiveScore += (subject[i].grade*subject[i].credits);
            totalCreadits += subject[i].credits;
        }

        SGPA=(double)effectiveScore/(double)totalCreadits;
    }
}
class Subject{
        int subjectMarks;
        int credits;
        int grade;

}
class Main{
        public static void main(String args[]){
                Student s1=new Student();
                s1.getStudentDetails();
                s1.getMarks();
                s1.computeSGPA();

                System.out.println("Name:"+s1.name);
                System.out.println("USN:"+s1.usn);
                System.out.println("SGPA :"+s1.SGPA);
        }
}
```

LAB: 3

Create a class Book which 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 the book. Develop a Java program to create n book objects.

```
import java.util.Scanner;
class Book{
   String name, author;
   int price,page_num;

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

   String toStrings(){
      String name,author,price,page_num;
      name="Book name: "+this.name+"\n";
      author="Author name: "+this.author+"\n";
      price="Price: "+this.price+"\n";
      page_num="Number of pages: "+this.page_num+"\n";

      return (name+author+price+page_num);
   }

   public static void main(String args[]){
      int n;
      String name,author;
      int price,page_num;

      Scanner sc=new Scanner(System.in);

      System.out.println("Enter the no. of books:");
      n=sc.nextInt();

      Book b[]=new Book[n];

      for (int i=0;i<n;i++){
         System.out.println("Enter name of book:");
         sc.nextLine();
         name=sc.nextLine();

         System.out.println("Enter author of a book:");
         author=sc.nextLine();

         System.out.println("Enter the price of book:");
         price=sc.nextInt();

         System.out.println("Enter the no. of pages of book:");
```

```
            b[i]=new Book(name,author,price,page_num);
        }

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

LAB: 4

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 only the method printArea( ) that prints the area of the given shape.

```
import java.util.Scanner;
public class InputScanner {
	Scanner s;
	InputScanner(){
		s=new Scanner(System.in);
	}
}
abstract public class Shape extends InputScanner{
	double a,b;
	abstract void getInput();
	abstract void displayArea();
}
public class Rectangle extends Shape {
	void getInput() {
		System.out.println("Enter the dimension of rectangle:");
		a=s.nextDouble();
		b=s.nextDouble();
	}

	void displayArea() {
		System.out.println("Area of reactangle:"+(a*b));
	}
}
public class Circle extends Shape{
	void getInput() {
		System.out.println("Enter the dimension of circle:");
		a=s.nextDouble();
	}

	void displayArea() {
		System.out.println("Area of circle:"+(3.14*a*a));
	}
}
public class Triangle extends Shape {
	void getInput() {
		System.out.println("Enter the dimension of triangle:");
		a=s.nextDouble();
		b=s.nextDouble();
	}

	void displayArea() {
		System.out.println("Area of triangle:"+((a*b)/2));
	}
}
public class MainMethod {
```

```
                Rectangle R=new Rectangle();
                R.getInput();
                R.displayArea();

                Triangle T=new Triangle();
                T.getInput();
                T.displayArea();

                Circle C=new Circle();
                C.getInput();
                C.displayArea();
        }

}
```

LAB: 5

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 balance falls below this level, a service charge is imposed.

```
import java.util.Scanner;
public class Account {
	String customer_name;
	int account_no;
	String type_acc;
	double balance=0;
	Scanner sc;

	Account(String customer_name,int account_no,String type_acc){
		this.customer_name=customer_name;
		this.account_no=account_no;
		this.type_acc=type_acc;
		sc=new Scanner(System.in);
	}

	void diposite() {
		System.out.println("Enter the deposit amount:");
		int dipo=sc.nextInt();
		balance+=dipo;
	}

	void withdrawal() {
		System.out.println("Enter the withdrawl amount:");
		int with=sc.nextInt();
		if(with>balance) {
			System.out.println("Insufficient Balance");
		}
		else{
			balance-=with;
		}
	}

	void display() {
		System.out.println("Customer name:"+customer_name);
		System.out.println("Account number:"+account_no);
		System.out.println("Type of Account:"+type_acc);
		System.out.println("Balance:"+balance);
	}
	void applyinterest() {}

}

public class Cur_acct extends Account {

	Cur_acct(String customer_name,int account_no,String type_acc){
```

```
	void withdrawal() {
		System.out.println("Enter the withdrawl amount:");
		int with=sc.nextInt();
		if (balance<=2000) {
			double pen=balance/(0.06);
			System.out.println("Insuffiecient balance penalty to be paid:"+pen);
			balance+=pen;
		}
		else{
			balance-=with;
		}
	}
}

public class Sav_acct extends Account{
Sav_acct(String customer_name,int account_no,String type_acc){
super(customer_name,account_no,type_acc);
}

void applyinterest() {
System.out.println("Enter the interest rate:");
int rate=sc.nextInt();
double interest=balance*rate;
balance+=interest;
System.out.println("Balance after interest:"+balance);
}
}

import java.util.Scanner;
public class Bank {
	public static void main(String[] args) {
		Scanner sc=new Scanner(System.in);
		System.out.println("Enter customer name:");
		String cus_name=sc.nextLine();
		System.out.println("Enter account number:");
		int acc_no=sc.nextInt();

		Account ca=new Cur_acct(cus_name,acc_no,"Current Account");
		Account sa=new Sav_acct(cus_name,acc_no,"Saving Account");

		int choice;
		while(true){
			System.out.println("----MENU----");
			System.out.println("1.Deposite\n2.Withdrawl\n3.Compute intereset for Saving
			account\n4.Display account details\n5.Exit");
			choice=sc.nextInt();

			switch(choice) {
			case 1:
			System.out.println("Enter the type of account:\n1.Saving Account \n2.Current Account");
			int acc=sc.nextInt();
```

```
                    sa.diposite();
                    }
                    else {
                    ca.diposite();
                    }
                    break;
                    case 2:
                    System.out.println("Enter the type of account:\n1.Saving Account \n2.Current Account");
                    int acc1=sc.nextInt();
                    if(acc1==1) {
                    sa.withdrawal();
                    }
                    else {
                    ca.withdrawal();
                    }
                    break;
                    case 3:
                    sa.applyinterest();
                    break;
                    case 4:
                    System.out.println("Enter the type of account:\n1.Saving Account \n2.Current Account");
                    int acc2=sc.nextInt();
                    if(acc2==1) {
                    sa.display();
                    }
                    else {
                    ca.display();
                    }
                    break;
                    case 5:
                    break;
                    default:
                    System.out.println("Invalid Choice");
                    break;
                    }

                    if(choice==5) {
                    break;
                    }
                }
        }
}
```

LAB: 6

Create a package CIE which has two classes- Student and 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 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 n students in all five courses.

```
package CIE;
import java.util.Scanner;
public class Student{
   public String usn,name;
   public int sem;
   public void inputStudentDetails(){
      Scanner sc=new Scanner(System.in);
      System.out.println("Enter the student usn:");
      usn=sc.nextLine();
      System.out.println("Enter the student name:");
      name=sc.nextLine();
      System.out.println("Enter student semester:");
      sem=sc.nextInt();
   }

   public void disply(){
      System.out.println("Student USN:"+usn);
      System.out.println("Student Name:"+name);
      System.out.println("Student Sem:"+sem);
   }
}

package CIE;
import java.util.Scanner;
public class Internals extends Student{
   public int marks[]=new int[5];
   public void inputCIEmarks(){
      Scanner sc=new Scanner(System.in);
      for(int i=0;i<5;i++){
         System.out.println("Enter the marks for subject "+(i+1)+":");
         marks[i]=sc.nextInt();
      }
   }
}
package SEE;
import CIE.Internals;
import java.util.Scanner;
public class Externals extends CIE.Internals{
   public int marks[];
   public int finalMarks[];

   public Externals(){
      marks=new int[5];
      finalMarks=new int[5];
```

```
    public void inputSEEMarks(){
        Scanner sc=new Scanner(System.in);
        for(int i=0;i<5;i++){
            System.out.println("Enter subject "+(i+1)+" marks:");
            marks[i]=sc.nextInt();
        }
    }

    public void calculateFinalMarks(){
        for(int i=0;i<5;i++){
            finalMarks[i]=marks[i]/2+super.marks[i];
        }
    }

    public void displayFinalMarks(){
        inputStudentDetails();
        for(int i=0;i<5;i++){
            System.out.println("Subject "+(i+1)+" final marks:"+finalMarks[i]);
        }
    }
}
import SEE.Externals;
class PackageMain{
    public static void main(String args[]){
        int numOfStudent=2;
        Externals finalMarks[]=new Externals[numOfStudent];
        for(int i=0;i<numOfStudent;i++){
            finalMarks[i]=new Externals();
            finalMarks[i].inputStudentDetails();
            System.out.println("Enter CIE Marks:");
            finalMarks[i].inputCIEmarks();
            System.out.println("Enter SEE Marks:");
            finalMarks[i].inputSEEMarks();
        }
        System.out.println("Displaying data:\n");
        for(int i=0;i<numOfStudent;i++){
            finalMarks[i].calculateFinalMarks();
            finalMarks[i].disply();
            finalMarks[i].displayFinalMarks();
        }
    }
}
```

LAB: 7

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 cases both father and son's age and throws an exception if son's age is >=father's age.

```
public class WrongAge extends Exception{
   WrongAge(String msg){
      super(msg);
   }
}
import java.util.Scanner;
class InputScanner{
   Scanner sc;
   InputScanner(){
      sc=new Scanner(System.in);
   }
}
class Father extends InputScanner{
   int fatherAge;
   Father() throws WrongAge{
      System.out.println("Enter father's age:");
      fatherAge=sc.nextInt();
         if(fatherAge<0){
         throw new WrongAge("Age cannot be negative");
      }

   }
   void display(){
      System.out.println("Father's age:"+fatherAge);
   }
}
class Son extends Father{
   int sonAge;
   Son() throws WrongAge{
      System.out.println("Enter Son's age:");
      sonAge=sc.nextInt();
      if(sonAge>= fatherAge){
         throw new WrongAge("Son's age cannot be greater than father's age");
      }
      else if(sonAge<0){
         throw new WrongAge("Age cannot be negative");
      }

   }

   void display(){
      super.display();
      System.out.println("Son's age:"+sonAge);
   }
```

```
        try{
            Son son=new Son();
            son.display();
        }
        catch(WrongAge e){
            System.out.println(e);

        }
    }
}
```

LAB: 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 two seconds.

```
class BMS_College_of_Engineering implements Runnable {
    public void run() {
        while (true) {
            try {
                System.out.println("BMS College of Engineering");
                Thread.sleep(10000); // Sleep for 10000 milliseconds (10 seconds)
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}

class CSE implements Runnable {
    public void run() {
        while (true) {
            try {
                System.out.println("CSE");
                Thread.sleep(2000); // Sleep for 2000 milliseconds (2 seconds)
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}

public class ThreadMain {
    public static void main(String[] args) {
        Thread t1 = new Thread(new BMS_College_of_Engineering());
        Thread t2 = new Thread(new CSE());
        t1.start();
        t2.start();
    }
}
```

LAB: 9

Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException. If Num2 were Zero, the program would throw an Arithmetic Exception Display the exception in a message dialog box.

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

class SwingDemo{
   SwingDemo(){
      // create jframe container
      JFrame jfrm = new JFrame("Divider App");
      jfrm.setSize(275, 150);
      jfrm.setLayout(new FlowLayout());
      // to terminate on close
      jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      // text label
      JLabel jlab = new JLabel("Enter the divider and divident:");

      // add text field for both numbers
      JTextField ajtf = new JTextField(8);
      JTextField bjtf = new JTextField(8);

      // calc button
      JButton button = new JButton("Calculate");

      // labels
      JLabel err = new JLabel();
      JLabel alab = new JLabel();
      JLabel blab = new JLabel();
      JLabel anslab = new JLabel();

      // add in order :)
      jfrm.add(err);  // to display error bois
      jfrm.add(jlab);
      jfrm.add(ajtf);
      jfrm.add(bjtf);
      jfrm.add(button);
      jfrm.add(alab);
      jfrm.add(blab);
      jfrm.add(anslab);

      ActionListener l = new ActionListener() {
         public void actionPerformed(ActionEvent evt) {
            System.out.println("Action event from a text field");
         }
      };
      ajtf.addActionListener(l);
```

```
        button.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                try{
                    int a = Integer.parseInt(ajtf.getText());
                    int b = Integer.parseInt(bjtf.getText());
                    int ans = a/b;

                    alab.setText("\nA = " + a);
                    blab.setText("\nB = " + b);
                    anslab.setText("\nAns = "+ ans);
                }
                catch(NumberFormatException e){
                    alab.setText("");
                    blab.setText("");
                    anslab.setText("");
                    err.setText("Enter Only Integers!");
                }
                catch(ArithmeticException e){
                    alab.setText("");
                    blab.setText("");
                    anslab.setText("");
                    err.setText("B should be NON zero!");
                }
            }
        });

        // display frame
        jfrm.setVisible(true);
    }

    public static void main(String args[]){
        // create frame on event dispatching thread
        SwingUtilities.invokeLater(new Runnable(){
            public void run(){
                new SwingDemo();
            }
        });
    }
}
```

LAB: 10

Demonstrate Inter process Communication and deadlock.

```
class Q {
	int n;
	boolean valueSet = false;

	synchronized int get() {
		while(!valueSet)
			try {
				System.out.println("\nConsumer waiting\n");
				wait();
			} catch(InterruptedException e) {
				System.out.println("InterruptedException caught");
			}
		System.out.println("Got: " + n);
		valueSet = false;
		System.out.println("\nIntimate Producer\n");
		notify();
		return n;
	}

	synchronized void put(int n) {
		while(valueSet)
			try {
				System.out.println("\nProducer waiting\n");
				wait();
			} catch(InterruptedException e) {
				System.out.println("InterruptedException caught");
			}
		this.n = n;
		valueSet = true;
		System.out.println("Put: " + n);
		System.out.println("\nIntimate Consumer\n");
		notify();
	}

}

class Producer implements Runnable {
	Q q;
	Producer(Q q) {
		this.q = q;
		new Thread(this, "Producer").start();
	}
	public void run() {
		int i = 0;
		while(i<5) {
			q.put(i++);
		}
	}
}
```

```
class Consumer implements Runnable {
	Q q;
	Consumer(Q q) {
		this.q = q;
		new Thread(this, "Consumer").start();
	}

	public void run() {
		int i=0;
		while(i<5) {
			int r=q.get();
			System.out.println("consumed:"+r);
			i++;
		}
	}

}

class PCFixed {
	public static void main(String args[]) {
		Q q = new Q();
		new Producer(q);
		new Consumer(q);
		System.out.println("Press Control-C to stop.");
	}
}
```

Deadlock:

```
class A {
	synchronized void foo(B b) {
		String name = Thread.currentThread().getName();
		System.out.println(name + " entered A.foo");
		try {
			Thread.sleep(1000);
		} catch(Exception e) {
			System.out.println("A Interrupted");
		}
		System.out.println(name + " trying to call B.last()");
		b.last();
	}

	void last() {
		System.out.println("Inside A.last");
	}
}

class B {
	synchronized void bar(A a) {
		String name =Thread.currentThread().getName();
		System.out.println(name + " entered B.bar");
		try {
```

```
                Thread.sleep(1000);
            } catch(Exception e) {
                System.out.println("B Interrupted");
            }
            System.out.println(name + " trying to call A.last()");
            a.last();
        }
        void last() {
            System.out.println("Inside A.last");
        }
}

class Deadlock implements Runnable
{
        A a = new A();
        B b = new B();
        Deadlock() {
            Thread.currentThread().setName("MainThread");
            Thread t = new Thread(this,"RacingThread");
            t.start();
            a.foo(b); // get lock on a in this thread.
            System.out.println("Back in main thread");

        }
        public void run() {
            b.bar(a); // get lock on b in other thread.
            System.out.println("Back in other thread");
        }

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

}
```