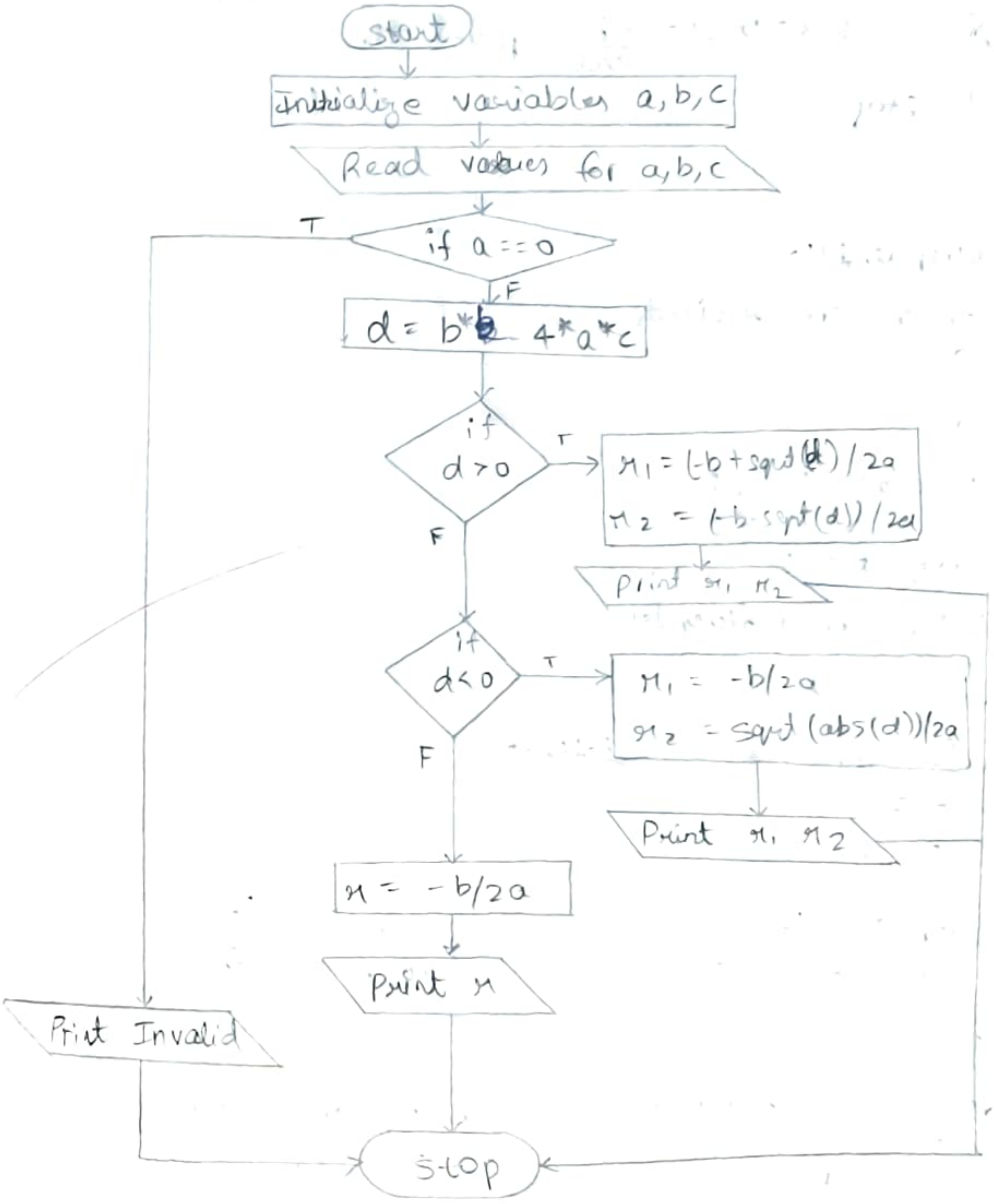
```
import java.util.scanner;
import java.lang.Math.sqrt;
import java.lang.Math.abs;

public class Quad{
  public static void main(String[] args){
      Scanner in = new Scanner(system.in);
      system.out.println("Enter co-efficients :");
      int a = in.nextInt();
      int b = in.nextInt();
      int c = in.nextInt();

    if (a==0){
       system.out.println("Invalid Input");
    }
    else {
       int d = b*b - 4*a*c;
       if (d>0){
         system.out.println("Roots are real");
         float r1 = (float)(-b + sqrt(d))/(2*a);
         float r2 = (float)(-b - sqrt(d))/(2*a);
         system. println("r1 = "+ r1 +"\n"+ "r2 = "+ r2);
       }
       else if (d<0){
         system.out.println("Roots are imaginary");
         float r1 = (float) -b/(2*a);
         float r2 = (float) sqrt(abs(d))/(2*a);
       system.out.println(r1 = "+ r1+"\n"+"r1 = "+r2);
       }
else {
    system.out.println ("Roots are equal.");
    float r = (float) -b/2*a;
    system. out. println ("r=" + r);
 }

}
}
```

Flowchart:



Algorithm:-

```
1. Start
2. Declare variables a, b, c, d
3. Read values for a, b and c.
4. if a = 0 print invalid, goto step 9
5. else $d = b^2 - 4ac$
6. if d > 0, $r_1 = \frac{-b + \sqrt{d}}{2a}$, $r_2 = \frac{-b - \sqrt{d}}{2a}$, print $r_1, r_2$
7. else if d < 0, $r_1 = \frac{-b}{2a}$, $r_2 = \left(\frac{\sqrt{abs(d)}}{2a}\right) i$, print $r_1, r_2$
8. else if d = 0 $r = \frac{-b}{2a}$ print r.
9. stop
```

Output 1):-

```
Enter co-efficients
0
4
9
Invalid Input
```

output 2):-

```
Enter co-efficients
1
-5
-6
roots are ~~equal~~ distinct
r1 = 6.0
r2 = -1.0
```

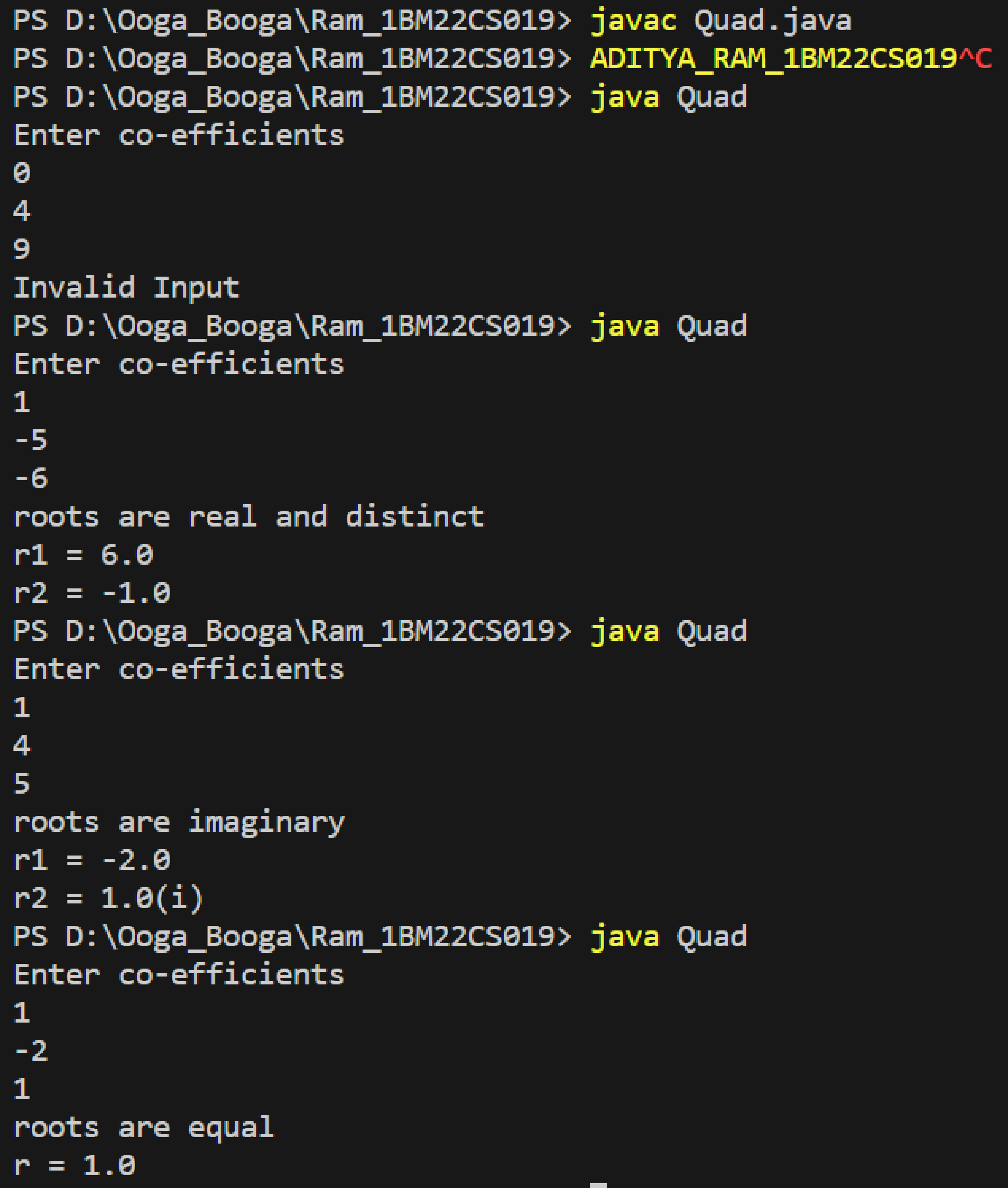
[illegible] 02/12

output 3):-

```
Enter co-efficients
1
4
5
roots are imaginary
r1 = -2
r2 = 1(i)
```

output 4):-

```
Enter co-efficients
1
-2
1
roots are equal
r = 1.0
```



```
PS D:\Ooga_Booga\Ram_1BM22CS019> javac Quad.java
PS D:\Ooga_Booga\Ram_1BM22CS019> ADITYA_RAM_1BM22CS019^C
PS D:\Ooga_Booga\Ram_1BM22CS019> java Quad
Enter co-efficients
0
4
9
Invalid Input
PS D:\Ooga_Booga\Ram_1BM22CS019> java Quad
Enter co-efficients
1
-5
-6
roots are real and distinct
r1 = 6.0
r2 = -1.0
PS D:\Ooga_Booga\Ram_1BM22CS019> java Quad
Enter co-efficients
1
4
5
roots are imaginary
r1 = -2.0
r2 = 1.0(i)
PS D:\Ooga_Booga\Ram_1BM22CS019> java Quad
Enter co-efficients
1
-2
1
roots are equal
r = 1.0
```

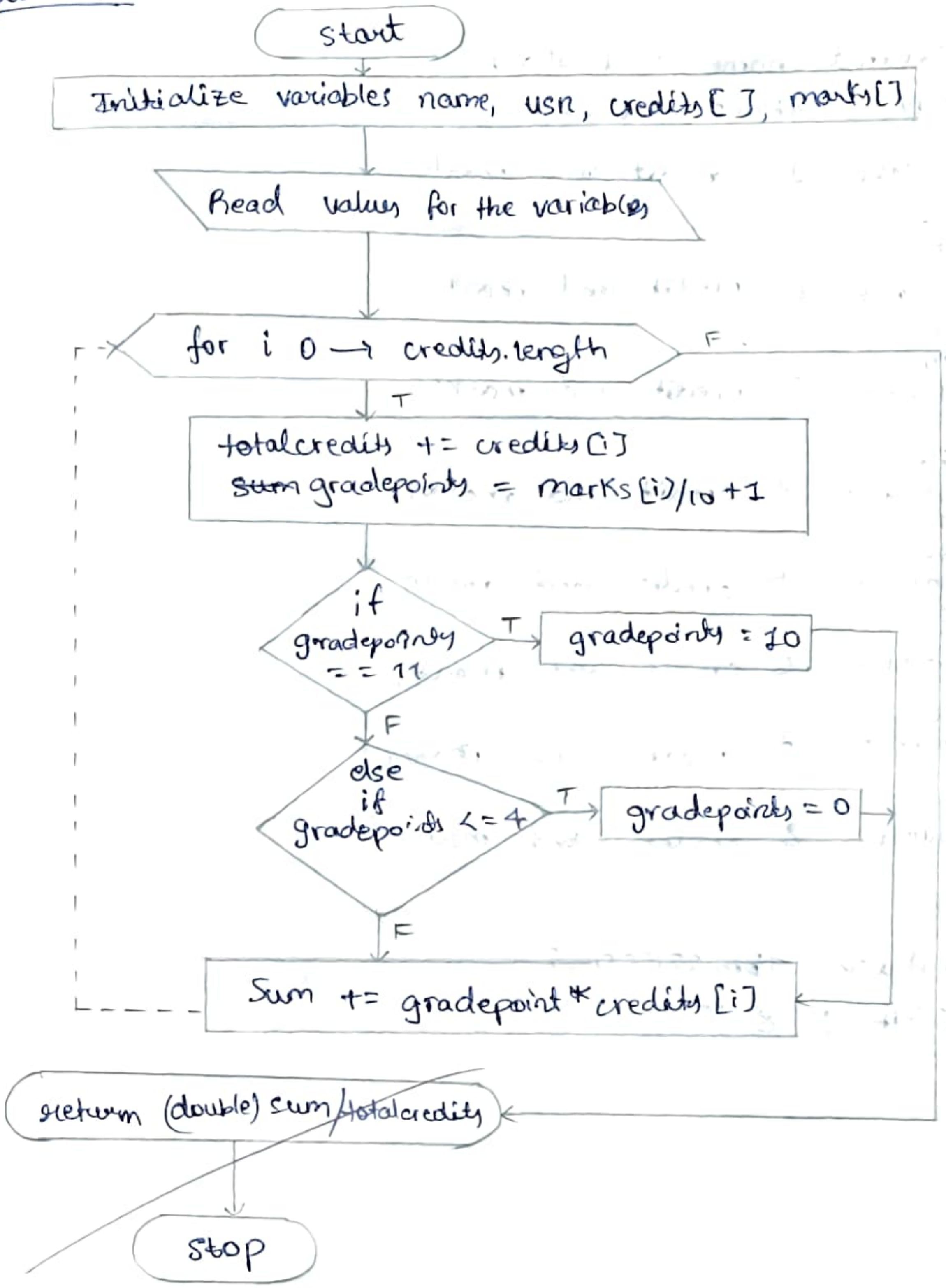
Lab-2

```
import java.util.*
class Student {

    String usn;
    String name;
    int[] credits = new int[8];
    int[] marks = new int[8];

    public void details() {
        Scanner reader = new Scanner(System.in);
        System.out.print("Student name : ");
         name = reader.nextLine();
        system.out.print("USN : ");
         usn = reader.nextLine();

        for(int i=0 ; i< credits.length ; i++)
        {                                        "credit and marks"
          system.out.print("~~Subject~~ + i+1 + ": ");
                            course"
          ~~for(~~ credits[i] = reader.nextInt();
                   marks[i] = reader.nextInt();
          }
    }
           double
    public ~~void~~ sgpa() {
        int totalcredits = 0 , gradepoint;
        int ~~weightedSum~~ = 0;
                 Sum
        for(int i=0; i< credits.length ; i++) {
              totalcredits += credits[i];
              gradepoint = marks[i]/10 + 1;
              if (gradepoint == 11) {
                   gradepoint = 10;
              }
              else if( gradepoint <= 4){
                   gradepoint = 0;
              }
              sum += gradepoint * credits[i];
        }
        return (double) sum / (double) total credits;
    }
}
~~class Sgpa~~
Public class Sgpa {
        public static void main (String[] args) {
            Student student = new Student ();
             student. details ();
            System. out . println (student.name + "\t" + Student.usn);
            system.out.println(" SGPA : " + student.Sgpa());
        }
}
```

Algorithm:-

```
1. Start
2. define class student with attributes name, usn, credits
   array and marks array
3. Read details for the student.
4. ~~Calcute~~ for i: 0 → credits.length
5. ~~total~~credits += credits [i]
   ~~weighted sum +=~~
6. grade points = marks [i]/10 + 1
7. If gradepoints == 11 set it to 10
8. else if gradepoints <= 4 set it to 0.
9. sum += gradepoints * credits [i]
10. return sum/total credits
11. End
```

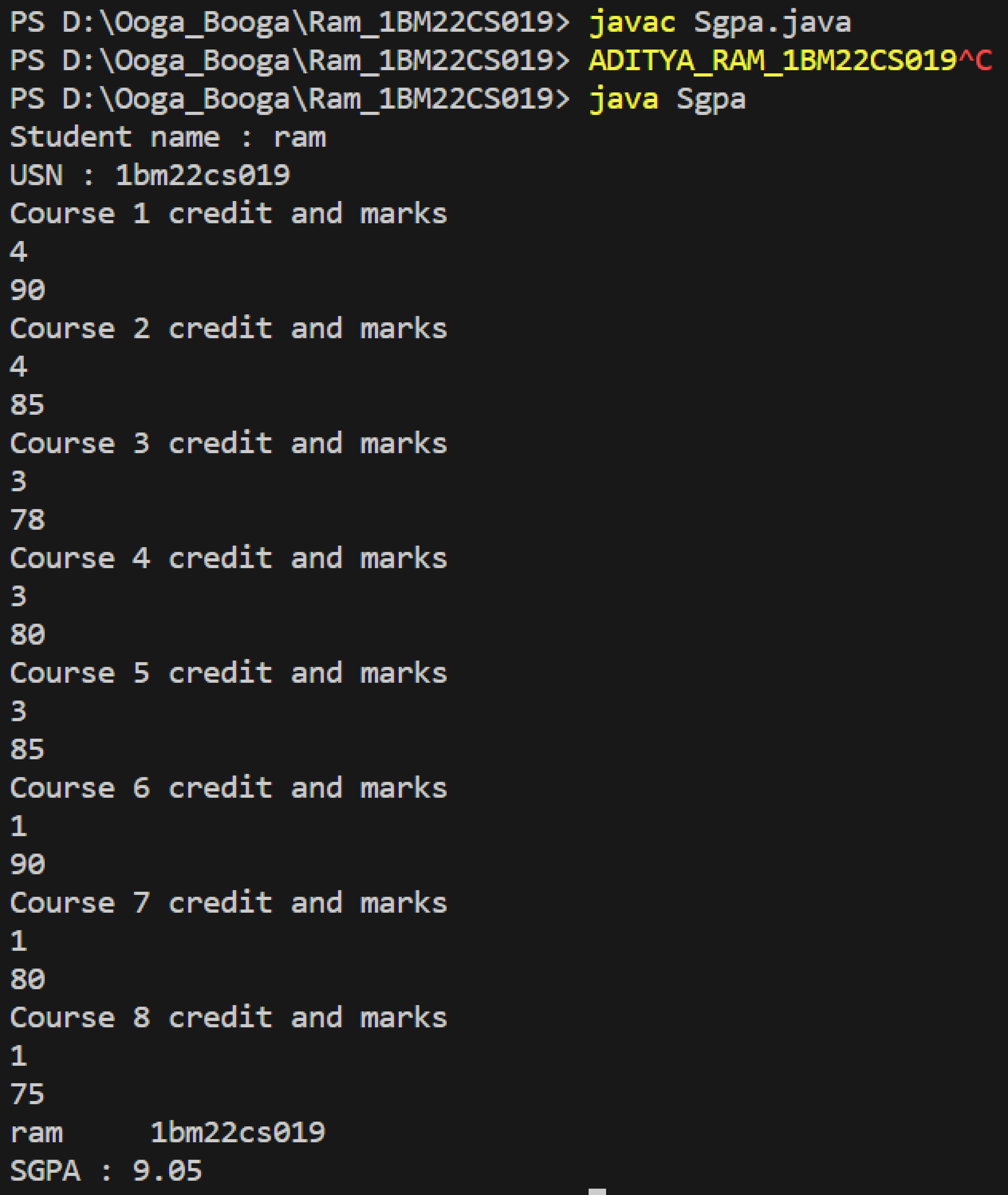
Flow Chart :-


start
Initialize variables name, usn, credits[ ], marks[ ]
Read values for the variables
for i 0 → credits.length
F
T
totalcredits += credits[i]
~~sum~~ gradepoints = marks[i]/10 + 1
if gradepoints == 11
T
gradepoints = 10
F
else if gradepoints <= 4
T
gradepoints = 0
F
Sum += gradepoint * credits[i]
return (double) sum/totalcredits
stop

Output:-

```
Student name : Prakash
USN : 1bm22cs055
Course 1 credit and marks
4
85
course 2 credit and marks
4
90
course 3 credit and marks
3
60
course 4 credit and marks
3
82
course 5 credit and marks
3
91
course 6 credit and marks
1
80
course 7 credit and marks
1
85
course 8 credit and marks
1
60
Prakash 1bm 22cs055
SGPA : 8.95
```

[illegible] 29/12/23


```
PS D:\Ooga_Booga\Ram_1BM22CS019> javac Sgpa.java
PS D:\Ooga_Booga\Ram_1BM22CS019> ADITYA_RAM_1BM22CS019^C
PS D:\Ooga_Booga\Ram_1BM22CS019> java Sgpa
Student name : ram
USN : 1bm22cs019
Course 1 credit and marks
4
90
Course 2 credit and marks
4
85
Course 3 credit and marks
3
78
Course 4 credit and marks
3
80
Course 5 credit and marks
3
85
Course 6 credit and marks
1
90
Course 7 credit and marks
1
80
Course 8 credit and marks
1
75
ram     1bm22cs019
SGPA : 9.05
```

Program 03 :

Create a class Book which contain 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. Include a toString() method.

```
class Book {
    private String name;
    private String author;
    private double price;
    private int numPages;

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

    public void setname (String name){
        this.name = name;
    }
    public void setauthor (string author){
        this.author = author;
    }
    public void setprice (double price){
        this.price = price;
    }
    public void setnumPages (int numPages){
        this.numPages = numPages;
    }
```

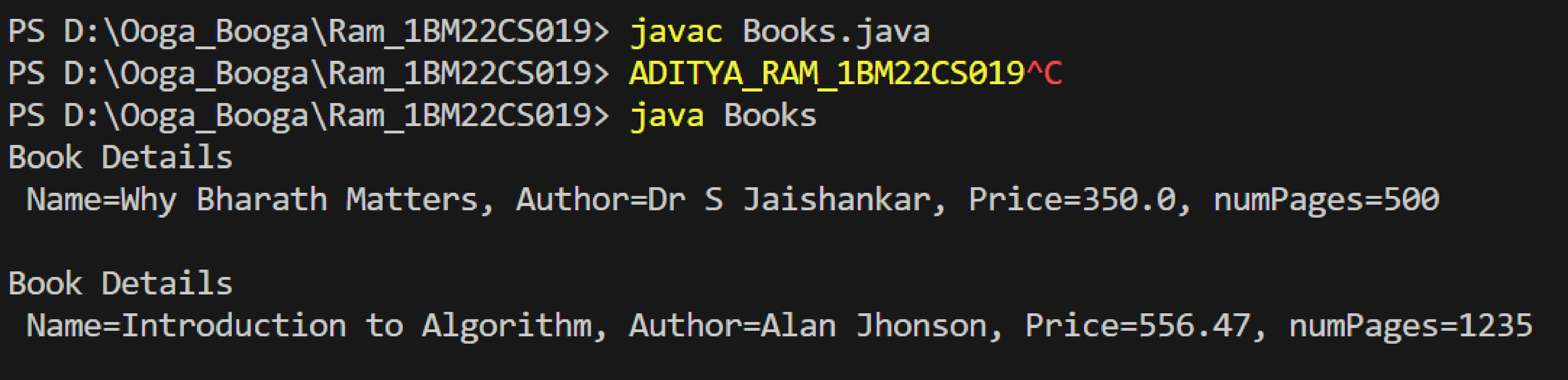
```
public ~~void~~ String getname(){
        return ~~this.~~ name;
}
public string getauthor(){
        return author;
}
public double getprice() {
        return price;
}
public int getnumPages() {
        return numPages;

}

public string toString() {
    return "Book details :\n" ~~+~~ Name : " + Name +"\n
        Author : " + author "\nPrice : " + price
        " ~~+~~\n Pages : " + numPages ;
}

public class Example {
    public static void main (String[] args) {
        int n = 2;
        Book [] books = new Book[n];
        books[0] = new Book ("Why Bharat Matters",
                             "Dr S Jaishankar",
                             350.00,
                             500);
        books[1] = new Book ("Introduction to Algorithm",
                             "Alan Thonson",
                             556.47,
                             1235);
~~System.out.println(~~
        for (Book b : books){
                system.out.println (b.toString());
        }
    }
}
```

Algorithm:-

1) start
2) create class book
3) make private members name, author, price, num_Page,
4) make parameterized ~~con~~ constructor
   Book(String name, String author, ~~float~~ double price, int numPages)
5) create set ~~methods~~ and get methods for all the members of the class
6) override toString() method and print all the details of the book.
7) Stop.

output!-

```
Book details;
 Name: why Bharath Matters
Author: Dr. S Jaishankar
 Price : 350.00
Pages :  500

Book details:
Name: Introduction to Algorithms
Author: Alan Thonson
Price: 556.47
Pages: 1235
```



```
PS D:\Ooga_Booga\Ram_1BM22CS019> javac Books.java
PS D:\Ooga_Booga\Ram_1BM22CS019> ADITYA_RAM_1BM22CS019^C
PS D:\Ooga_Booga\Ram_1BM22CS019> java Books
Book Details
 Name=Why Bharath Matters, Author=Dr S Jaishankar, Price=350.0, numPages=500

Book Details
 Name=Introduction to Algorithm, Author=Alan Jhonson, Price=556.47, numPages=1235
```

Program 4 :-

```
abstract class shape {
double ~~int~~ x;
double ~~int~~ y;
      abstract void print_area();
}

class Rectangle extends shape {
      Rectangle (~~int~~ double x, ~~int~~ double y){
         this. x = x
         this. y = y
      }
   public void print_area(){
         system.out.println ("rectangle_area =" + x*y);
   }
}
class Triangle extends shape {
      Triangle (double x, double y) {
            this. x = x ;
            this. y = y ;
      }
      public void print_area () {
          system.out. println ("Triangle_area" + x*y*0.5));
      }

}
```

```
class circle extends shape{
        circle (double x) {
            this.x = x;
        }
        public void print-area(){
            system.out.println ("Cricle area =" + 3.14*x*x);
        }
}

public class Example{
    public static void main(String[] args) {
        Rectangle r = new Rectangle (5, 3);
        Triangle  t = new Triangle (10, 5);
        Circle    c = new Circle (7);

        r. print-area();
        t. print_area();
        c. print-area();
    }
}
```

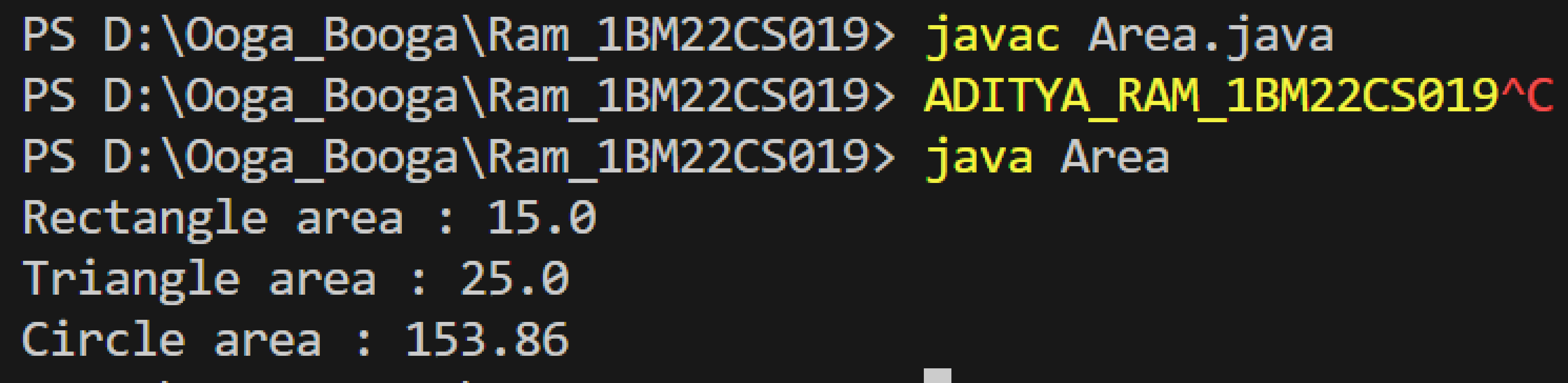
Output:-

```
rectangle_area = 15
triangle-area = 25
Circle_area = 153.86
```

Algorithm:-

1) start
2) create an abstract class shape with two members x and y
3) create an abstract method print_area()
4) create 3 classes rectangle, triangle, circle that extends shape class
5) override print-area in all the classes.
6) in rectangle print $x*y$
7) in triangle print $x*y*0.5$
8) in circle print $3.14*x*x$
9) stop.

[illegible] 12/01/24



```
PS D:\Ooga_Booga\Ram_1BM22CS019> javac Area.java
PS D:\Ooga_Booga\Ram_1BM22CS019> ADITYA_RAM_1BM22CS019^C
PS D:\Ooga_Booga\Ram_1BM22CS019> java Area
Rectangle area : 15.0
Triangle area : 25.0
Circle area : 153.86
```

19/01/2024

```
class Account {

  public String CustomerName;
  public int accountNumber;
  public String accountType;
  public double balance;

  public Account (String customerName, int accountNumber
                  String accountType , double balance).
  {
     this. customerName = customerName;
     this. accoent Nueber = accountNumber;
     this. accountType = account Tyre;
     this. balance = balance;
  }

  public void deposit (double amount)
  {    if (amout > 0) {
               Balance += amount;
      }   System.out.println ("Deposit of" + amount+" Succesful").
      }
      else {
         system.out.println ("Invalid amount. Deposit Failed").
      }
   }
  public void displayBalance () {
     System.out.println (" The balance of account number"
                        + accountNumber +" is" + balance);
  }
```

```
public void CompoundInterest() { }
public void withdraw(double amount)
{   if(amount > 0 && amount <= balance)
      {   Balance -= amount;
          System.out.println("withdrawal of" + amount + "succesful");
          System.out.println("New balance :" + balance);
      }
    else {
      System.out.println("Invalid amount");
    }
}
public void checkBalance() {}
{
}

Class sav_acct extends Account
{
   public double InterestRate;
   public double interest;

   public sav_acct(String customerName, int accountNumber
                  double balance, double interestRate)
   {
       Super(CustomerName, accountNumber, "savings", balan)
       this.interestRate = interestRate;
       this.interest = 0;
   }
@override
public void computeInterest() {
      ~~Interest = Balance * interestRate/100;~~
      ~~Deposit(Interest);~~
}     Interest = balance * (1 + (0.05/12)) - balance;
      Deposit(Interest);
```

```
class  Cur-acct  extends  Account
{
   public  double  minBalance ;
   public  double  service Charge ;

   public  Cur-acct (String  customerName,  int  accountNumber
                        double  balance,  double  servicecharge)
   {
       Super (customerName,  accountNumber, "current", balance);
       this. minbalance  =  minbalance ;
       this. servicecharge  =  servicecharge ;
   }
@override
public  voidcheckbalance() {
       if (balance < minbalance) {
           balance  =  balance - servicecharge ;
           System.out.println ("Balance ~~Below~~ Low");
       }
}


public  class  BankApp {
    public  static  void  main(String[] args) {
        Sav_acct  sav = new  Sav_acct ("Alice", 1001, 5000, 5);
        sav. deposit (1000);
        sav. withdraw (2000);
        sav. computeInterest();
        sav. displayBalance();
    }
}
```

```
        cur_acct cur = new Cur_acct ("Bob", 1002, 10000,
                                              2000, 200);
        cur. deposit (5000);
        cur. withdraw(8000);
        cur. checkBalance();
        cur. display Balance();
    }
}
```

Algorithm:-

1 Define a class Bankapp with a main method

2 Define two classes Sav_acct and cur_act that represent savings and current accounts respectively.

3. In each class, declare the fields for customer name, account number, balance and other relevant details.

4. In each class, define a constructor that takes the parameters for the fields and assigns them to the object

5. In each class, define methods for depositing, withdrawing computing interest, checking balance and displaying balance

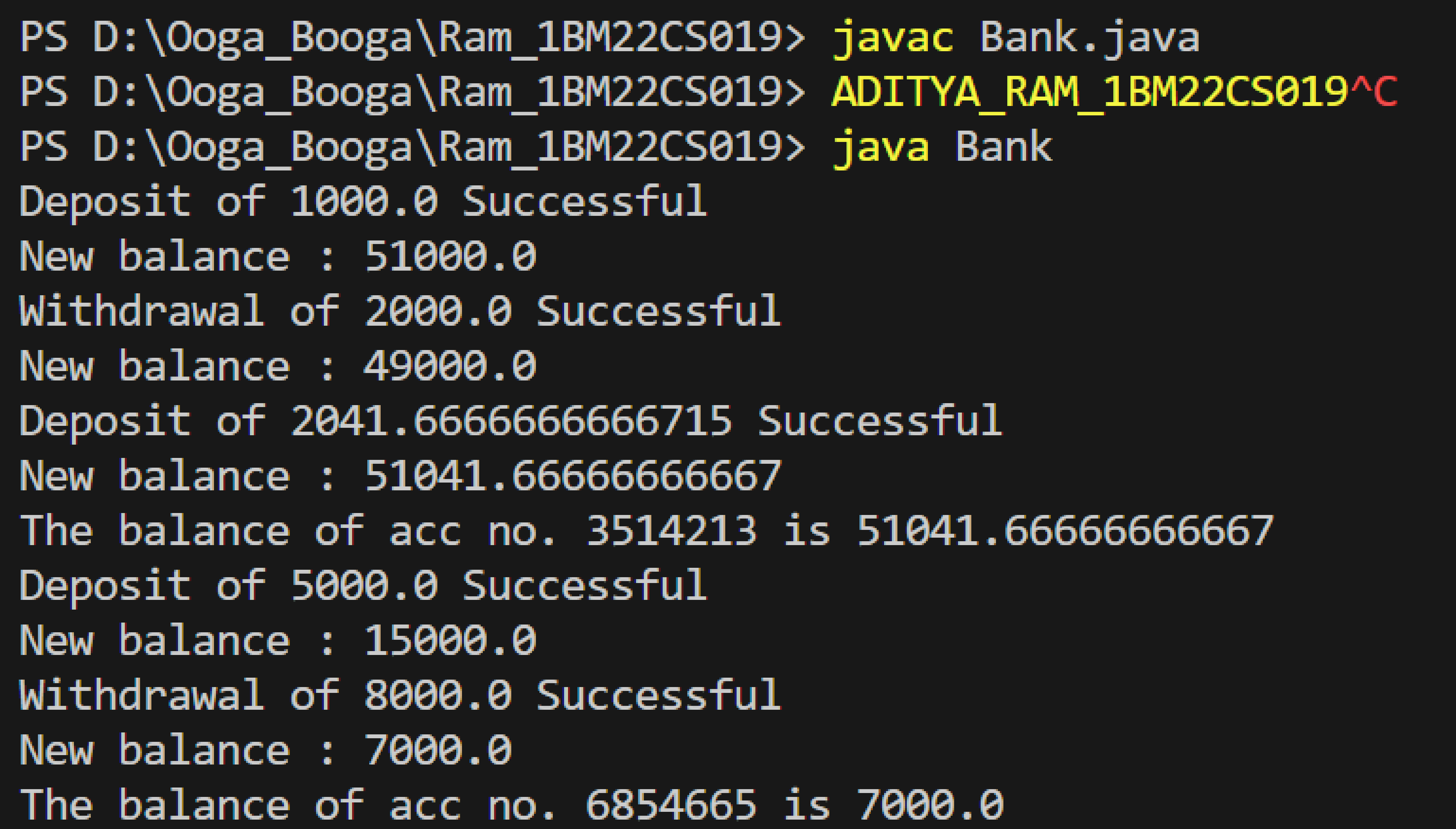
6. In main method, create a sav_acct object and a cur_acct object with some initial values.

7. Display the updated ~~bla~~ balances of the objects.

Output :-

```
Deposit of 1000.0 successful
New balance : 51000.0
withdrawal of 2000.0 successful
New balance : 49000.0

Deposit of 2041.667 successful.
New balance : 51041.667
The balance of acc no. 3514213 is 5104.667
Deposit of 5000.0 successful
New balance : 15000.0
withdrawal of 8000.0 successful
New balance : 7000.0
The balance of acc no. 6854665 is 7000.0
```

[illegible]
19/01/24



```
PS D:\Ooga_Booga\Ram_1BM22CS019> javac Bank.java
PS D:\Ooga_Booga\Ram_1BM22CS019> ADITYA_RAM_1BM22CS019^C
PS D:\Ooga_Booga\Ram_1BM22CS019> java Bank
Deposit of 1000.0 Successful
New balance : 51000.0
Withdrawal of 2000.0 Successful
New balance : 49000.0
Deposit of 2041.6666666666715 Successful
New balance : 51041.66666666667
The balance of acc no. 3514213 is 51041.66666666667
Deposit of 5000.0 Successful
New balance : 15000.0
Withdrawal of 8000.0 Successful
New balance : 7000.0
The balance of acc no. 6854665 is 7000.0
```

02/02/2024

Program - 6

```
package CIE;

public class Student {
public ~~protected~~ String usn, name;
public ~~protected~~ int sem;
}

public class Internals extends Student {
    ~~protected~~ int[] marks = new int[5];
}   public
```

```
package SEE;
import CIE.Student;

public class Externals extends Student {
    ~~protected~~ int[] marks = new int[5];
}   public
```

```
import CIE.Internals
import SEE.Externals

public class FinalMarks {
    public static void main(String[] args) {
        int n = 5;
        Internals[] I = new Internals[n];
        Externals[] E = new Externals[n];
for(int i=0; i<n ; i++) { I[i]= new Internals();
        I[i].usn     = "USN" + i;
        I[i].name    = "Student" + i;
        I[i]. sem    = 3;
        I[i]. marks  = new int[]{80, 75, 90, 85, 88};
        E[i] = new Externals();
        E[i].usn     = "USN" + i;
        E[i].name    = "Student" + i;
        E[i]. sem    = 3;
        E[i]. marks  = new int[]{75, 70, 88, 92, 85};
}

for (int i=0; i<n; i++) {
    int[] finalMarks = new int[5];
        for(int j=0 ; j<5; j++){
            finalMarks[j] = (I[i].marks[j] + E[i].marks[j])/2;
        }

    System.out.println("Final Marks for " + I[i].name + ":" +
Arrays.toString(finalMarks));

 }
}
}
```

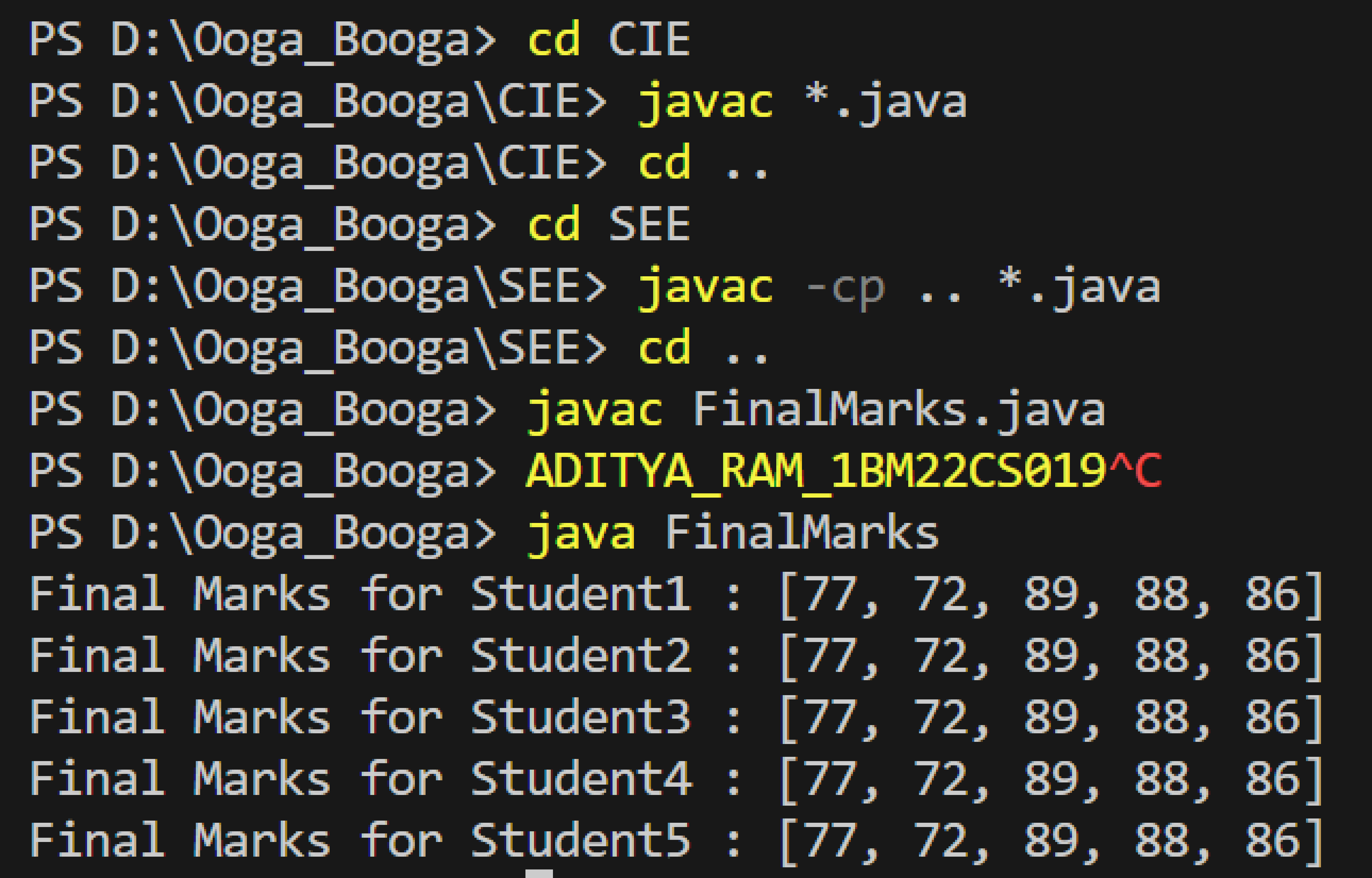
Algorithm

1. Start.
2. Create a package CIE with two files Student.java and Internals.java
3. create another package SEE with one file Externals.java
4. In another package create a ~~f~~ finalmarks.java file and import the CIE and SEE packages
5. Enter the student details and create a function to add the internals and external marks.
6. Student class will contain USN, name and semester
7. Internals class ~~[illegible]~~ extends the Student class and contains an array of 5 called marks
8. ~~External~~ In SEE package Externals.java import CIE package and Externals class extends student class.
9. External class contains an array of 5 called marks.
10. In finalmarks.java ~~add~~ ~~internals a~~ calculate the final marks = (internals + externals)/2 while importing the packages CIE and SEE.
11. Stop

output :-

```
Final marks for student 1 : [77.5, 72.5, 89, 88.5, 86.5]
Final marks for Student 2 : [77.5, 72.5, 89, 88.5, 86.5]
Final marks for Student 3 : [77.5, 72.5, 89, 88.5, 86.5]
Final marks for Student 4 : [77.5, 72.5, 89, 88.5, 86.5]
Final marks for Student 5 : [77.5, 72.5, 89, 88.5, 86.5]
```

[illegible] 02.02.24



```
PS D:\Ooga_Booga> cd CIE
PS D:\Ooga_Booga\CIE> javac *.java
PS D:\Ooga_Booga\CIE> cd ..
PS D:\Ooga_Booga> cd SEE
PS D:\Ooga_Booga\SEE> javac -cp .. *.java
PS D:\Ooga_Booga\SEE> cd ..
PS D:\Ooga_Booga> javac FinalMarks.java
PS D:\Ooga_Booga> ADITYA_RAM_1BM22CS019^C
PS D:\Ooga_Booga> java FinalMarks
Final Marks for Student1 : [77, 72, 89, 88, 86]
Final Marks for Student2 : [77, 72, 89, 88, 86]
Final Marks for Student3 : [77, 72, 89, 88, 86]
Final Marks for Student4 : [77, 72, 89, 88, 86]
Final Marks for Student5 : [77, 72, 89, 88, 86]
```

16/02/2024

Program - 7

```
import java.Util.Scanner;
class ~~wrongAge~~extends Exception {
        public wrongAge(String message){
                super(message);
        }
}

class  Father { public int FatherAge;
      Father(int FatherAge) throws wrongAge{
          ~~try{~~
            if (FatherAge < 0) {
                throw new wrongAge("Father's age is -ve");
            }
            this.FatherAge = FatherAge;
      }
}

class  Son extends Father{
  public int SonAge;
                          int
      son(int sonAge, FatherAge) throws wrongAge{
          ~~try~~  super(FatherAge);
            if (SonAge <0) {
                throw new wrongAge("Son's age is -ve");
            }
            if (SonAge >= FatherAge) {
                throw new wrongAge("Son's should be less
                                        than Father");
            }
            this.SonAge = SonAge;
  };
}
```

```
public class Main {
    public static void main (String[] args){
        Scanner s = new Scanner(System.in);
        int f-age, s-age;
        try {
            system.out.print("Father's age :");
            f-age = s.nextInt();
            system.out.print("Son's age :");
            s-age = s.nextInt();
        }catch (Exception e){
            system.out.println(e);
        }

        try {
            Son s = new Son(s-age, f-age);
        }
        catch (WrongAge wa) {
            system.out.println(wa);
        }
    }
}
```

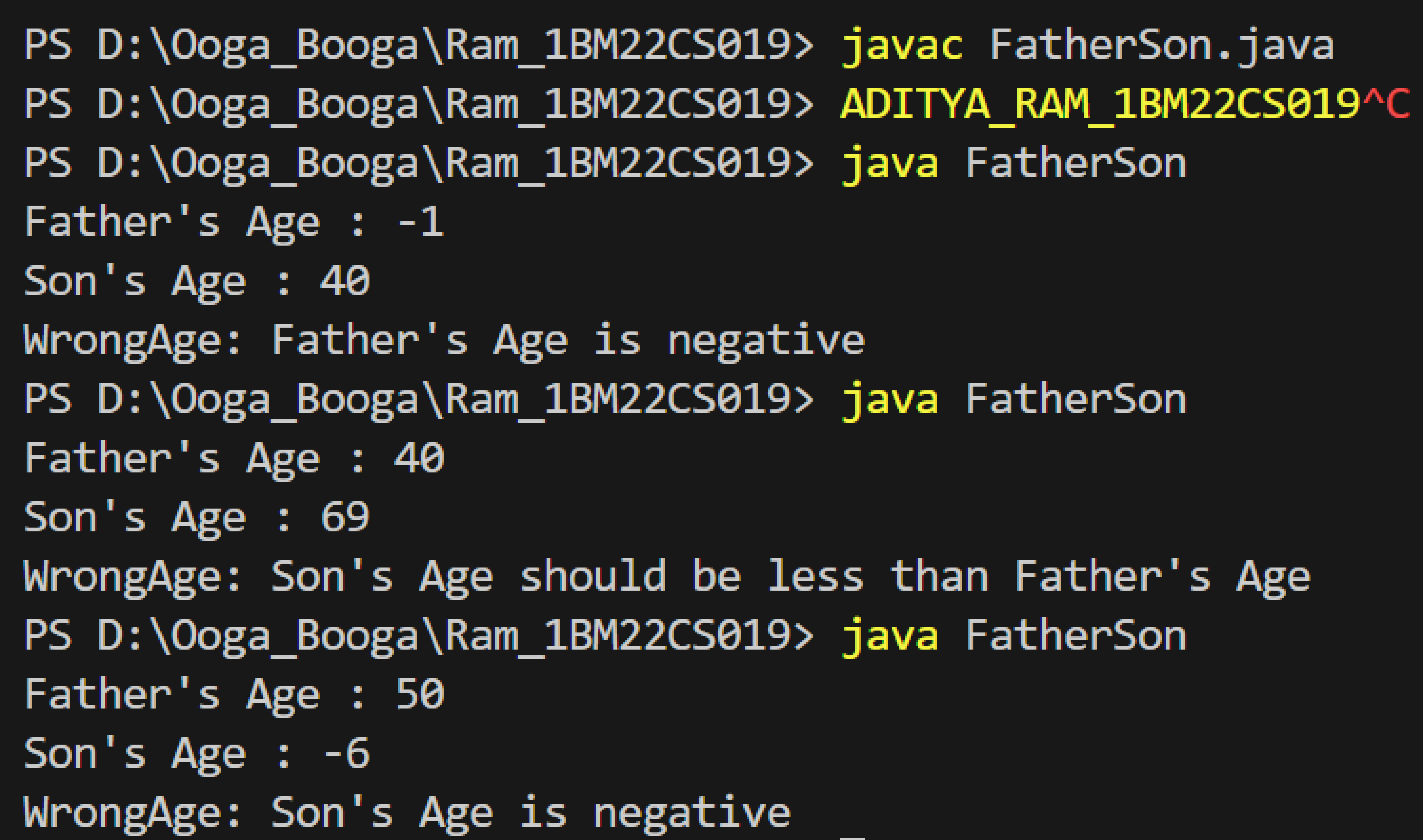
[illegible] 16/2

Algorithm :-

1. start
2. create a class wrongAge which extends exception class
3. create a class Father, Father constructor takes Father Age and if Father age is less than zero throws wrong age exception
4. create a son class, that expands the father class and throws wrongAge exception if son's age is ~~less than~~ more then or equal to Father's age or son's age is less than zero.
5. Input a son's and father's ages and handle the exception in main method
6. Stop.

output :-

```
Father's Age : -1
Son's Age : 40
wrongAge : Father's Age is negative.

Father's Age : 40
Son's Age : 65
wrongAge : son's Age should be less than Father's Age.

Father's Age : 50
Son's Age : -6
wrongAge : Son's Age is negative.
```



```
PS D:\Ooga_Booga\Ram_1BM22CS019> javac FatherSon.java
PS D:\Ooga_Booga\Ram_1BM22CS019> ADITYA_RAM_1BM22CS019^C
PS D:\Ooga_Booga\Ram_1BM22CS019> java FatherSon
Father's Age : -1
Son's Age : 40
WrongAge: Father's Age is negative
PS D:\Ooga_Booga\Ram_1BM22CS019> java FatherSon
Father's Age : 40
Son's Age : 69
WrongAge: Son's Age should be less than Father's Age
PS D:\Ooga_Booga\Ram_1BM22CS019> java FatherSon
Father's Age : 50
Son's Age : -6
WrongAge: Son's Age is negative
```

16/02/2024

Program - 8

```
class A extends Threads {
    public void run() {
   try{ for(int i=0; i<5 ;i++){
            System.out.println("BMS college of Engineering");
            Thread.sleep(10000);
        }
    }
   catch(InterruptedException ie){ }
  }
}

class B extends Threads {
  public void run() {
    try {
      for(int i=0; i<5; i++){
        system.out.println("CSE");
         Thread.sleep(2000);
      }
    }catch(InterruptedException ie) { }
  }
}
public class Main{
   public static void main(String[] args) {
         A a = new A();
         B b = new B();
         a.start();
         b.start();
   }
}
```

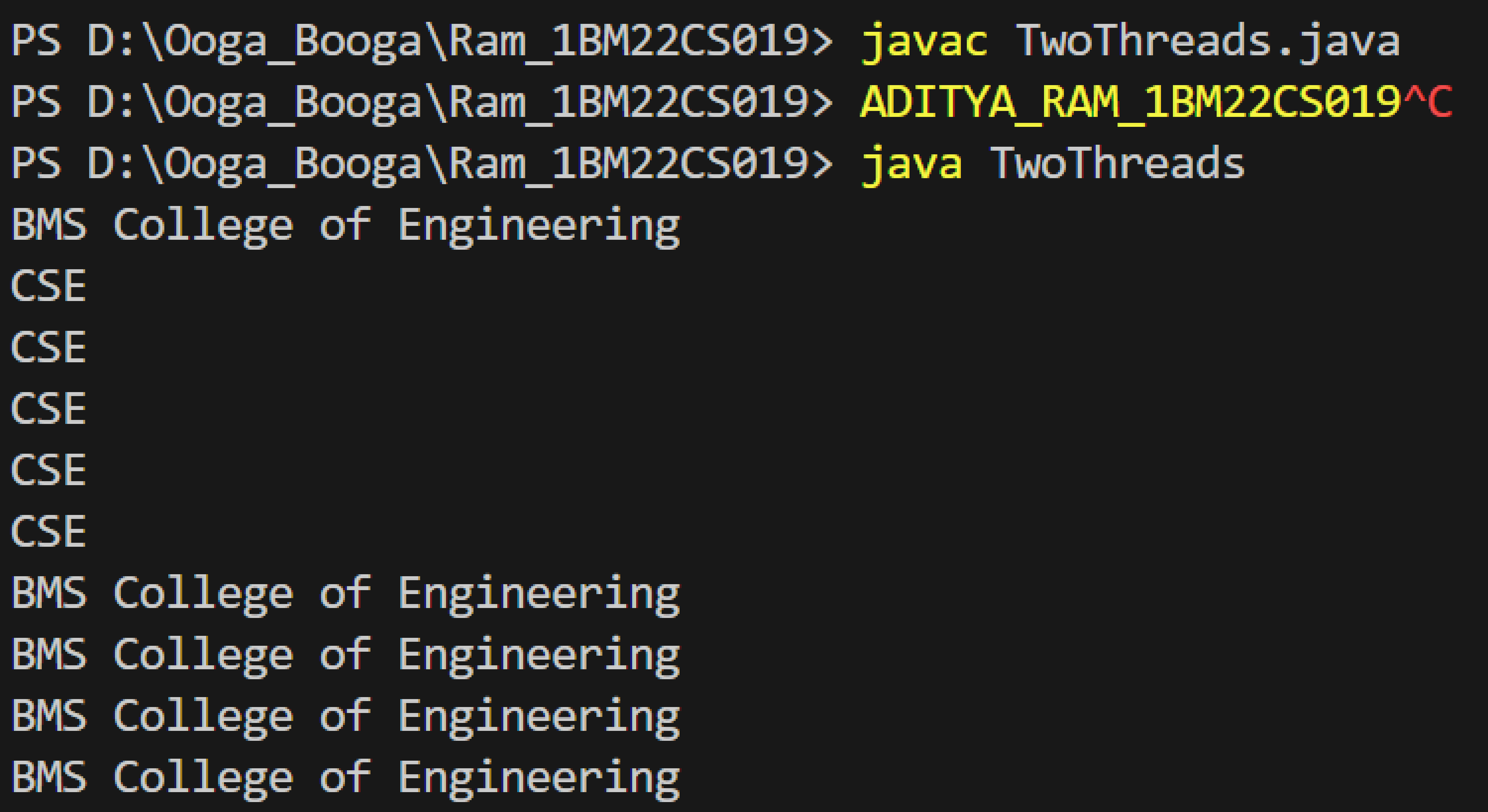
Algorithm :-

1. Start
2. create two classes ~~as~~ which extends thread class.
3. on both classes override run method.
4. First class prints "BMS college of Engineering" ~~[illegible]~~ and the thread is put on sleep of 10 seconds.
5. Second class prints "CSE" and the therad is put on sleep for 2 seconds.
6. Stop.

Output :-

```
BMS college of Engineering
CSE
CSE
CSE
CSE
CSE
BMS college of Engineering
BMS college of Engineering
BMS college of Engineering
BMS college of Engineering
```

[illegible] 16/2



```
PS D:\Ooga_Booga\Ram_1BM22CS019> javac TwoThreads.java
PS D:\Ooga_Booga\Ram_1BM22CS019> ADITYA_RAM_1BM22CS019^C
PS D:\Ooga_Booga\Ram_1BM22CS019> java TwoThreads
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
BMS College of Engineering
BMS College of Engineering
BMS College of Engineering
```

23/02/2024

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

public class LastLab{
    public LastLab(){
        JFrame jfrm = new JFrame("Divider App");
        jfrm.setSize(300,300);
        jfrm.setLayout(new FlowLayout());
        jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE)

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

       JTextField ajtf  = new JTextField(10);
       JTextField bjtf  = new JTextField(10);

       JButton button = new JButton("Calculate");

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

      jfrm.add(err);
      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);
bjtf. addActionListener (l);

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

        alab. setText ("\nA = " +a);
        blab. setText ("\nB = " + b);
        anslab. setText("\nAns = " + ans);
      }catch (NumberFormatException e){
          alab. setText(" ");
          blab. setText(" ");
          anslab. setText (" ");
          err. setText(~~"\nAns = " + ans~~ "Integer only\n");
      } catch(ArithmeticException e) {
          alab. setText(" ");
          blab. setText(" ");
          anslab. setText(" ");
          err. setText("B should be NON zero(\n");
      }
    }
});
jfrm. setVisible(true);
}
```

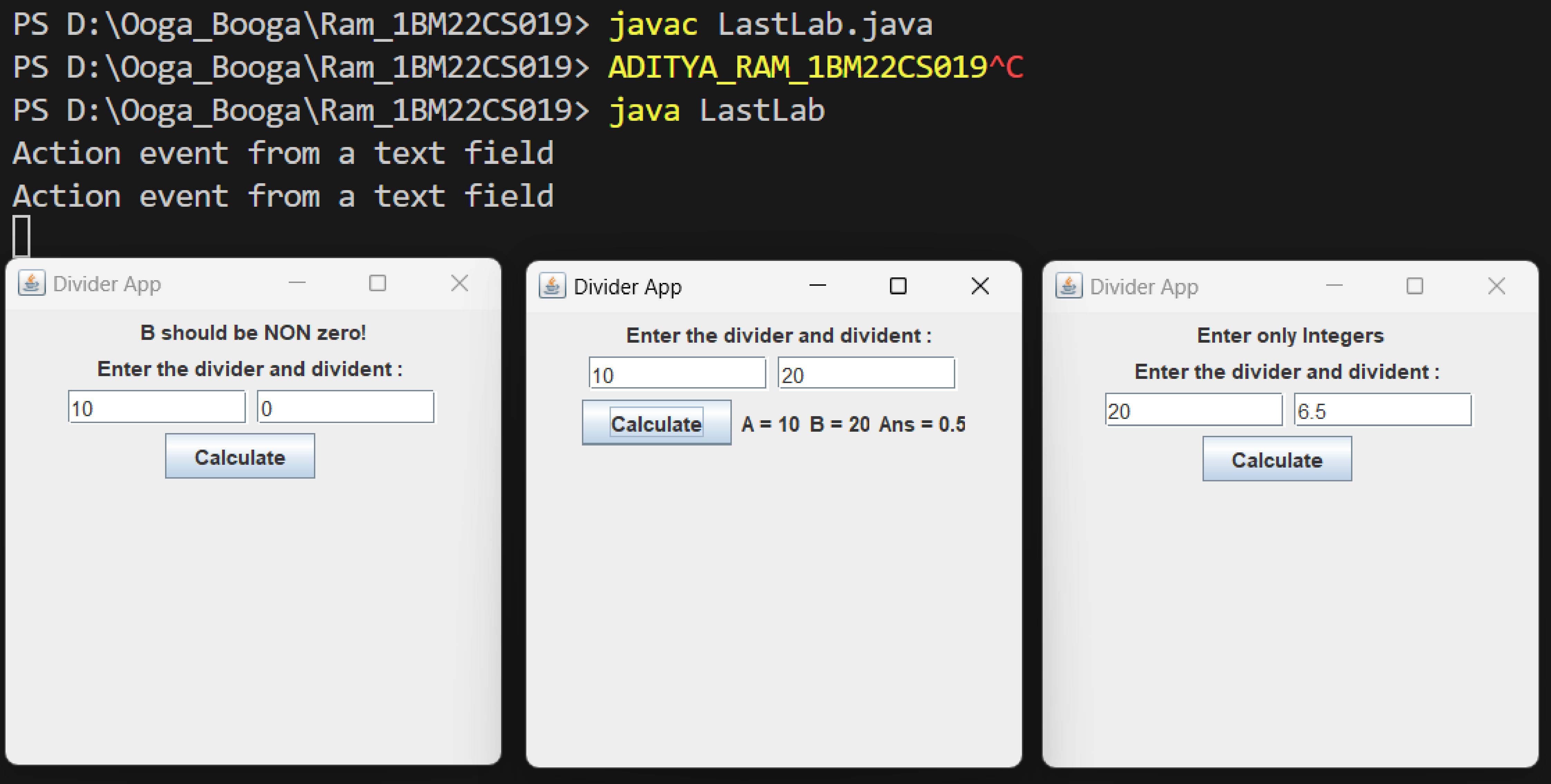
```
public static void main (String [] args) {
        SwingUtilities.invokeLater (new Runnable () {
                public void run () {
                        new LastLab();
                }
        });
    }
}
```

Output:-

```
Enter the divider and divident

[ 10 ]          [ 3 ]

[ calculate ]     A = 10     B = 3

ANS = 3.333


Enter the divider and divident

[ 10 ]          [ 0 ]

[ calculate ]

B should ~~as~~ be NON ZERO
```

* The constructor Lastlab() initializes the GUI components and sets up the layout of the frame.
* It creates a JFrame (a window) with a specified title, size, layout, and default close operation.
* Labels JLabel, TextFields JTextField and a Button JButton are added to the frame to input the numbers and display the result
* Action Listeners are attached to the text fields and the button to handle interactions.
* The actionperformed method is overridden to respond to events like clicking the button or pressing Enter.
* when the button is clicked, it calculates the division and handles the errors like division by zero and number format error.
* setText It is used to set the text inside a label or other screen element
* ~~set~~getText() :
  It returns the value from the single line text field

[illegible]
23.02.24



```
PS D:\Ooga_Booga\Ram_1BM22CS019> javac LastLab.java
PS D:\Ooga_Booga\Ram_1BM22CS019> ADITYA_RAM_1BM22CS019^C
PS D:\Ooga_Booga\Ram_1BM22CS019> java LastLab
Action event from a text field
Action event from a text field
```

Divider App

B should be NON zero!

Enter the divider and divident :

10 | 0

Calculate

Divider App

Enter the divider and divident :

10 | 20

Calculate   A = 10 B = 20 Ans = 0.5

Divider App

Enter only Integers

Enter the divider and divident :

20 | 6.5

Calculate