OOPS LAB RECORD

Name: Manika Rajpal

Sap Id: 500084172

Roll No.: R214220669

Batch 3

* Experiment 1
  + Lab Exercise

1. Java Versions
2. JDK and JRE

JRE: The Java runtime environment is a collection of software tools for creating Java applications. It's the NM implementation. It is a real thing. It comprises a collection of libraries as well as other files that JVM uses during execution.

JDK: The Java Development Kit (JDK) is a software development environment that is used to create Java applications.

Java application development It is a real thing. It includes the Java Runtime Environment (JRE) as well as developer tools.

1. Setting Path

For setting the permanent path of JDK,follow these steps:

Go to MyComputer properties -> advanced tab -> environment variables -> new tab of user variable -> write path in variable name -> write path of bin folder in variable value -> ok -> ok -> ok

1. Code Editors

Source-code editors have features specifically designed to simplify and speed up typing of source code, such as syntax highlighting, indentation, autocomplete and brace matching functionality. These editors also provide a convenient way to run a compiler, interpreter, debugger, or other program relevant for the software-development process.

A source-code editor can check syntax while code is being entered and immediately warn of syntax problems.

A few source-code editors compress source code, typically converting common keywords into single-byte stokens, removing unnecessary whitespace, and converting numbers to a binary form.

Some of the best code editors for java are Eclipse, Kite, IntelliJ Idea, Blue J, Apache NetBeans.

1) Eclipse :Eclipse is an IDE for writing and compiling Java programmes.

It highlights the syntax you have written.

This tool enables you to easily debug the programme.

2) Kite

Kite is IDE for Java that automatically completes multiple line codes.

This editor supports more than 16 languages.

It helps you to code faster with no hassle.

3) IntelliJ IDEA

IntelliJ IDEA is a multi-purpose IDE which focuses on Java development.

It \sis one of the best Java compiler that offers advanced support for the web, mobile, and also hybrid application development.

1. Sample Hello World Program.

* Code

class Hello{

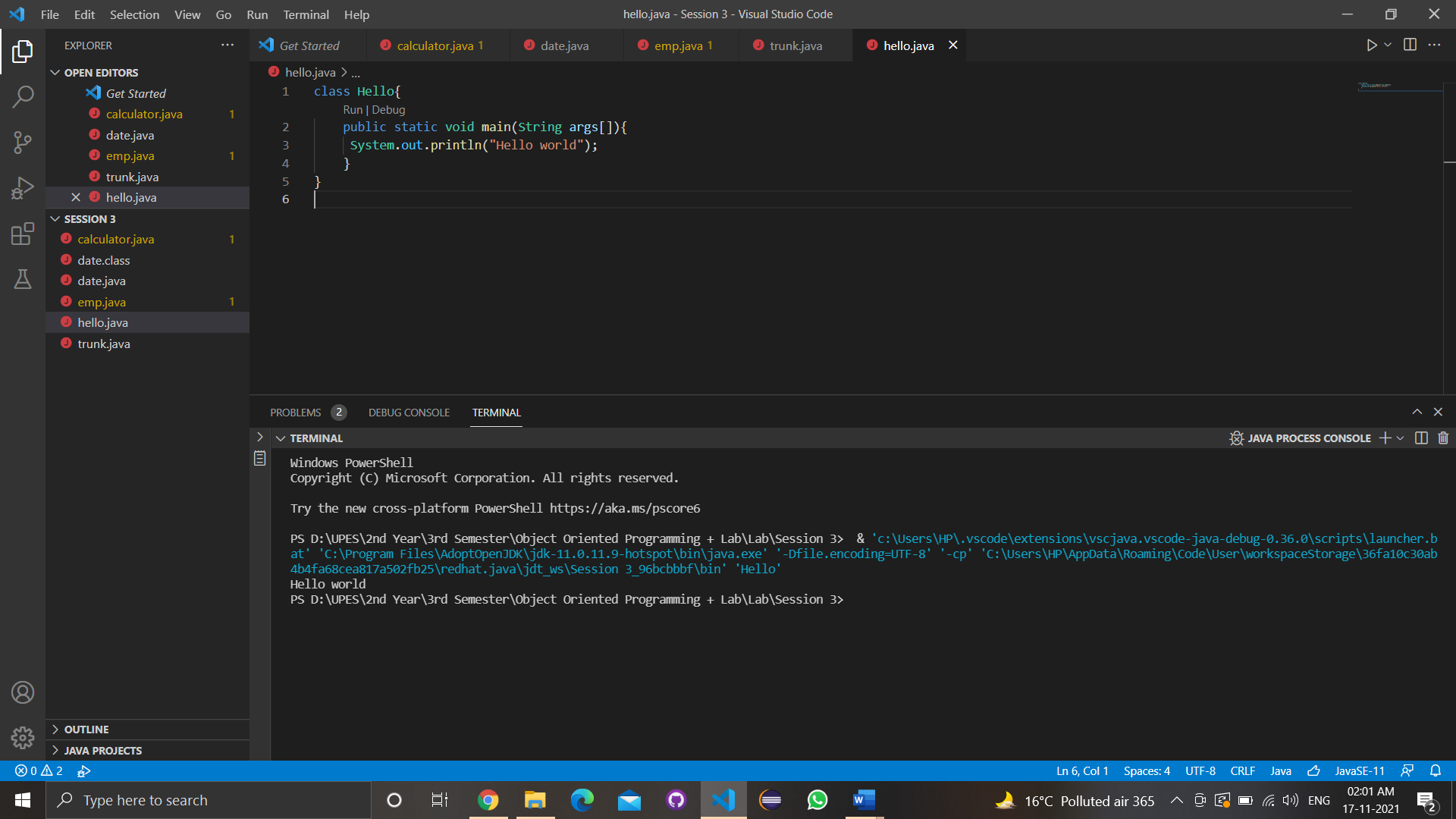
    public static void main(String args[]){

     System.out.println("Hello world");

    }

}

* Output



---------------------------------------------------------------------------------------------------------

* Experiment 2
  + Lab Exercise

1. To find the largest of three numbers.

* Code

import java.util.Scanner;

public class Biggest

{

    public static void main(String[] args)

    {

        int a, b, c, largest, temp;

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter the first number:");

        a = sc.nextInt();

        System.out.println("Enter the second number:");

        b = sc.nextInt();

        System.out.println("Enter the third number:");

        c = sc.nextInt();

        temp=a>b?a:b;

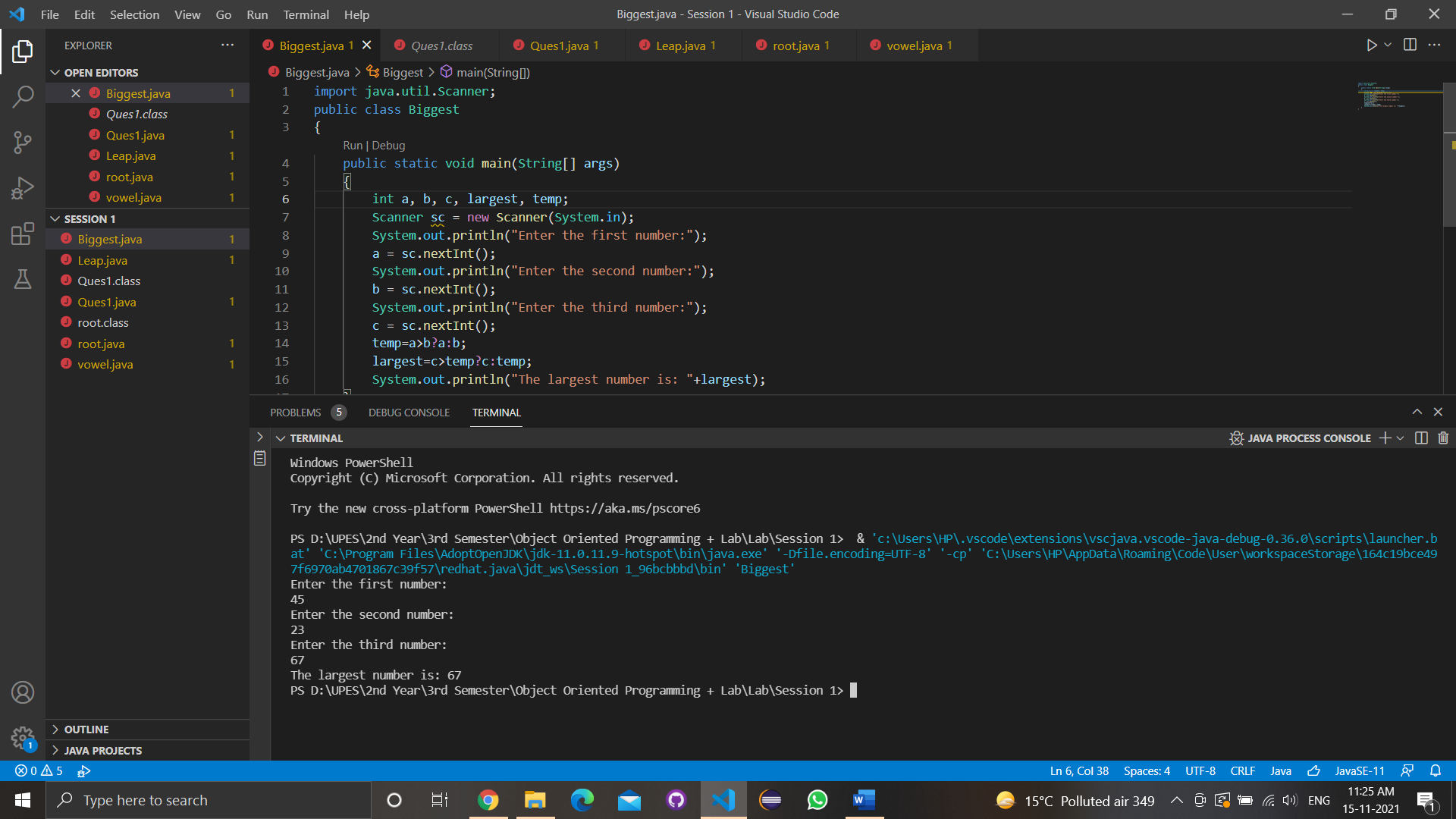
        largest=c>temp?c:temp;

        System.out.println("The largest number is: "+largest);

    }

}

* Output



1. Write a program to implement a command line calculator. (Try for Add sub Mul Division in same program for 2 digits.) Integer.parseInt will be used For e.g. java calc 20 + 30 Output should be Sum of 20 and 30 is 50 java calc 50 \* 30 Output should be Product of 50 and 30 is 1500 .

* Code

import java.util.Scanner;

public class calculator

{

    public static void main(String[] args)

    {

        int num1, num2, c,ans;

        Scanner scl = new Scanner(System.in);

        System.out.println("Enter the first number:");

        num1 = scl.nextInt();

        System.out.println("Enter the second number:");

        num2 = scl.nextInt();

        System.out.println("1. ADD \n 2. SUBTRACT \n 3.MULTIPLY \n 4. DIVIDE");

        c=scl.nextInt();

        switch(c){

        case 1:

        {

            ans=num1+num2;

            System.out.println("Sum of "+num1+" and "+num2+" is "+ans);

            break;

        }

        case 2:

        {

            ans=num1-num2;

            System.out.println("difference of "+num1+" and "+num2+" is "+ans);

            break;

        }case 3:

        {

            ans=num1\*num2;

            System.out.println("multiplication of "+num1+" and "+num2+" is "+ans);

            break;

        }case 4:

        {

            ans=num1%num2;

            System.out.println("division of "+num1+" and "+num2+" is "+ans);

            break;

        }

        default:

        System.out.println("ENTERED WRONG CHOICE!!!!");

    }

}

}

* Output



1. Write a program to accept 10 student’s marks in an array, arrange it into ascending order, convert into the following grades and print marks and grades in the tabular form. Between 40 and 50 : PASS Between 51 and 75 : MERIT and above : DISTINCTION.

* Code

import java.util.Scanner;

public class marks

{

    public static void main(String args[])

    {

        int size;

        Scanner sc=new Scanner(System.in);

        System.out.print("Enter marks of 10 students: ");

        int[] array = new int[10];

        size=array.length;

        for(int i=1; i<10; i++)

        {

            array[i] = sc.nextInt();

        }

        for (int i = 0 ;i< size-1; i++)

        {

            int min = i;

            for (int j = i+1; j<size; j++)

            {

                if (array[j] < array[min])

                {

                    min = j;

                }

            }

            int temp = array[min];

            array[min] = array[i];

            array[i] = temp;

        }

        for (int i = 0 ;i< size; i++)

        {

            if(array[i]>55)

            {

                System.out.print(" "+array[i]+ " "+"Merit\n");

            }

            else if(array[i]<55 && array[i]>41)

            {

                System.out.print(" "+array[i]+ " "+"Pass\n");

            }

            else

            {

                System.out.print(" "+array[i]+ " "+"Fail\n");

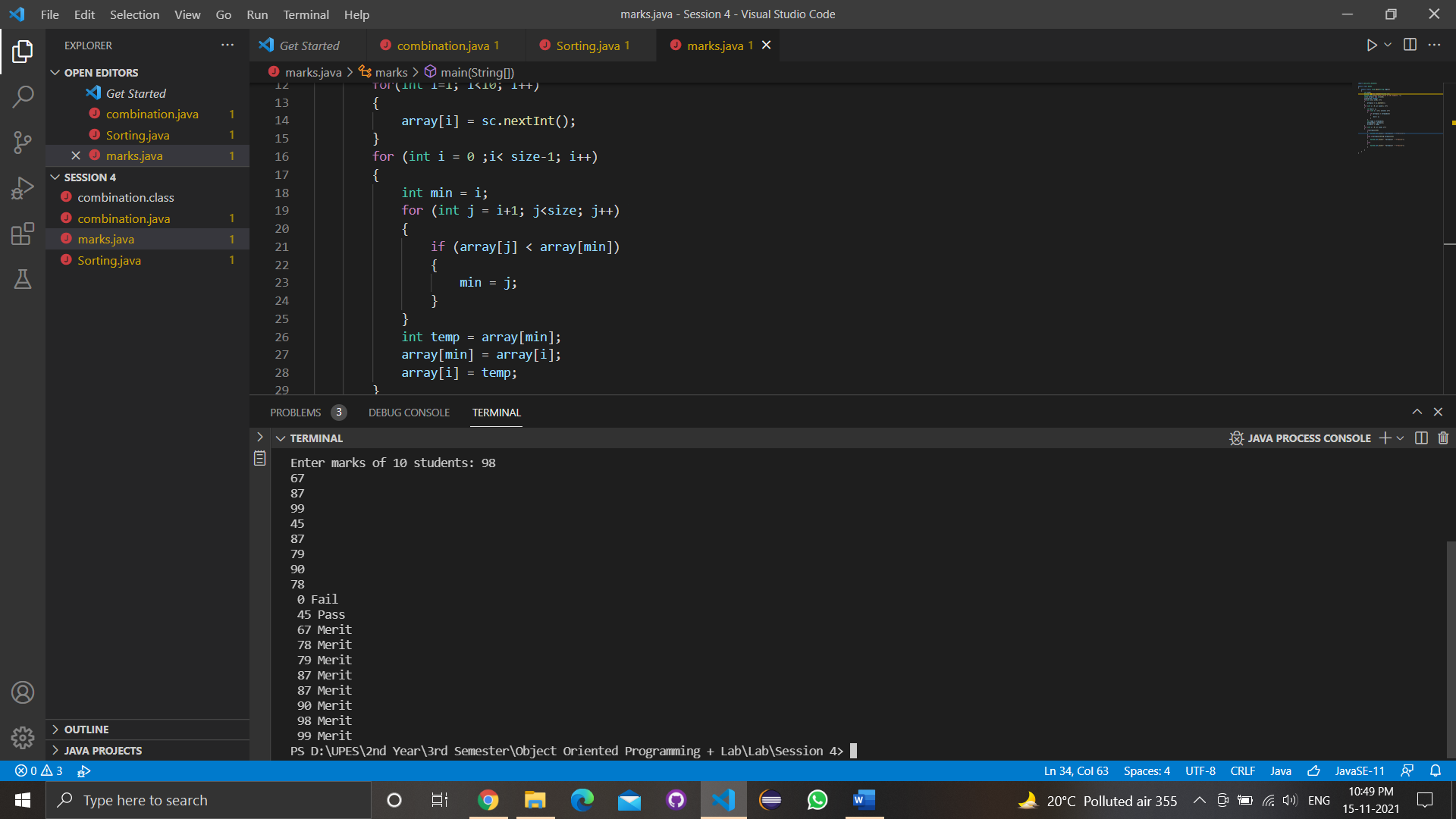
            }

        }

    }

}

* Output



4.WAP to Take input as DD MM YYYY(04 08 2021) in command line and calculate number of days since 1 January 1970.

* Code

public class date

{

 public static void main(String args[])

 {

 int d = Integer.parseInt(args[0]);

 int m = Integer.parseInt(args[1]);

 int y = Integer.parseInt(args[2]);

 int monthDays[] = {31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334,

365};

 int dyear = y - 1970;

 int x, num = 0;

 for(x = 1970; x<=y; x++)

 {

 if(x%4 == 0 && x%400 == 0 && x%100 != 0){

 num++;

 }

 }

 int days;

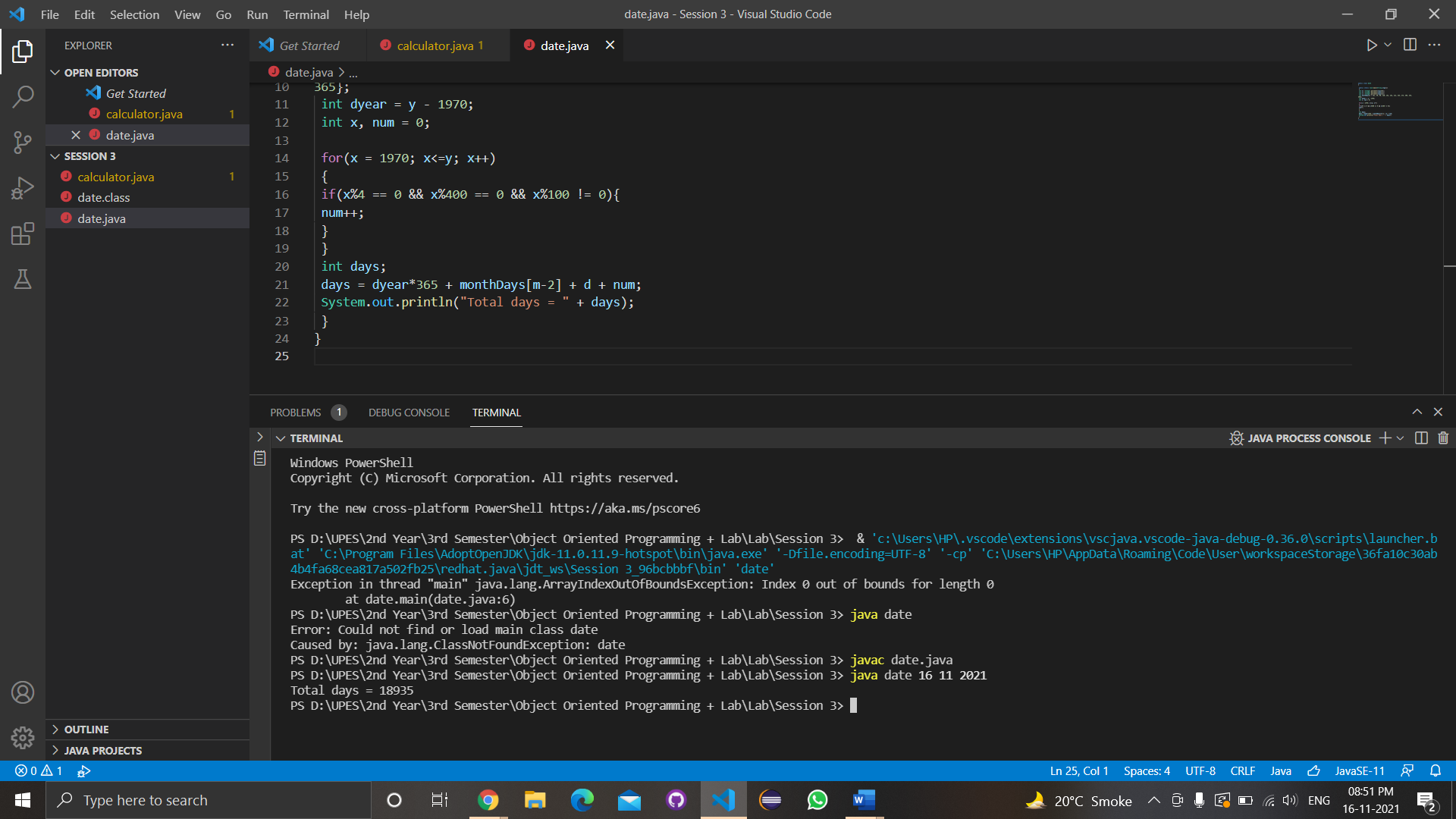
 days = dyear\*365 + monthDays[m-2] + d + num;

 System.out.println("Total days = " + days);

 }

}

* Output



5. WAP to print the following pattern using loops

\*

\*\*\*

\*\*\*\*\*

* Code

import java.util.Scanner;

public class pattern

{

    public static void main(String args[])

    {

        int i, j, row=6;

        for(i=0; i<row; i=i+2)

        {

            for(j=0; j<=i; j++)

            {

            System.out.print("\* ");

            }

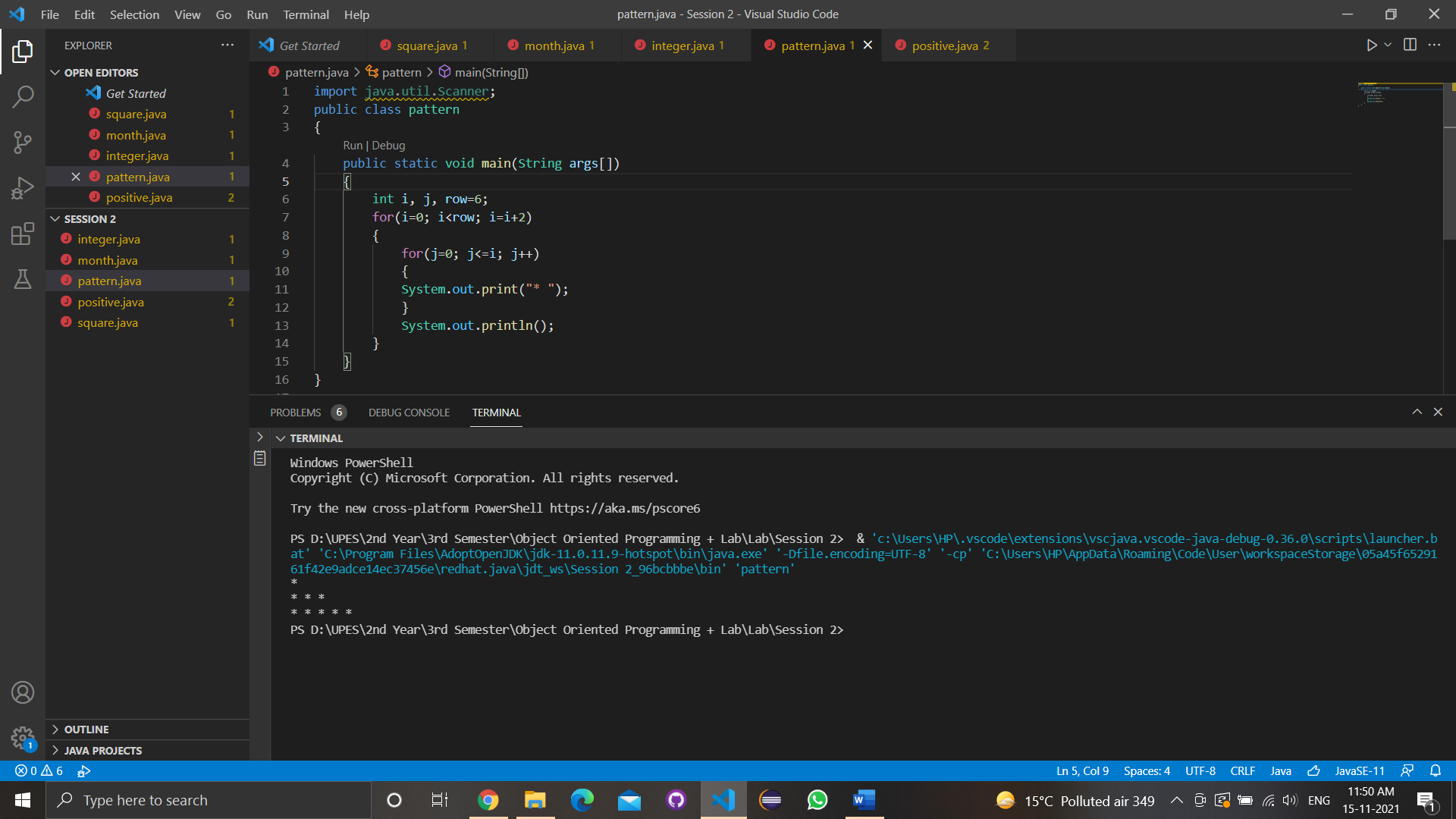
            System.out.println();

        }

    }

}

* Output

  
------------------------------------------------------------------------------------------------

* Experiment 3
  + Lab Exercise

1. Write a program to accept three digits (i.e., 0 - 9) and print all its possible combinations. (For example if the three digits are 1, 2, 3 than all possible combinations are : 123, 132, 213, 231, 312, 321.)

* Code

import java.util.Scanner;

public class Ques1{

    public static void main(String[] args)

    {

        int a, b, c;

        Scanner s = new Scanner(System.in);

        System.out.print("Enter the first digit: ");

        a = s.nextInt();

        System.out.print("Enter the second digit: ");

        b = s.nextInt();

        System.out.print("Enter the third digit: ");

        c = s.nextInt();

        int[] ar = { a, b, c };

        for (int x = 0; x < 3; x++)

        {

            for (int y = 0; y < 3; y++)

            {

                for (int z = 0; z < 3; z++)

                {

                    if (x != y && y != z && z != x)

                    {

                        System.out.println(ar[x] + "" + ar[y] + "" + ar[z]);

                    }

                }

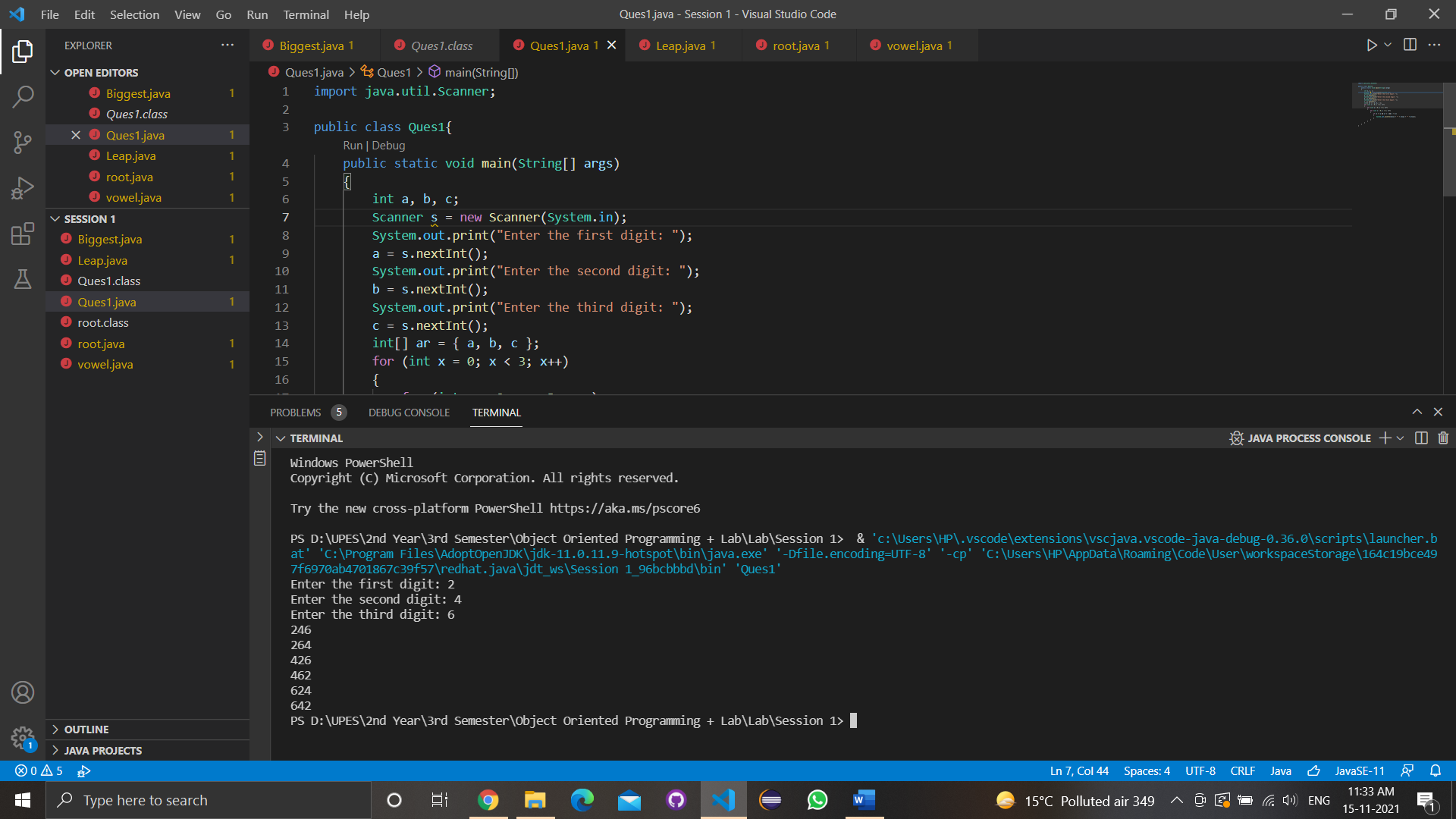
            }

        }

    }

}

* Output



1. Write a Java Program to accept 10 numbers in an array and compute the square of each number. Print the sum of these numbers.

* Code

import java.util.Scanner;

class square

{

    public static void main(String[] args)

    {

        int i,sum=0;

        for (i = 1; i <= 10; i++)

            sum += (i \* i);

            System.out.println(sum);

    }

}

* Output



1. Write a program to input a number of a month (1 - 12) and print its equivalent name of the month. ( e.g 1 to Jan, 2 to Feb. 12 to Dec.)

* Code

import java.util.Scanner;

class month

{

    public static void main(String[] args)

    {

        Scanner month = new Scanner(System.in);

        int n;

        System.out.print("\nMonth No: ");

        n = month.nextInt();

        switch(n)

        {

            case 1:

                    System.out.print("\nJanuary");

                    break;

            case 2:

                    System.out.print("\nFebruary");

                    break;

            case 3:

                    System.out.print("\nMarch");

                    break;

            case 4:

                    System.out.print("\nApril");

                    break;

            case 5:

                    System.out.print("\nMay");

                    break;

            case 6:

                    System.out.print("\nJune");

                    break;

            case 7:

                    System.out.print("\nJuly");

                    break;

            case 8:

                    System.out.print("\nAugust");

                    break;

            case 9:

                    System.out.print("\nSeptember");

                    break;

            case 10:

                    System.out.print("\nOctober");

                    break;

            case 11:

                    System.out.print("\nNovember");

                    break;

            case 12:

                    System.out.print("\nDecember");

                    break;

            default:

                    System.out.print("\nInvalid Month number\n");

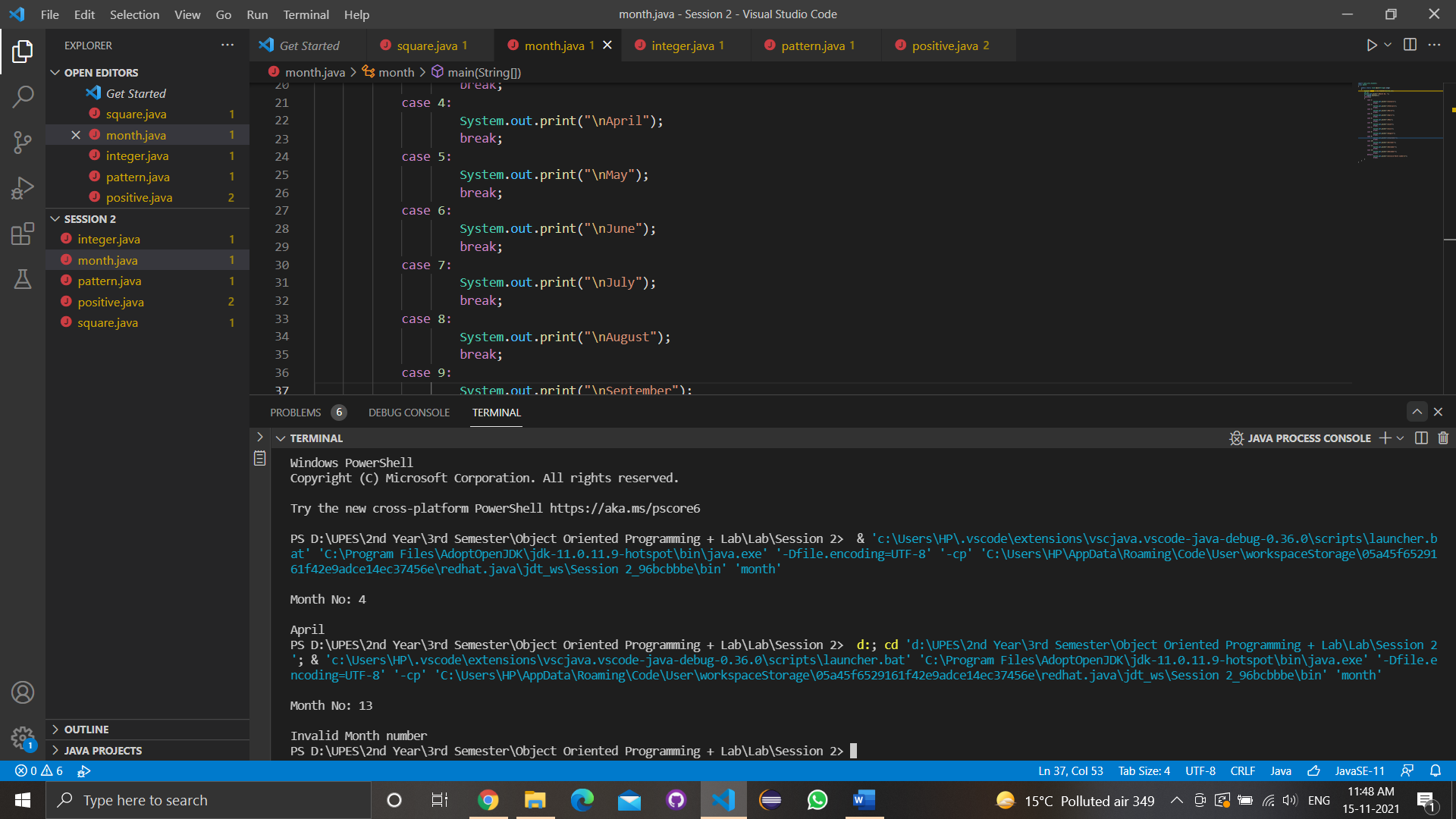
                    break;

        }

    }

}

* Output



1. Write a program to find the sum of all integers greater than 40 and less than 250 that are divisible by 5.

* Code

import java.util.Scanner;

public class integer

{

    public static void main(String arg[])

    {

        int sum = 0;

        for (int i = 41; i < 250; i++)

        {

            if (i % 5 == 0)

            {

                System.out.println(i);

                sum = sum + i;

            }

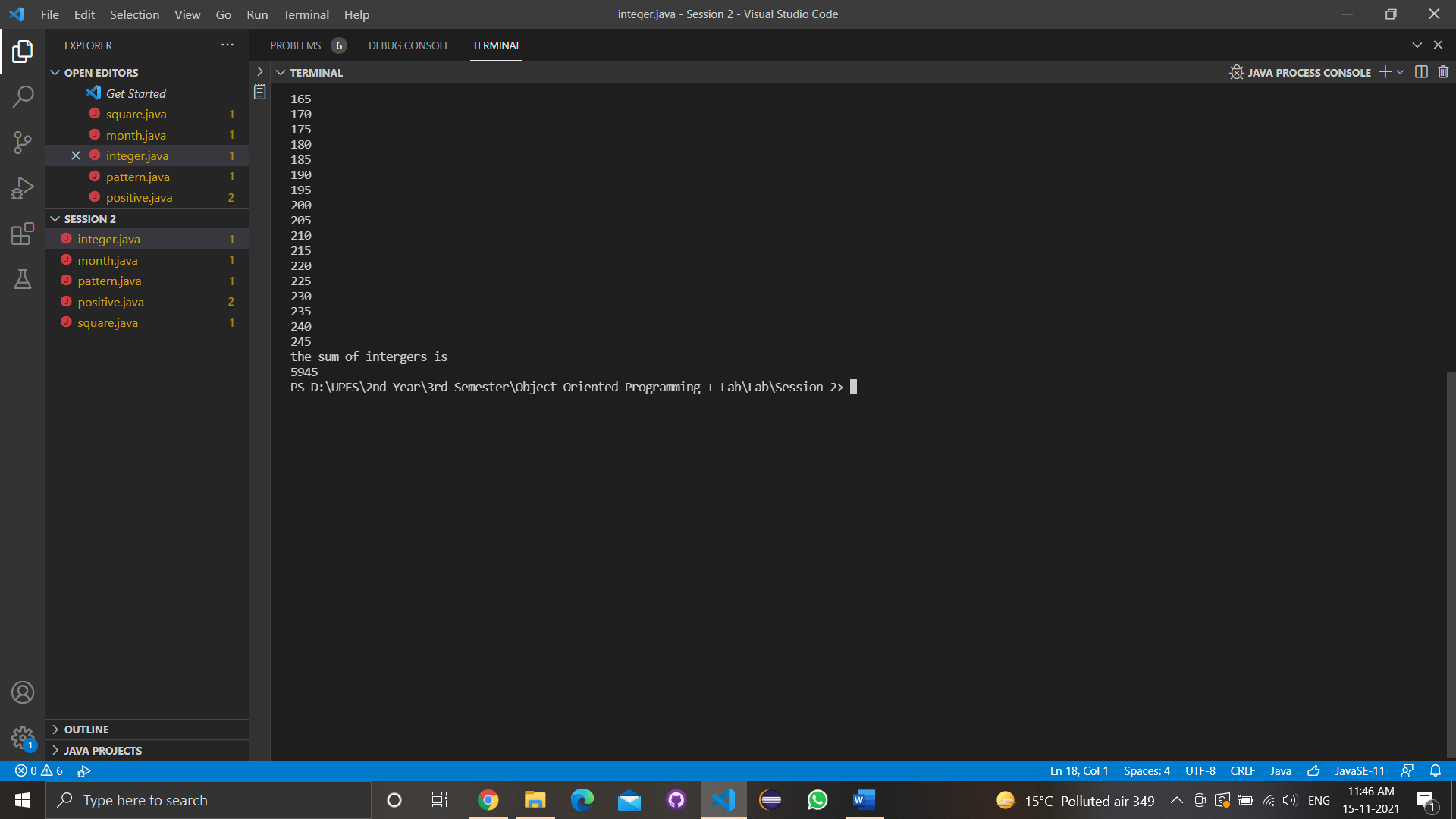
        }

        System.out.println("the sum of intergers is \n" + sum);

    }

}

* Output



------------------------------------------------------------------------------------------------

* Experiment 4
  + Lab Exercise

1. Write a Java program to show that private member of a super class cannot be accessed from derived classes.

* Code

class shape

{

    private int l,b;

        int area()

        {return(l\*b);}

}

class cuboid extends shape

{

    int h;

    cuboid(int l,int b,int d)

{

    super(l,b);

    h=d;

}

    int volume()

{

    return(area()\*h);

}

}

    class lab6

{

    public static void main(String args[])

    {

    cuboid c=new cuboid(16,12,4);

    int ar=c.area();

    int vl=c.volume();

    System.out.println("Area of Room : "+ar);

    System.out.println("Volume of Room : "+vl);

    }

}

* Output

2. Write a program in Java to create a Player class. Inherit the classes Cricket \_Player, Football \_Player and Hockey\_ Player from Player class.

* Code

class Player

{

    String name;

    int age;

    Player(String n,int a)

    {

        name=n; age=a;

    }

    void show()

    {

        System.out.println("Player name: "+name);

        System.out.println("Age: "+age);

    }

}

class cricket\_player extends Player

{

    String type;

    cricket\_player(String n,String t,int a)

    {

        super(n,a);

        type=t;

    }

    public void show()

    {

    super.show();

    System.out.println("Player type : "+type);

    System.out.println("------------------------");

    }

}

class football\_player extends Player

{

    String type;

    football\_player(String n,String t,int a)

    {

        super(n,a);

        type=t;

    }

    public void show()

    {

        super.show();

        System.out.println("Player type : "+type);

        System.out.println("------------------------");

    }

}

class hockey\_player extends Player

{

    String type;

    hockey\_player(String n,String t,int a)

    {

        super(n,a);

        type=t;

    }

    public void show()

    {

        super.show();

        System.out.println("Player type : "+type);

        System.out.println("------------------------");

    }

}

class Main

{

    public static void main(String[] args)

    {

        cricket\_player c=new cricket\_player("Manika ","Football",21);

        football\_player f=new football\_player("Tanisha","Cricket",21);

        hockey\_player h=new hockey\_player("Vidushi","Hockey",21);

        c.show();

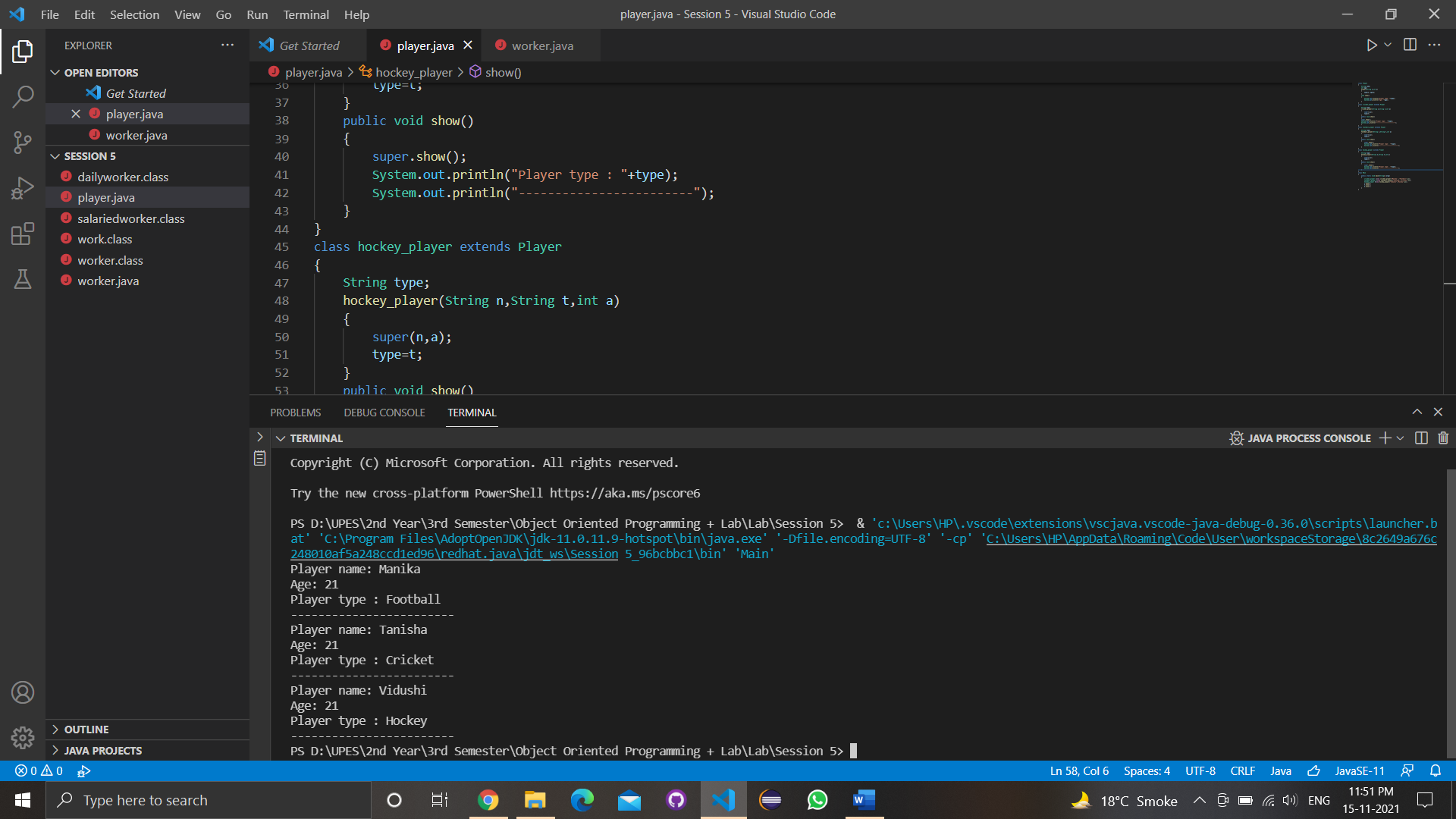
        f.show();

        h.show();

    }

}

* Output



1. Write a class Worker and derive classes DailyWorker and SalariedWorker from it. Every worker has a name and a salary rate. Write method ComPay (int hours) to compute the week pay of every worker. A Daily Worker is paid on the basis of the number of days he/she works. The Salaried Worker gets paid the wage for 40 hours a week no matter what the actual hours are. Test this program to calculate the pay of workers. You are expected to use the concept of polymorphism to write this program.

* Code

class worker {

    String name;

    int empno;

    worker(int no,String n){

    empno=no; name=n;

    }

    void show() {

    System.out.println("\n--------------------------");

    System.out.println("Employee number : "+empno);

    System.out.println("Employee name : "+name);

    }

    }

    class dailyworker extends worker {

    int rate;

    dailyworker(int no,String n,int r) {

    super(no,n);

    rate=r;

    }

    void compay(int h) {

    show();

    System.out.println("Salary : "+rate\*h);

    }

    }

    class salariedworker extends worker {

    int rate;

    salariedworker(int no,String n,int r) {

    super(no,n);

    rate=r;

    }

    int hour=40;

    void compay() {

    show();

    System.out.println("Salary : "+rate\*hour);

    }

    }

    class work {

    public static void main(String[] args) {

    dailyworker d=new dailyworker(2654465,"Satwik Jha",705);

    salariedworker s=new salariedworker(62646655,"Saksham Ratan",231);

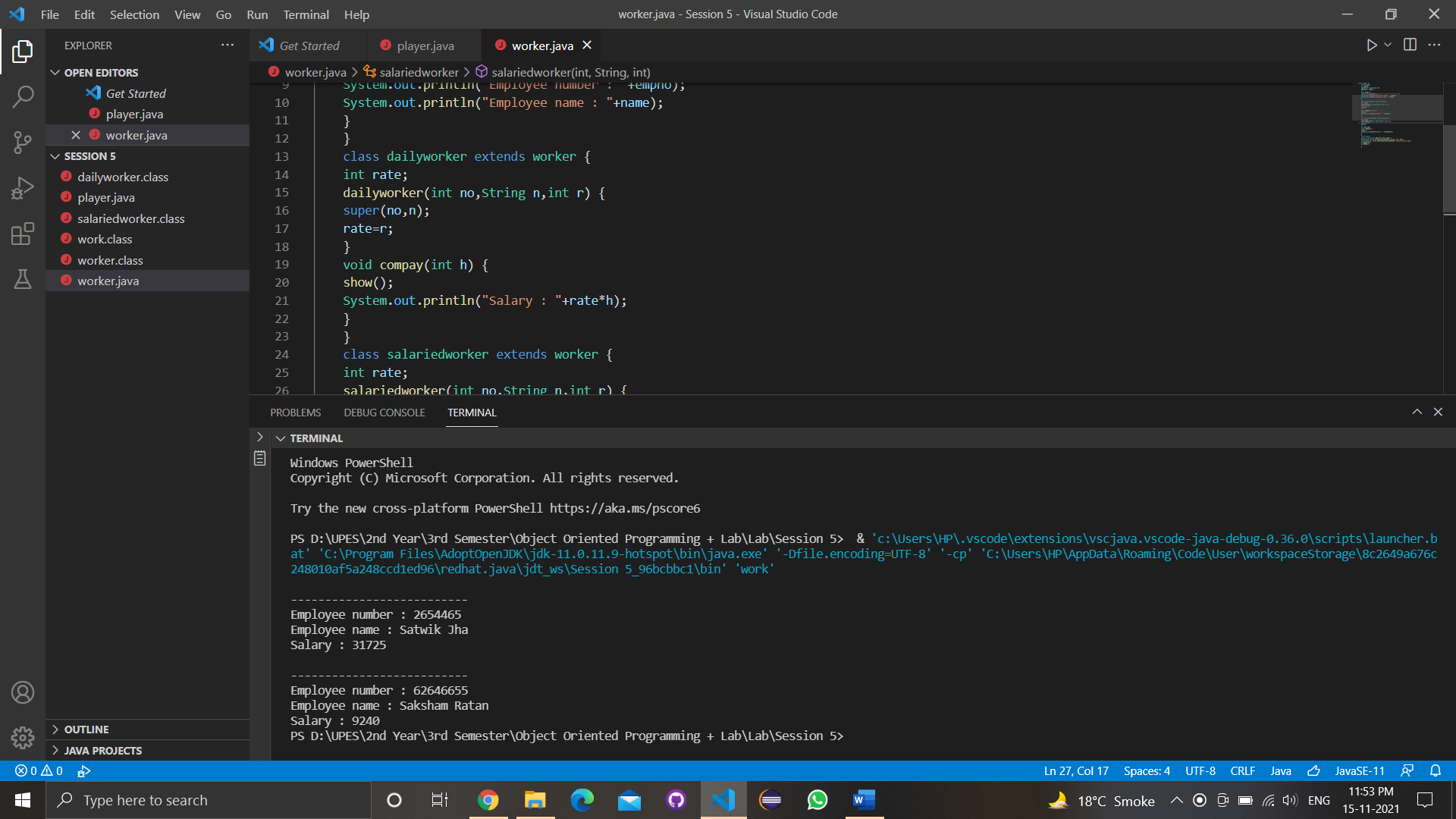
    d.compay(45);

    s.compay();

    }

    }

* Output



1. Consider the trunk calls of a telephone exchange. A trunk call can be ordinary, urgent or lightning. The charges depend on the duration and the type of the call. Write a program using the concept of polymorphism in Java to calculate the charges.

* Code

class call {

    String x;

    int y;

    call(String n,int time)

    { x=n; y=time; }

    void show()

    {

        System.out.println("\n--------------------------");

        System.out.println("Call Type: "+x);

        System.out.println("Time used: "+y);

    }

}

class Ordinary extends call {

    int z;

    Ordinary(String name,int min,int r)

    {

        super(name,min);

        z=r;

    }

    void show()

    {

        super.show();

        System.out.println("Call Rate: "+z+" Rs/min");

        System.out.println("Total Cost: "+y\*z);

    }

}

class Urgent extends call {

    int z;

    Urgent(String name,int min,int r)

    {

        super(name,min);

        z=r;

    }

    void show()

    {

        super.show();

        System.out.println("Call Rate: "+z+" Rs/min");

        System.out.println("Total Cost: "+y\*z);

    }

}

class Lightening extends call {

    int z;

    Lightening(String name,int min,int r)

    {

        super(name,min);

        z=r;

    }

    void show()

    {

        super.show();

        System.out.println("Call Rate: "+z+" Rs/min");

        System.out.println("Total Cost: "+y\*z);

    }

}

public class trunk {

    public static void main(String[] args) {

        Lightening lc=new Lightening("lightening",14, 12);

        Urgent uc=new Urgent("urgent", 15, 10);

        Ordinary oc= new Ordinary("ordinary", 18,10);

        lc.show();

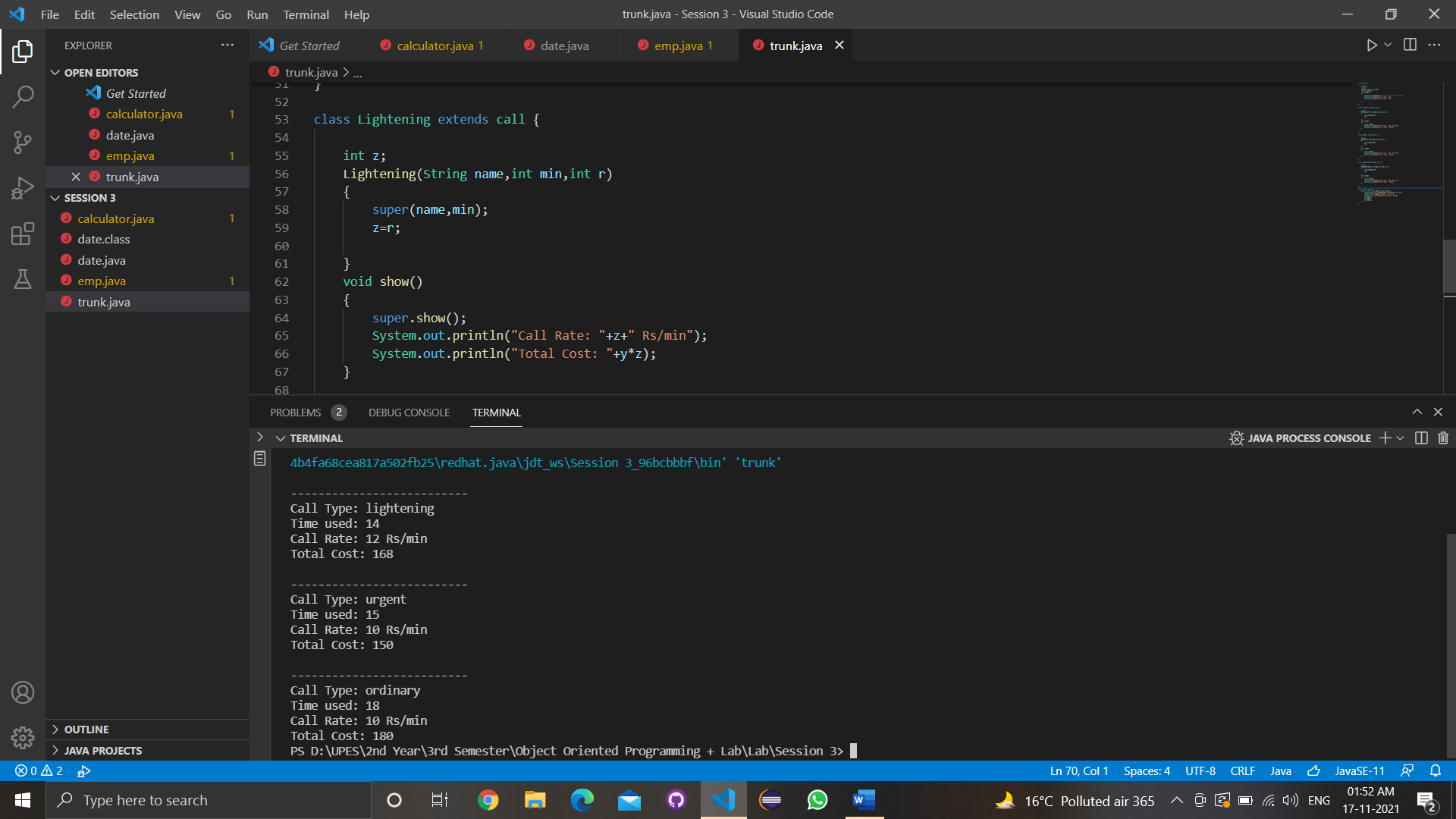
        uc.show();

        oc.show();

    }

}

* Output



1. Design a class employee of an organization. An employee has a name, empid, and salary. Write the default constructor, a constructor with parameters (name, empid, and salary) and methods to return name and salary. Also write a method increaseSalary that raises the employee’s salary by a certain user specified percentage. Derive a subclass Manager from employee. Add an instance variable named department to the manager class. Supply a test program that uses theses classes and methods.

* Code

class EmployeeDetails {

    String n;

    int id;

    int pay;

    EmployeeDetails(String name,int emp\_id, int salary)//parameterised constructor

    { n=name; id=emp\_id;pay=salary; }

    void increase\_salary(int x) {

        pay = pay+ ((x\*pay)/100);

        System.out.println("The increased salary is : "+pay);

    }

    EmployeeDetails(){}//default constructor

    void show()

    {

    System.out.println("\n--------------------------");

    System.out.println("Name of Employee: "+n);

    System.out.println("Employee id: "+id);

    System.out.println("Salary of Employee: "+pay);

    }

    void type() {System.out.println("This is a Employee");}

}

class Manager extends EmployeeDetails {

    String department="Technical";

    Manager(String name,int emp\_id, int salary){

        super(name,emp\_id,salary);

    }

        void print()

        {

            show();

            System.out.println("Department: "+ department);

        }

    void type() {System.out.println("This is a Manager");}//method overriding

    void type(int a) {System.out.println("This is also a Manager form "+a+" years");}//method overloading

    Manager(){

        show();

        System.out.println("This is also a manager");

        System.out.println("Default constructor Created");

        }

    }

public class emp {

    public static void main(String args[])

    {

    Manager m = new Manager("Manika", 18, 48200);  //parameterised constructor

                      //name/ emp\_id/ salary

    m.print();

    m.increase\_salary(50);

    m.type();

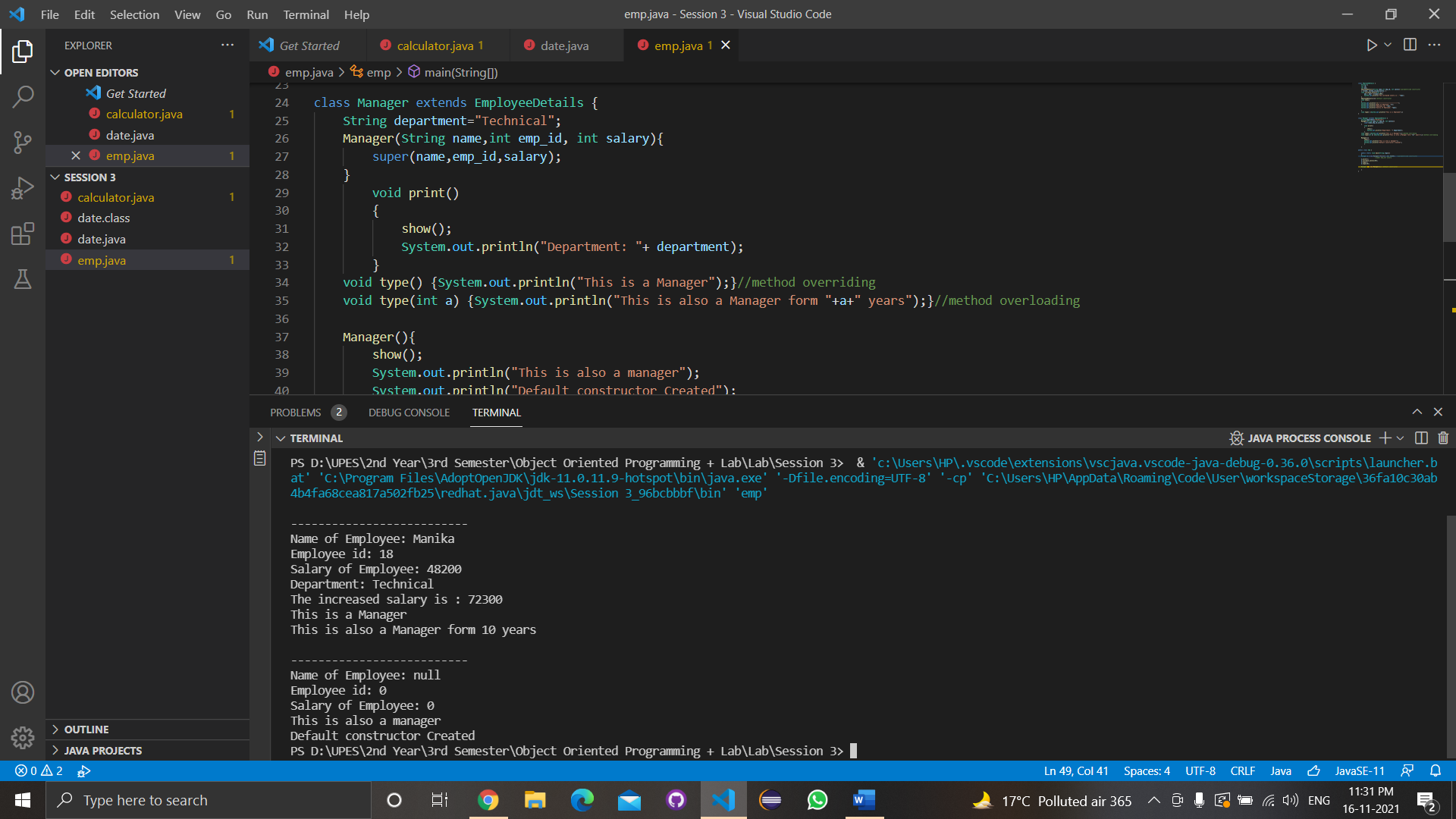
    m.type(10);

    Manager m2= new Manager();// default constructor

    }

}

* Output



----------------------------------------------------------------------------------------------------------------

* Experiment 5
  + Lab Exercise

1.Write a program to create interface named test. In this interface the member function is square. Implement this interface in arithmetic class. Create one new class called ToTestInt. In this class use the object of arithmetic class.

* Code

interface Test

{

    int operation(int x, int y);

}

class Addition implements Test

{

    public int operation(int x, int y)

    {

        return x+y;

    }

}

class Multiplication implements Test

{

    public int operation(int x, int y)

    {

        return x\*y;

    }

}

class ToTestInt

{

    public static void main(String arg[])

    {

        Addition a = new Addition();

        Multiplication m = new Multiplication();

        Test c;

        c = a;

        System.out.println("Addition: "+c.operation(8,10));

        c = m;

        System.out.println("Multiplication: "+c.operation(5,8));

    }

}

* Output



2. Write a program to create interface A, in this interface we have two method meth1 and meth2. Implements this interface in another class named MyClass.

* Code

interface A

{

    void meth1();

    void meth2();

}

class MyClass implements A

{

        public void meth1()

        {

            System.out.println("Hello");

        }

        public void meth2()

        {

            System.out.println ("World");

        }

        public static void main (String arg[])

        {

            MyClass m = new MyClass();

            m.meth1();

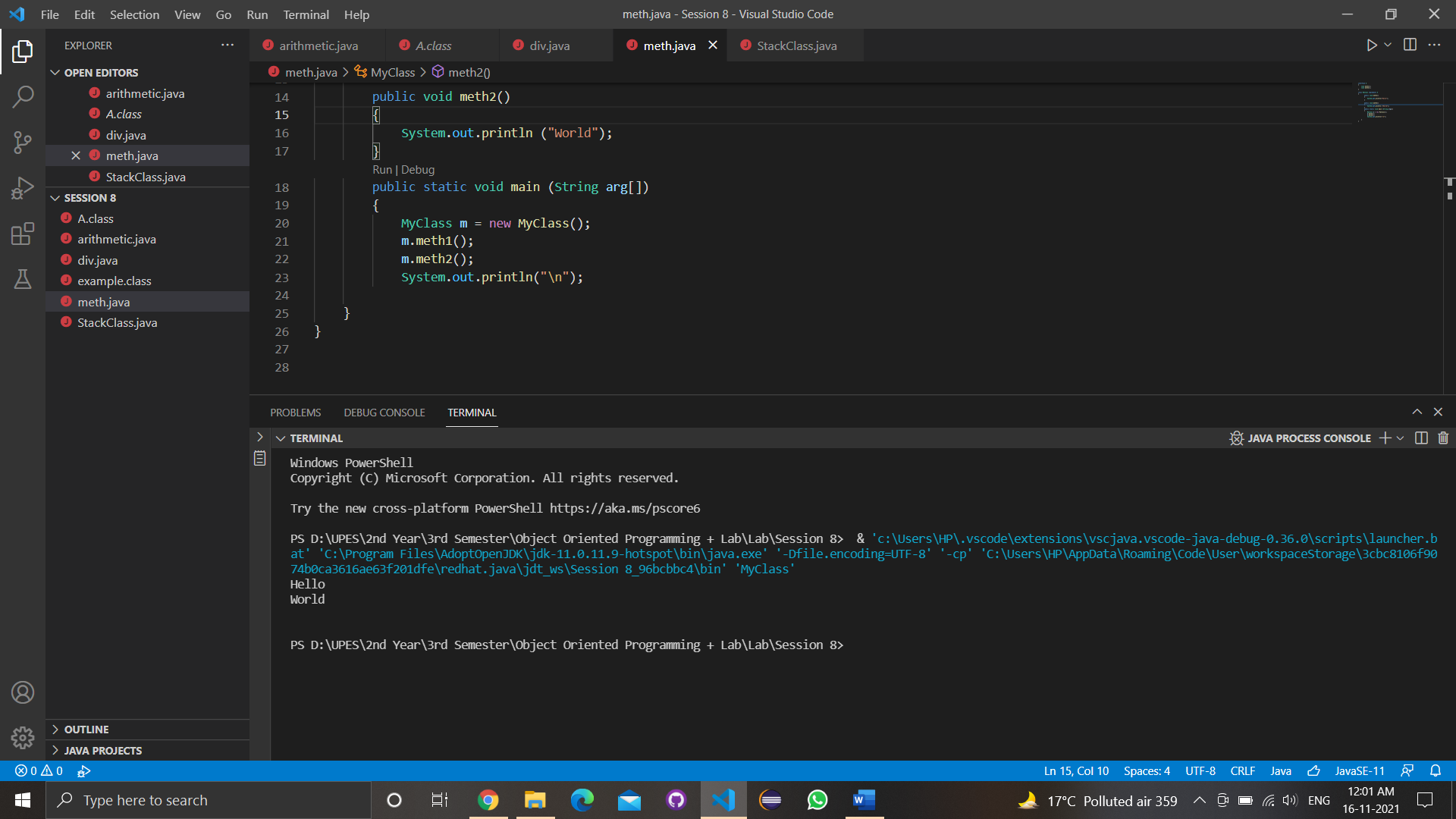
            m.meth2();

            System.out.println("\n");

    }

}

* Output



3. Write a program in Java to show the usefulness of Interfaces as a place to keep constant value of the program.

* Code

interface area

{

static final float pi=3.142f;

float compute(float x,float y);

}

class rectangle implements area

{

public float compute(float x,float y)

{return(x\*y);}

}

class circle implements area

{

public float compute(float x,float y)

{return(pi\*x\*x);}

}

class constant

{

public static void main(String args[])

{

rectangle rect=new rectangle();

circle cr=new circle();

area ar;

ar=rect;

System.out.println("Area of the rectangle= "+ar.compute(20,5));

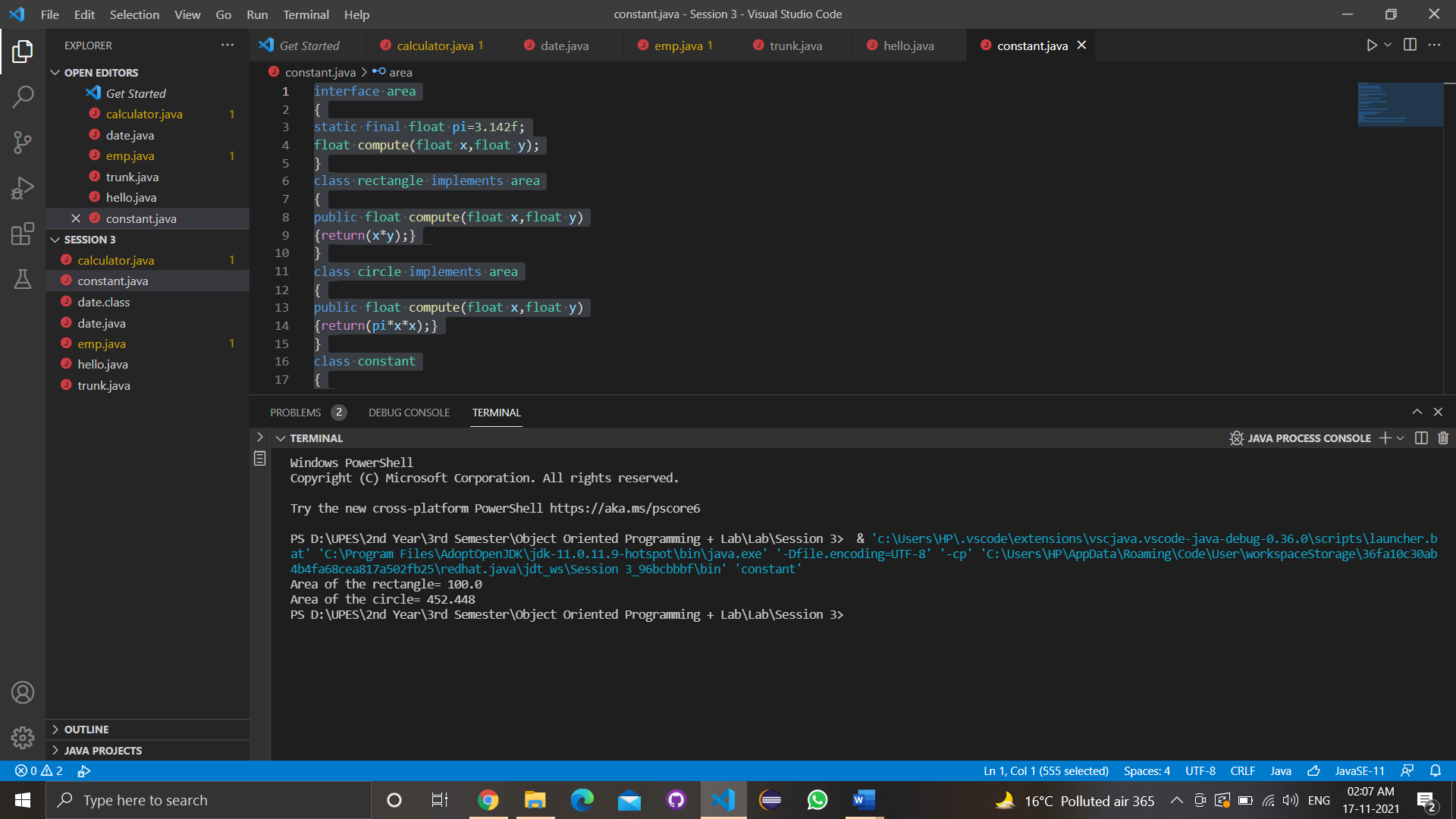
ar=cr;

System.out.println("Area of the circle= "+ar.compute(12,0));

}

}

* Output



4.Write a program to create an Interface having two methods division and modules. Create a class, which overrides these methods.

* Code

interface subject

{

    void division(int x);

    void modules(int y);

}

class sub implements subject

{

    int div, mod;

    public void division(int x)

    {

        div = x;

    }

    public void modules(int y)

    {

        mod = y;

    }

    void print(){

        System.out.println("Division: "+div);

        System.out.println("Modules: "+mod);

    }

}

class div

{

    public static void main(String arg[])

    {

        sub s = new sub();

        s.division(1);

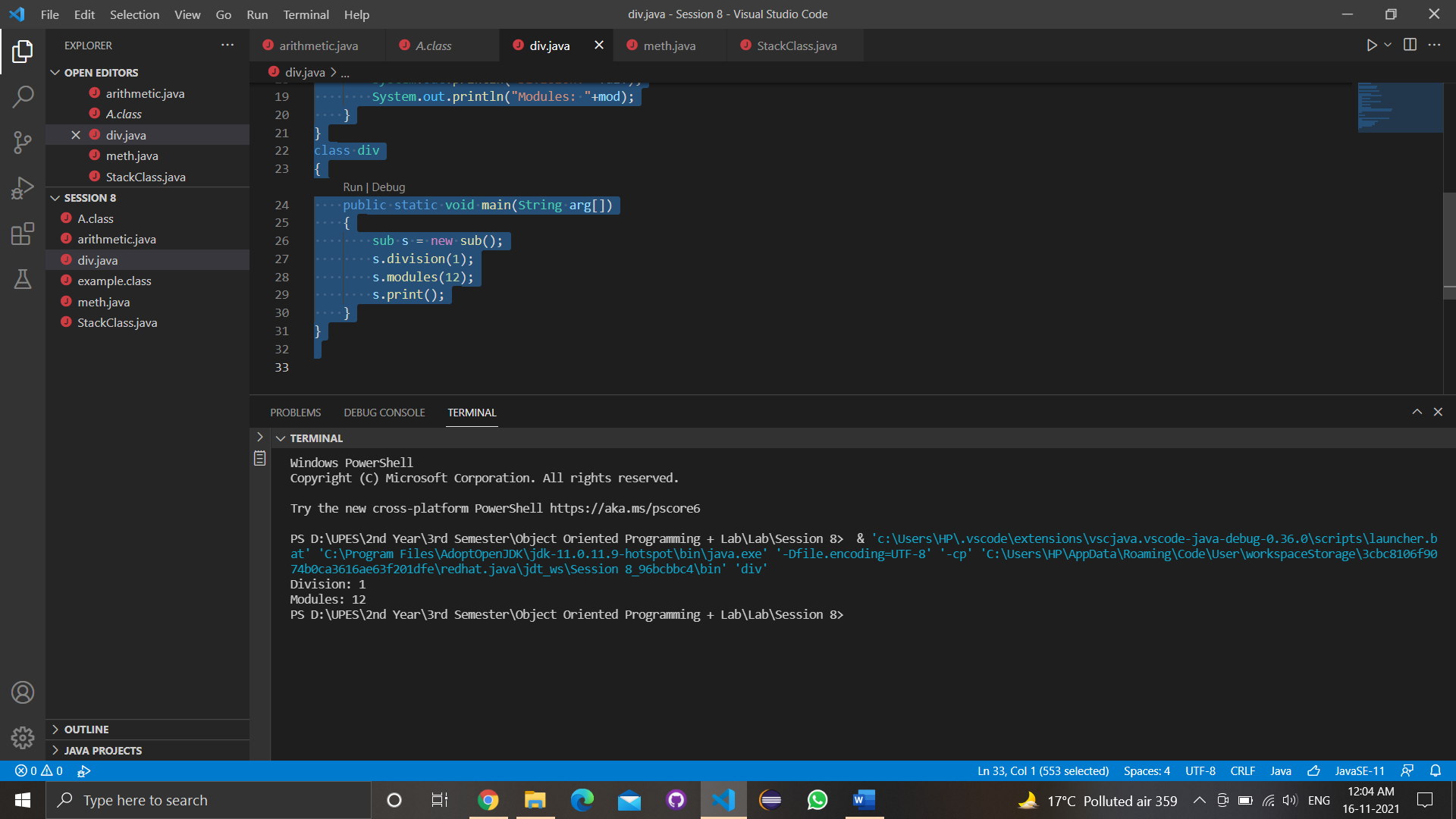
        s.modules(12);

        s.print();

    }

}

* Output



1. Write program to create an interface StackInterface having methods push (), pop () and display (). StackClass implements StackInterface. Class StackClass contains the main method which is having a switch case for selecting the particular operation of the stack.

* Code

interface StackInterface

class stack

{

// Pushing element on the top of the stack

static void stack\_push(Stack<Integer> stack)

{

System.out.println("Pushing elements>>");

for(int i = 0; i < 5; i++)

{

stack.push(i);

}

System.out.println("Stack formed:");

for(int i = 0; i < 5; i++)

{

System.out.println(i);

}

}

// Popping element from the top of the stack

static void stack\_pop(Stack<Integer> stack)

{

System.out.println("Pop Operation:");

Integer y = (Integer) stack.pop();

System.out.println(y);

}

// Displaying element on the top of the stack

public static void main (String[] args)

{

Stack<Integer> stack = new Stack<Integer>();

stack\_push(stack);

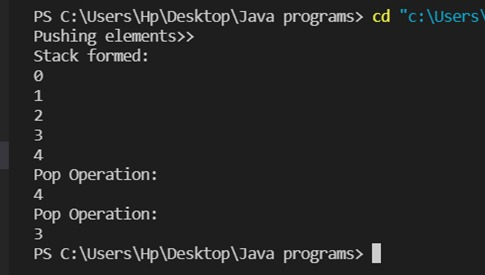
stack\_pop(stack);

stack\_pop(stack);

}

}

* Output



--------------------------------------------------------------------------------------------------------------------------

* Experiment 6
  + Lab Exercise

1. Write a Java program to implement the concept of importing classes from user defined package and created packages.

* Code

package pack;

public class A

{

    public void msg()

    {

        System.out.println("Hello");

    }

}

import pack.\*;

class B

{

    public static void main(String args[])

    {

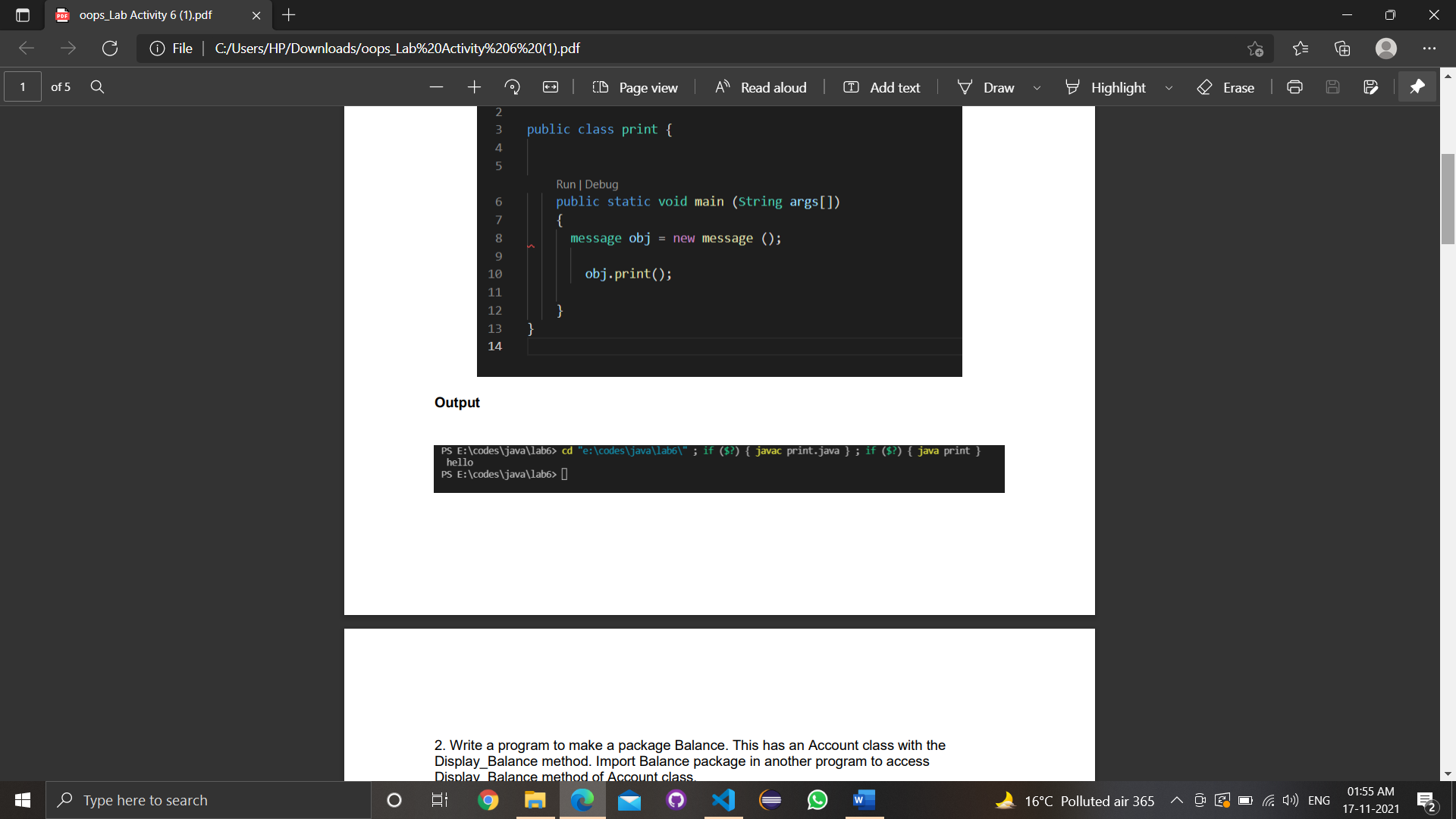
        A obj = new A();

        obj.msg();

    }

}

* Output



1. Write a program to make a package Balance. This has an Account class with Display\_Balance method. Import Balance package in another program to access Display\_Balance method of Account class.

* Code

package balance;

public class Account

{

    public void Display\_Balance(ina a)

    {

        System.out.println("balance",+a);

    }

}

import balance.\*;

{

    class Ques2

    {

        public static void main(String args[])

        {

            Account obj = new Account();

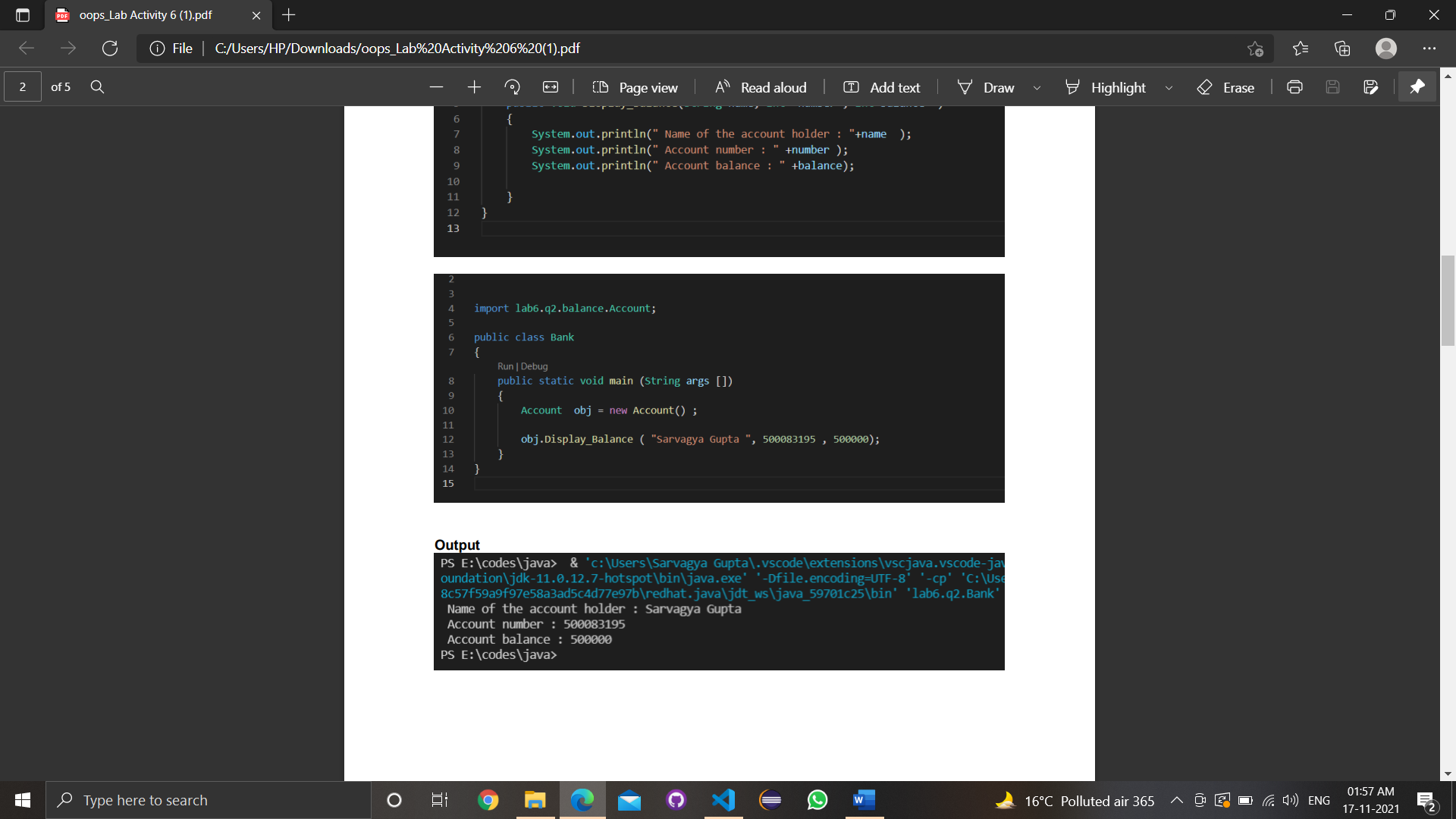
            obj.Display\_Balance(500000);

        }

    }

}

* Output



1. WAP to create a package p with class A with 4 types of access protected methods. How we will use these methods in different packages class i.e. there is main() in class B in package Q and 4 methods are in Class A in package p.

* Code

package p;

public class A

{

    public static void meth1()

    {

        System.out.println("This is a Public Method");

    }

    protected static void meth2()

    {

        System.out.println("This is a Protected Method");

    }

    private static void meth3()

    {

        System.out.println("This is a private Method");

    }

    static void meth4()

    {

        System.out.println("this is default method");

    }

    public static void main(String[] args)

    {

        A a = new A();

        A.meth1();

        A.meth2();

        A.meth3();

        A.meth4();

    }

}

public class B extends A

{

    public static void main(String[] args)

    {

        B b = new B();

        B.meth1();

        B.meth2();

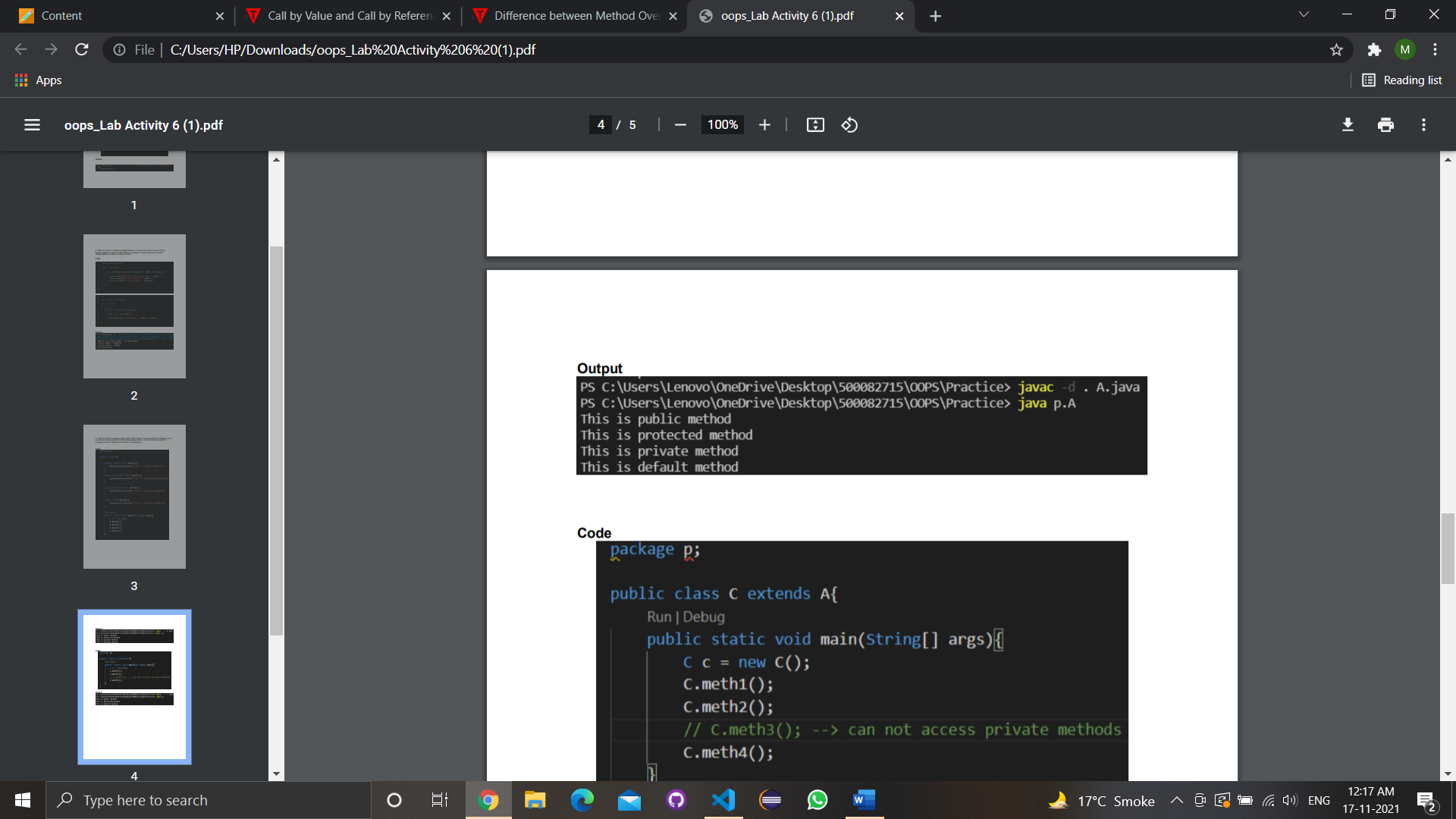
        //B.meth3(); cannot access as private

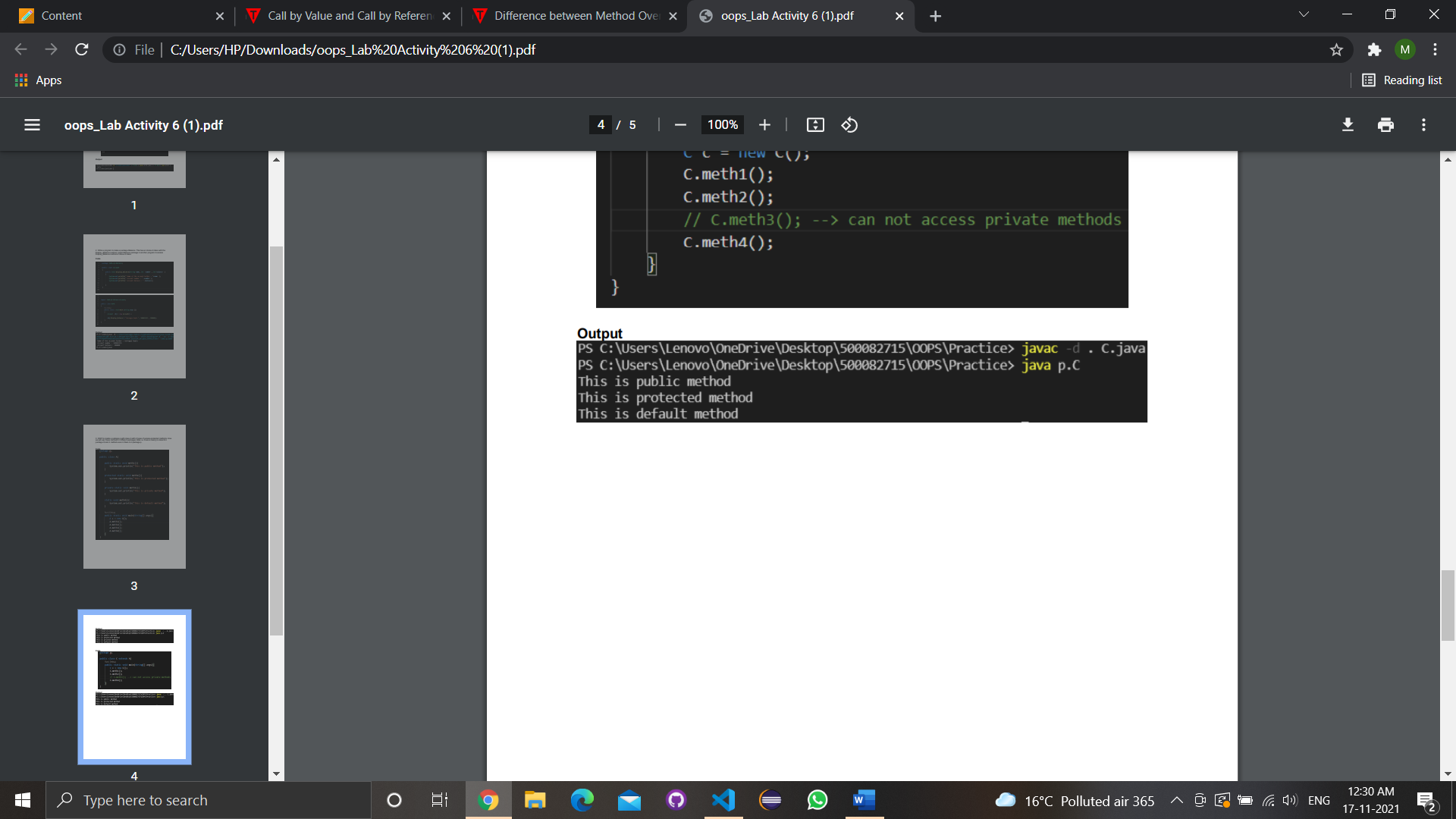
        B.meth4();

    }

}

* Output





------------------------------------------------------------------------------------------------