**Name : Nandini Srivastava**

**Pin no : 17551A0598**

1,Create a class Publication with data members title(String) and price(int). From this class derive two classes Book and CD. Class Book adds pages(int) and CD adds Size(int). Each of these classes should have constructors and display(). Write a java program to implement this using super, this and method overriding concepts.

**package** com.rsc.stringpalindrome;

**import** java.util.Scanner;

**class** Publication

{

String title;

**int** price;

**public** Publication(String title,**int** price)

{

**this**.title=title;

**this**.price = price;

}

**void** display()

{

System.***out***.println("I AM IN PUBLICATION CLASS");

System.***out***.println("TITLE : " + title +" PRICE : " + price);

}

}

**class** Book **extends** Publication//adds pages

{

**int** pages;

Book(String title,**int** price,**int** pages)

{

**super**(title,price);

**this**.pages =pages;

}

**void** display()

{

**super**.display();

System.***out***.println("I AM IN BOOK CLASS");

System.***out***.println("TITLE : " + title +" PRICE : " + price + " PAGES : " +pages);

}

}

**class** CD **extends** Book//adds size

{

**int** size;

CD(String title,**int** price,**int** pages,**int** size)

{

**super**(title,price, pages);

**this**.size=size;

}

**void** display()

{

**super**.display();

System.***out***.println("I AM IN CD CLASS");

System.***out***.println("TITLE : " + title +" PRICE : " + price + " PAGES : " +pages + " SIZE : " + size);

}

}

**public** **class** Inheritence {

**public** **static** **void** main(String a[])

{

Scanner sc = **new** Scanner(System.***in***);

String title= sc.next();

**int** price = sc.nextInt();

**int** pages = sc.nextInt();

**int** size = sc.nextInt();

CD cd = **new** CD(title,price,pages,size);

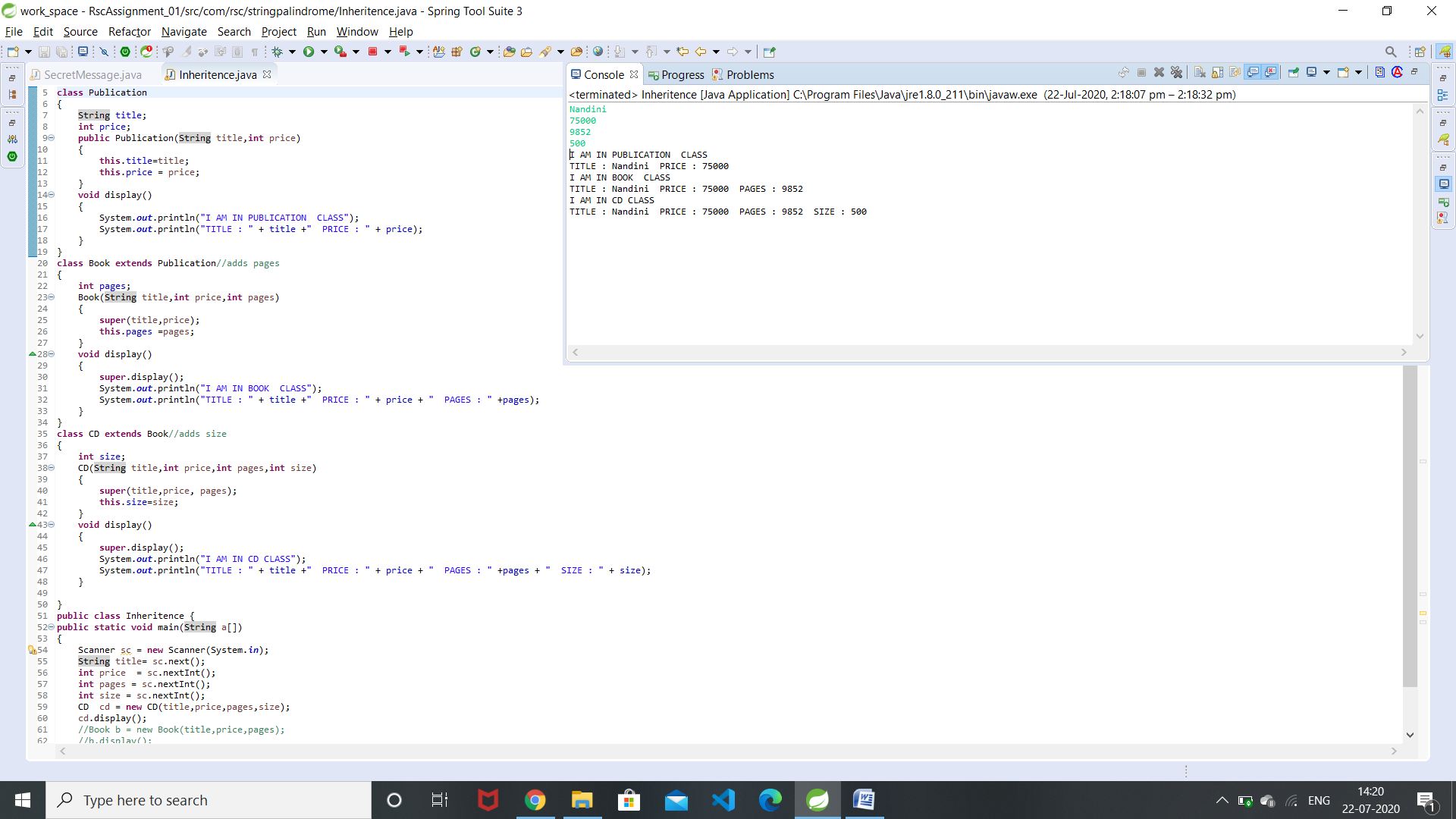
cd.display();

//Book b = new Book(title,price,pages);

//b.display();

}

}



2.Write a simple java program to demonstrate method overriding.

**package** com.rsc.stringpalindrome;

**import** java.util.Scanner;

**class** A

{

**void** name(String n)

{

System.***out***.print(" I am " + n + " from class A");

}

}

**class** B **extends** A

{

**void** name(String n)

{

System.***out***.print(" I am " + n + " from class B");

}

}

**public** **class** MethodOverriding {

**public** **static** **void** main(String a[])

{

Scanner sc =**new** Scanner(System.***in***);

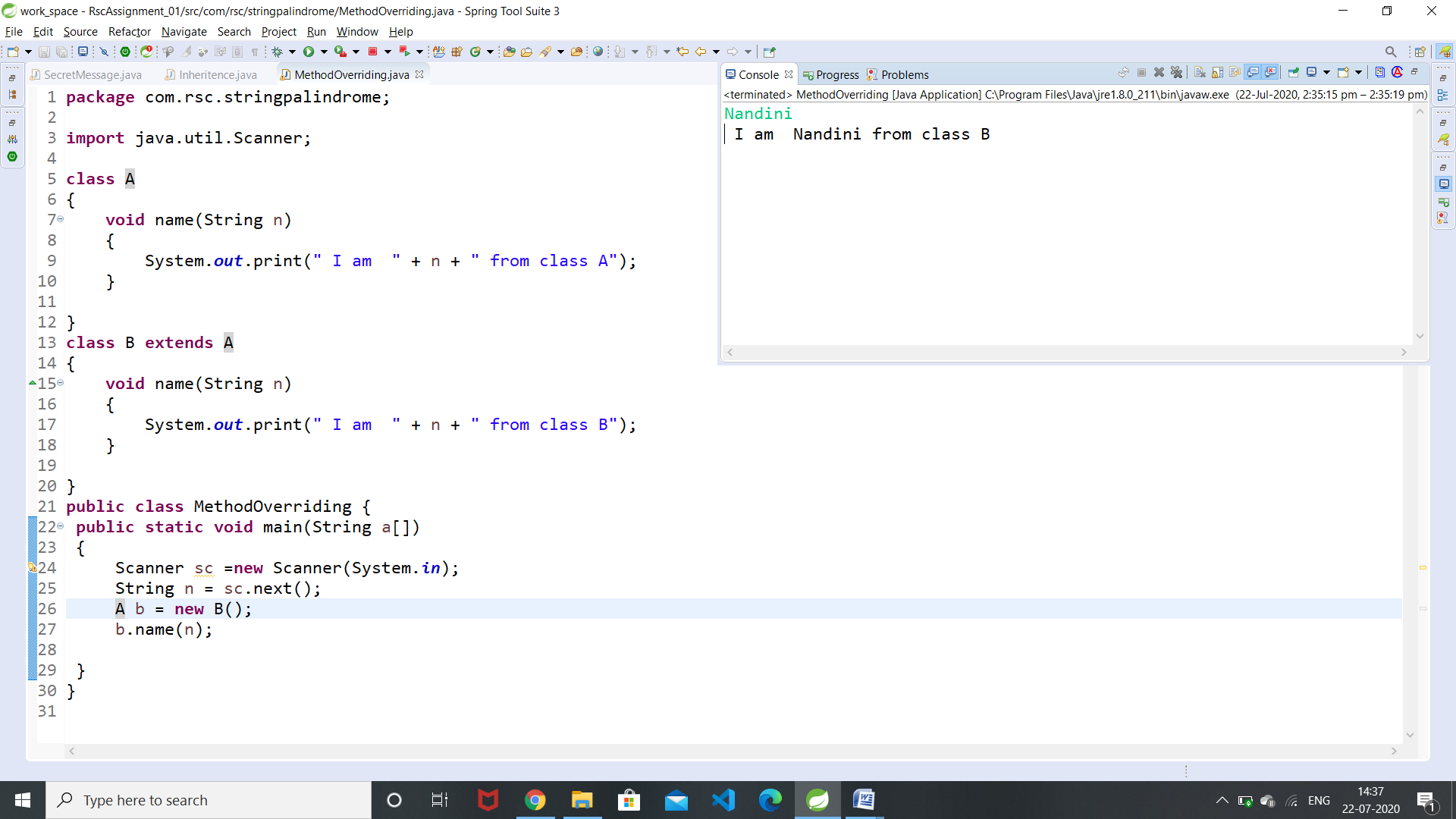
String n = sc.next();

A b = **new** B();

b.name(n);

}

}



3.Write a java program to create an interface called Shape with CalculateArea(). Create three classes namely Square,Circle,Triangle which implements Shape.

**package** com.rsc.stringpalindrome;

**import** java.util.Scanner;

**interface** Shape

{

**void** CalculateArea();

}

**class** Square **implements** Shape

{

**float** side ;

Square(**float** side)

{

**this**.side = side;

}

**public** **void** CalculateArea()

{

**double** result = side\*side;

System.***out***.printf("Area of a Square is : %.4f %n" , result);

}

}

**class** Circle **implements** Shape

{

**float** radius;

Circle(**float** radius)

{

**this**.radius = radius;

}

**public** **void** CalculateArea()

{

**double** result = 3.14f\*radius\*radius;

System.***out***.printf("Area of a Circle is : %.4f %n" , result);

}

**class** Triangle **implements** Shape

{

**float** base,height;

Triangle(**float** base,**float** height)

{

**this**.base=base;

**this**.height = height;

}

**public** **void** CalculateArea()

{

**double** result = 0.5\*base\*height;

System.***out***.printf("Area of a Triangle is : %.4f %n" , result);

}

}

**public** **class** InterfaceExample {

**public** **static** **void** main(String a[])

{

Scanner sc = **new** Scanner(System.***in***);

**float** side = sc.nextFloat();

**float** radius = sc.nextFloat();

**float** base = sc.nextFloat();

**float** height = sc.nextFloat();

Square s = **new** Square(side);

Circle c = **new** Circle(radius);

Triangle t = **new** Triangle(base,height);

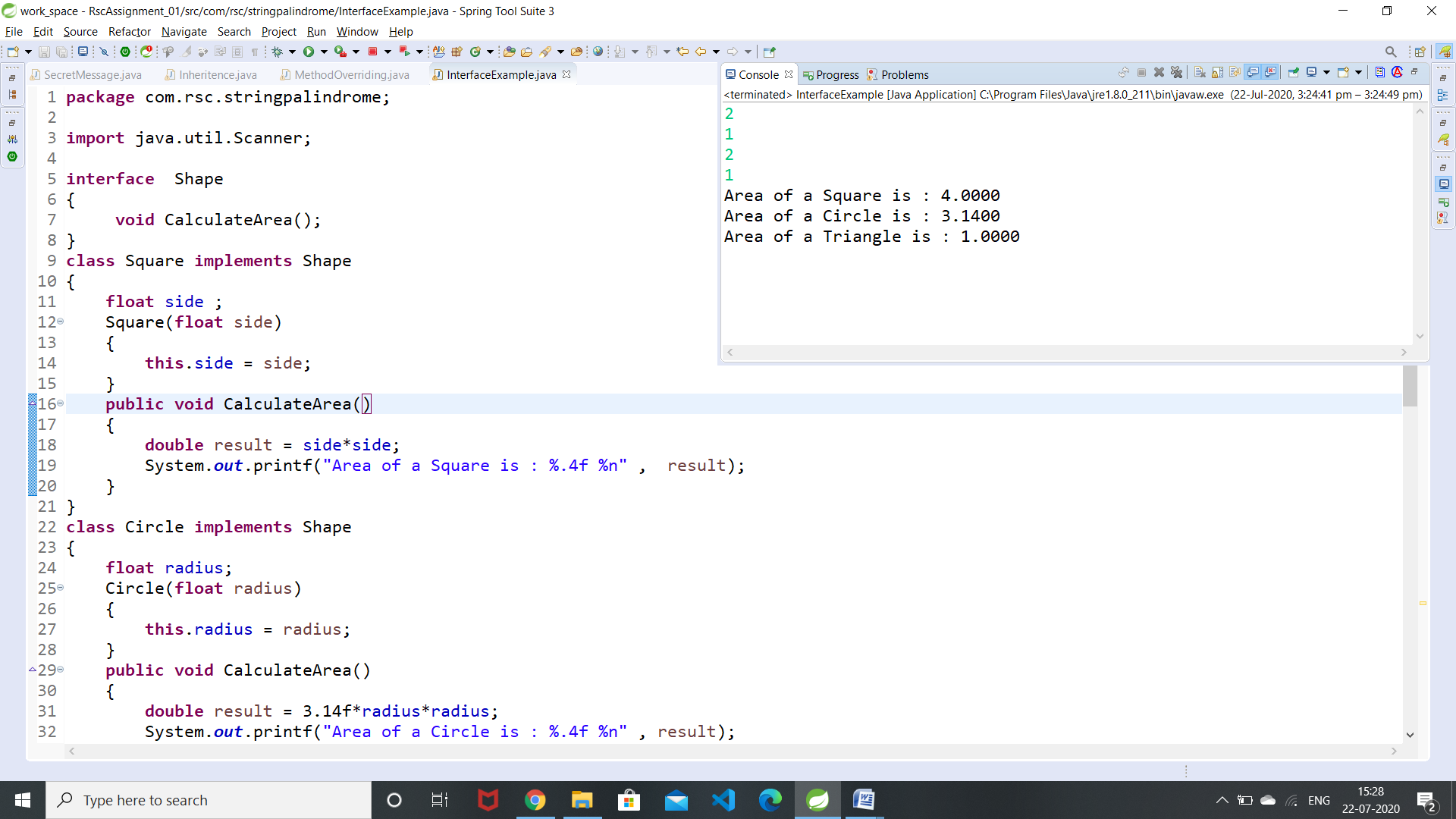
s.CalculateArea();

c.CalculateArea();

t.CalculateArea();

}

}



4.Create two packages p1 and p2. The package p1 contains class A which contains one display(). Create class B in package p2. The main method of class B invoke A’s display(). Write a java program to do this.

**package** p1;

**public** **class** A {

**public** **static** **void** display()

{

System.***out***.println("I am in Class A ");

}

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

*display*();

}

}

**package** p2;

**import** p1.A;

**public** **class** B {

**public** **static** **void** display()

{

System.***out***.println("I am in Class B ");

}

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

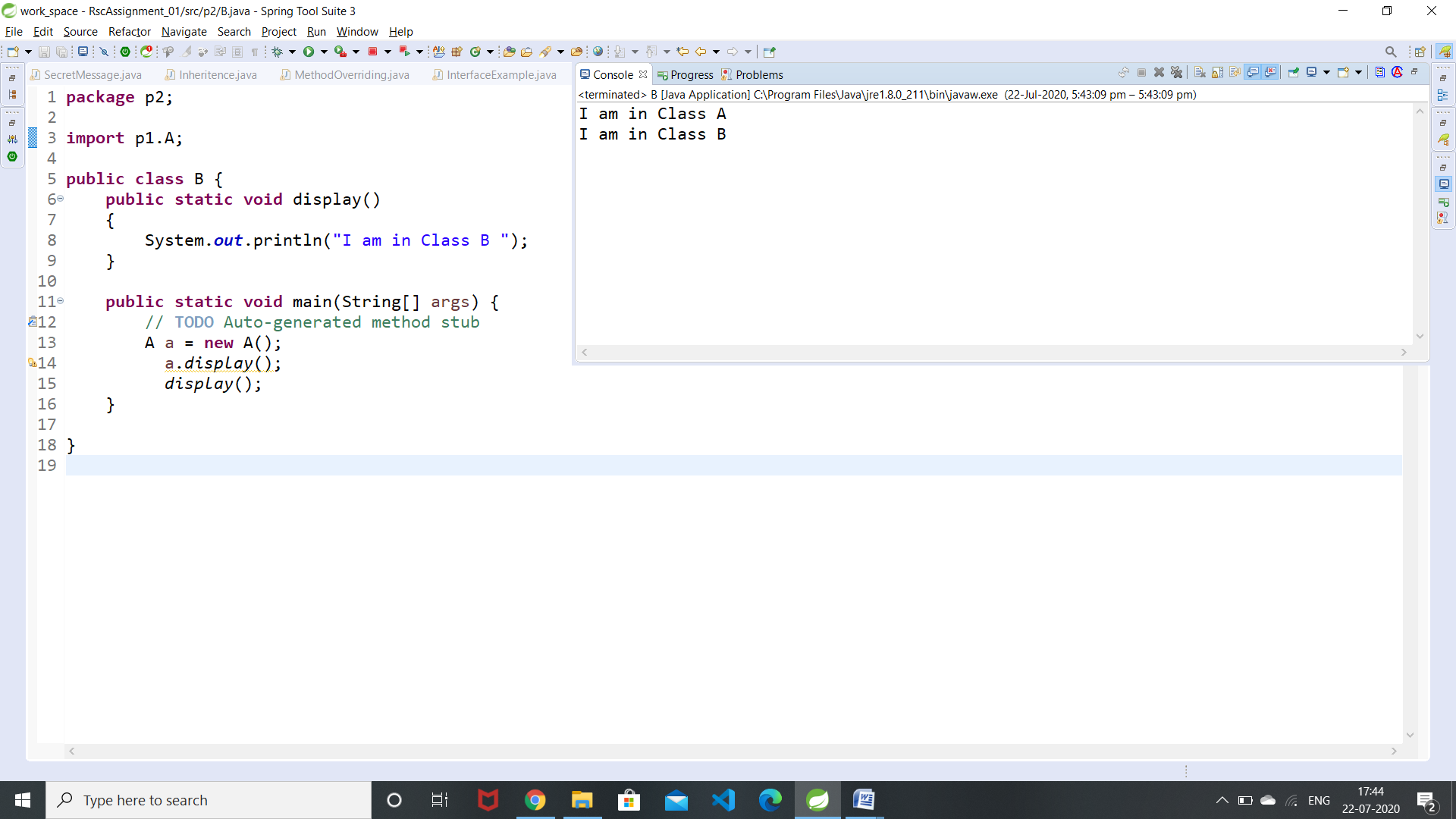
A a = **new** A();

a.*display*();

*display*();

}

}



5.Write a java program to count numbers, characters in the command line arguments using Exception handling mechanism.

public class CmdLineArg

{

public static void main(String args[])

{

int n1=0;

int c1=0;

int n=args.length;

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

{

for(int j=0;j<args[i].length();j++)

{

if(Character.isDigit(args[i].charAt(j)))

n1+=1;

else

c1+=1;

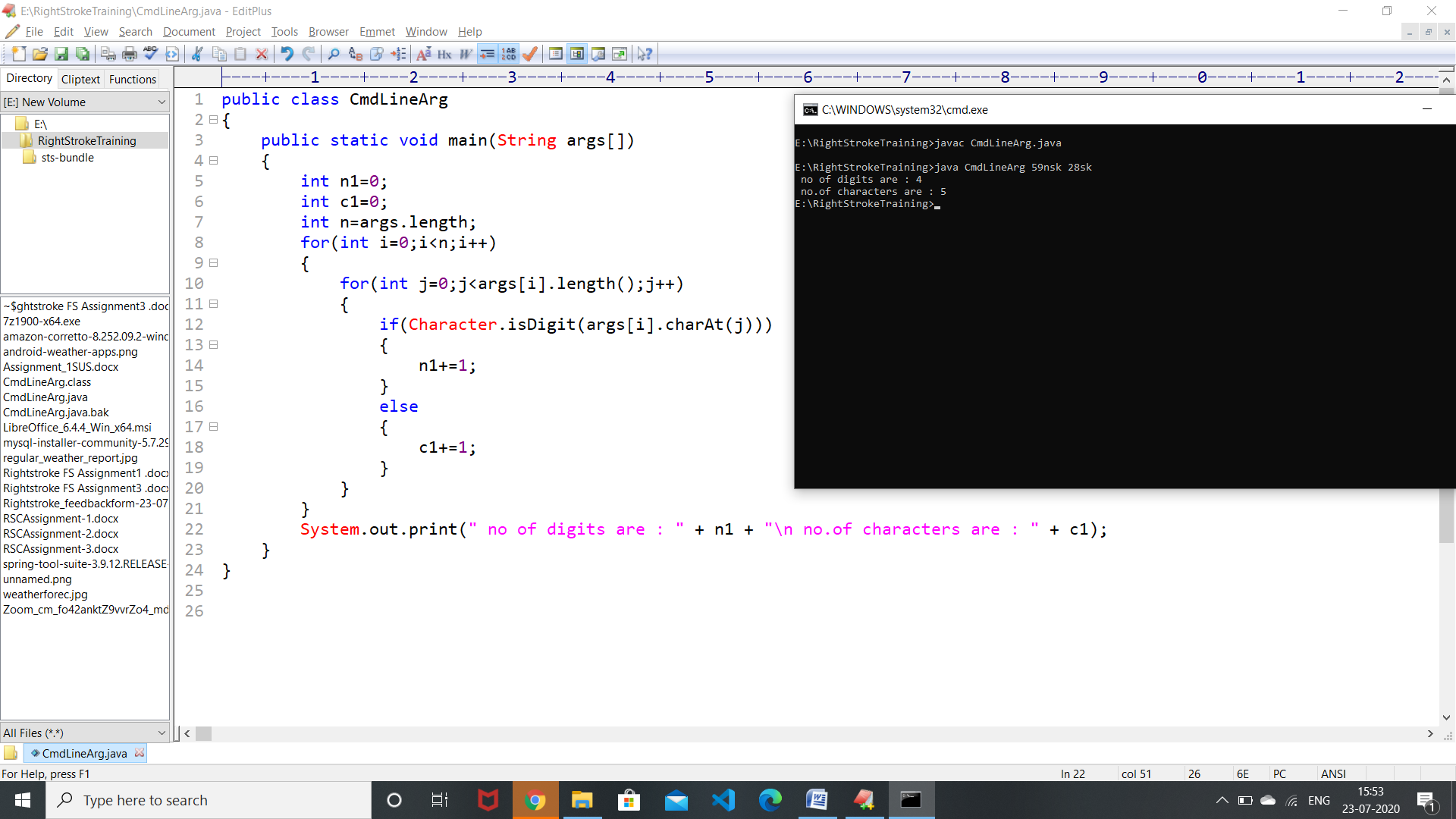
}

}

System.out.print(" no of digits are : " + n1 + "\n no.of characters are : " + c1);

}

}

****

**1. What is Inheritance?**

**Inheritance** is a process of defining a new class based on an existing class by extending its common data members and methods The class which **inherits** the properties of other is known as subclass (derived class, child class) and the class whose properties are **inherited** is known as superclass (base class, parent class).**Inheritance** is a process of defining a new class based on an existing class by extending its common data members and methods.

**2. What is Multiple Inheritance?**

When one class extends more than one classes then this is called **multiple inheritance**.  **java** doesnt **support multiple inheritance** but through interface java supports multiple inheritance. For **example**: Class C extends class A and B then this type of **inheritance** is known as **multiple inheritance**

**3. What is the use of Super keyword?**

Super keyword  is used when we want to call parent class variables and methods. The **super() in Java** is a reference variable that is used to refer parent class constructors.

**4. What is abstract method?**

An **abstract method can** only set a visibility modifier, one of public or protected. **abstract** keyword is used to create a **abstract** class and **method.** An **abstract method can** only set a visibility modifier, one of public or protected.It should not be private.

**5. What is abstract class?**

A **class** that is declared using “**abstract**” keyword is known as **abstract class**.  A Java **abstract class** is a **class** which cannot be instantiated, which means we cannot create object for abstract classes.it contains abstract methods and also methods with body.

**6. What is the use of final modifier?**

 In **Java**, variable, method, and class can have a **final** non-**access modifier**. This keyword is used to make any class, method, or variable **final**. Once a **final** variable is initialized, you cannot change its value again ,methods cannot be overridden ,classes cannot be extended.

**7. What is interface? Write the syntax interface.**

An **interface** is just like Java Class, but it only has static constants and abstract method. All methods in an **interface** are implicitly public and abstract.

interface <interface\_name> {

// declare constant fields

// declare methods that abstract

// by default.

}

interface Player

{

    final int id = 10;

    int move();

}

**8. What is package?**

A **package** is a container of related classes and interfaces. It helps to organize your classes into a folder structure and make it easy to locate and use them.

**9. What is exception?**

**Exceptions** are the problems which can occur at runtime and compile time and stops the flow of a program. **Exceptions** are divided into two categories such as checked **exceptions** and unchecked **exceptions**. These exceptions are handled and continues the normal flow of execution.

**10. What is the use of finally block?**

A **finally block** is used to execute all the crucial statements that must be executed whether exception occurs or not.