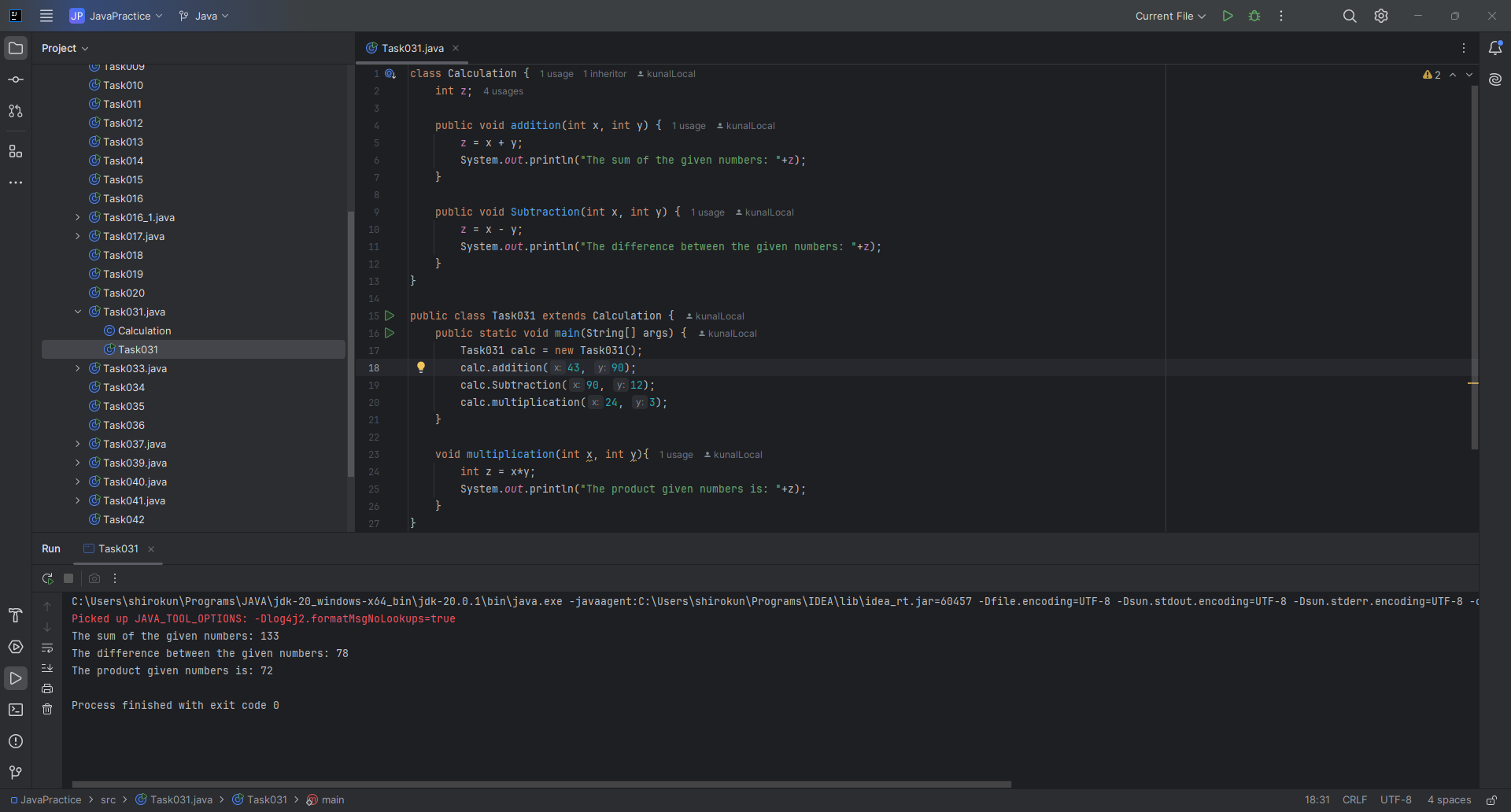
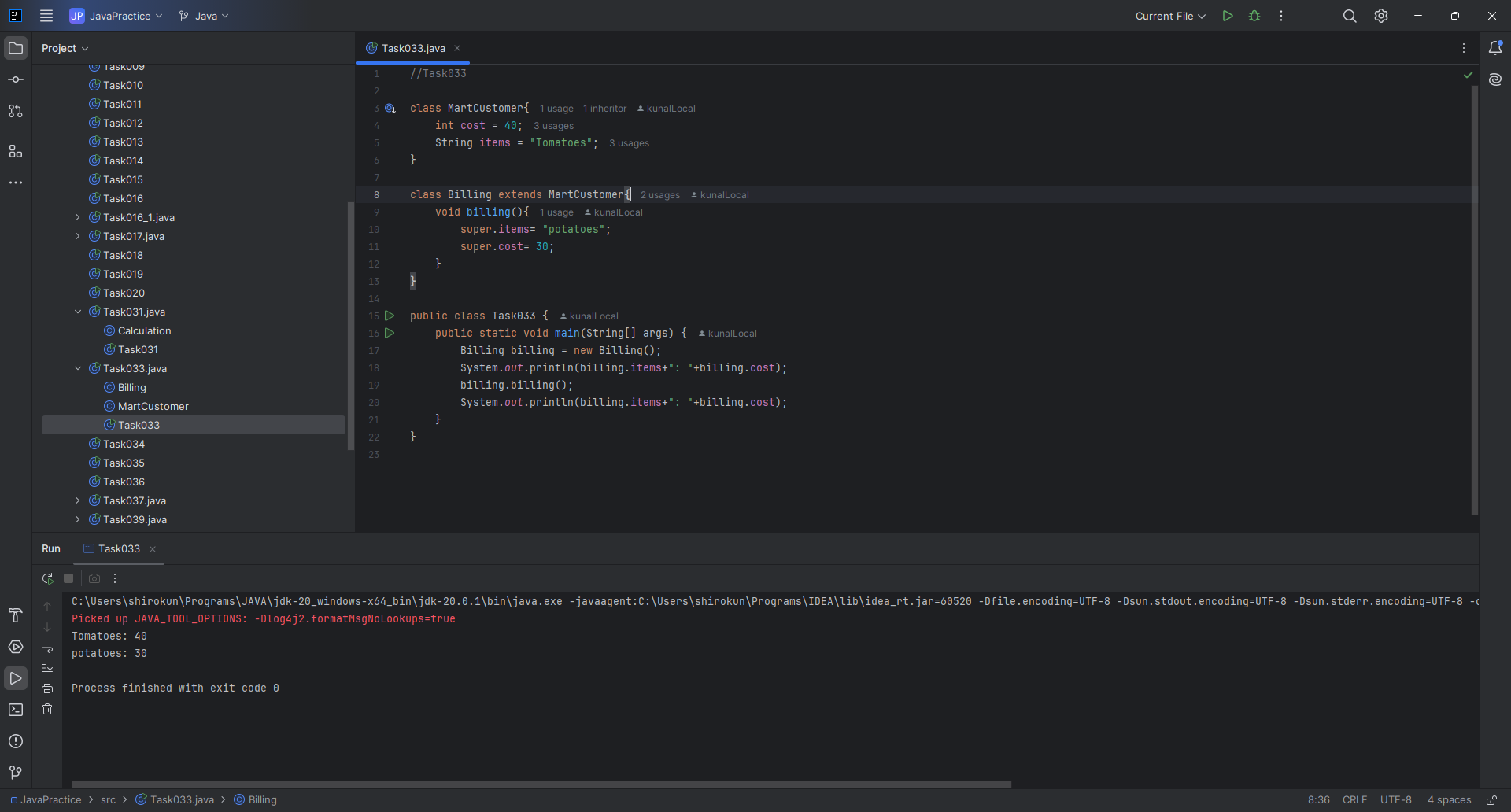
Day 8 – 07/06/2025

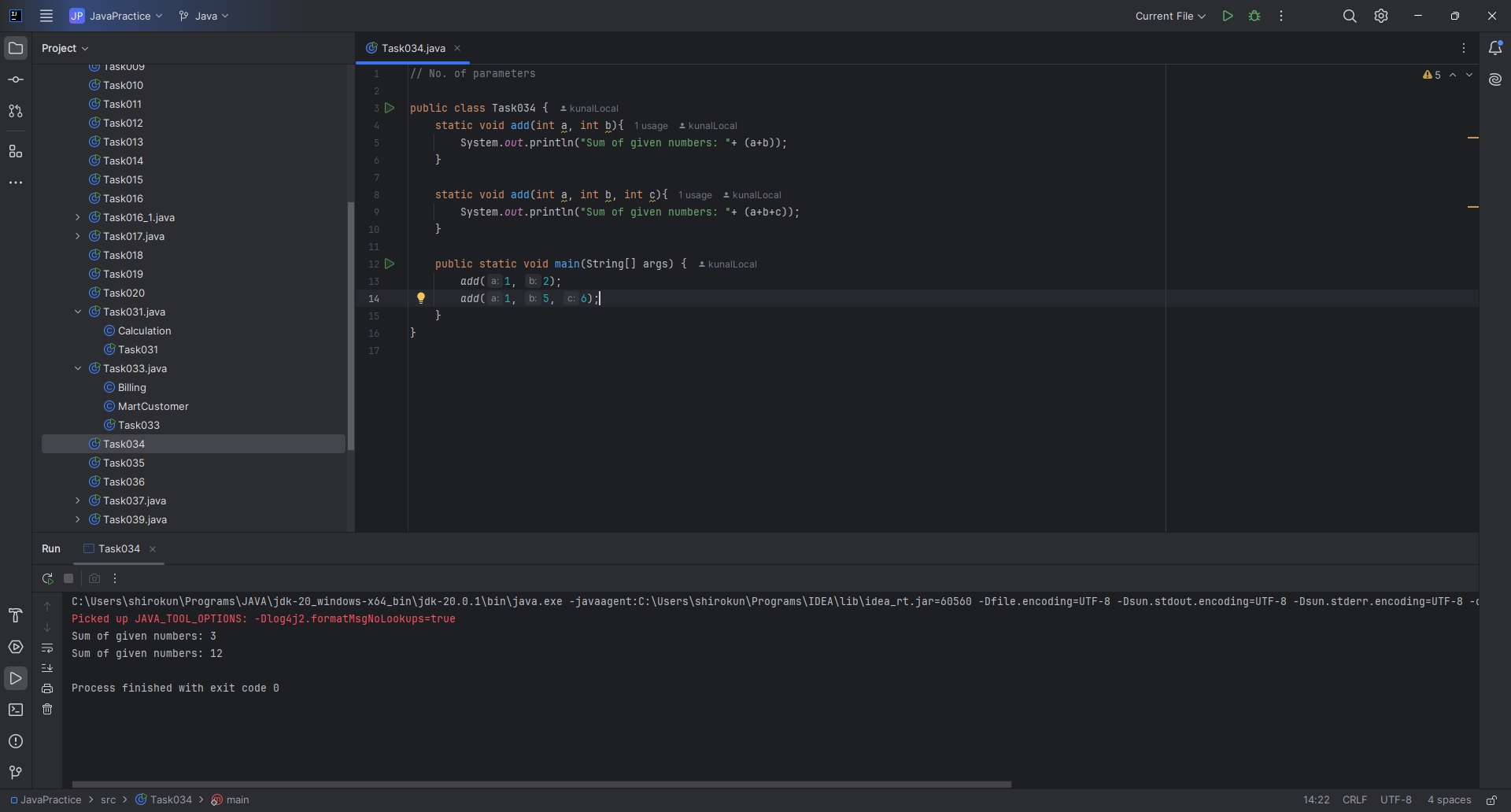
//Task031  
  
class Calculation {  
 int z;  
  
 public void addition(int x, int y) {  
 z = x + y;  
 System.*out*.println("The sum of the given numbers: "+z);  
 }  
  
 public void Subtraction(int x, int y) {  
 z = x - y;  
 System.*out*.println("The difference between the given numbers: "+z);  
 }  
}  
  
public class Task031 extends Calculation {  
 public static void main(String[] args) {  
 Task031 calc = new Task031();  
 calc.addition(43, 90);  
 calc.Subtraction(90, 12);  
 calc.multiplication(24, 3);  
 }  
  
 void multiplication(int x, int y){  
 int z = x\*y;  
 System.*out*.println("The product given numbers is: "+z);  
 }  
}



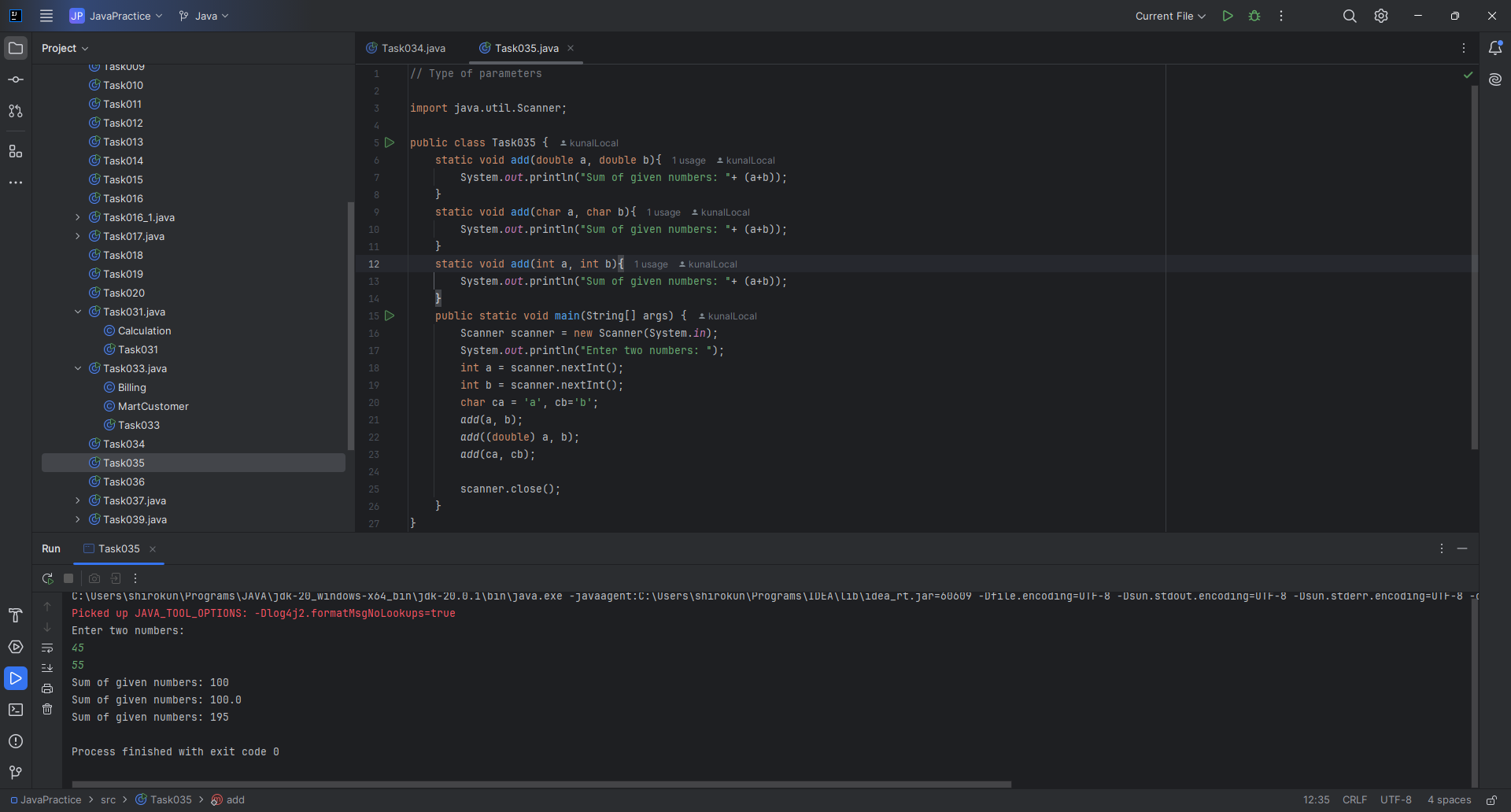
//Task033  
  
class MartCustomer{  
 int cost = 40;  
 String items = "Tomatoes";  
}  
  
class Billing extends MartCustomer{  
 void billing(){  
 super.items= "potatoes";  
 super.cost= 30;  
 }  
}  
  
public class Task033 {  
 public static void main(String[] args) {  
 Billing billing = new Billing();  
 System.*out*.println(billing.items+": "+billing.cost);  
 billing.billing();  
 System.*out*.println(billing.items+": "+billing.cost);  
 }  
}



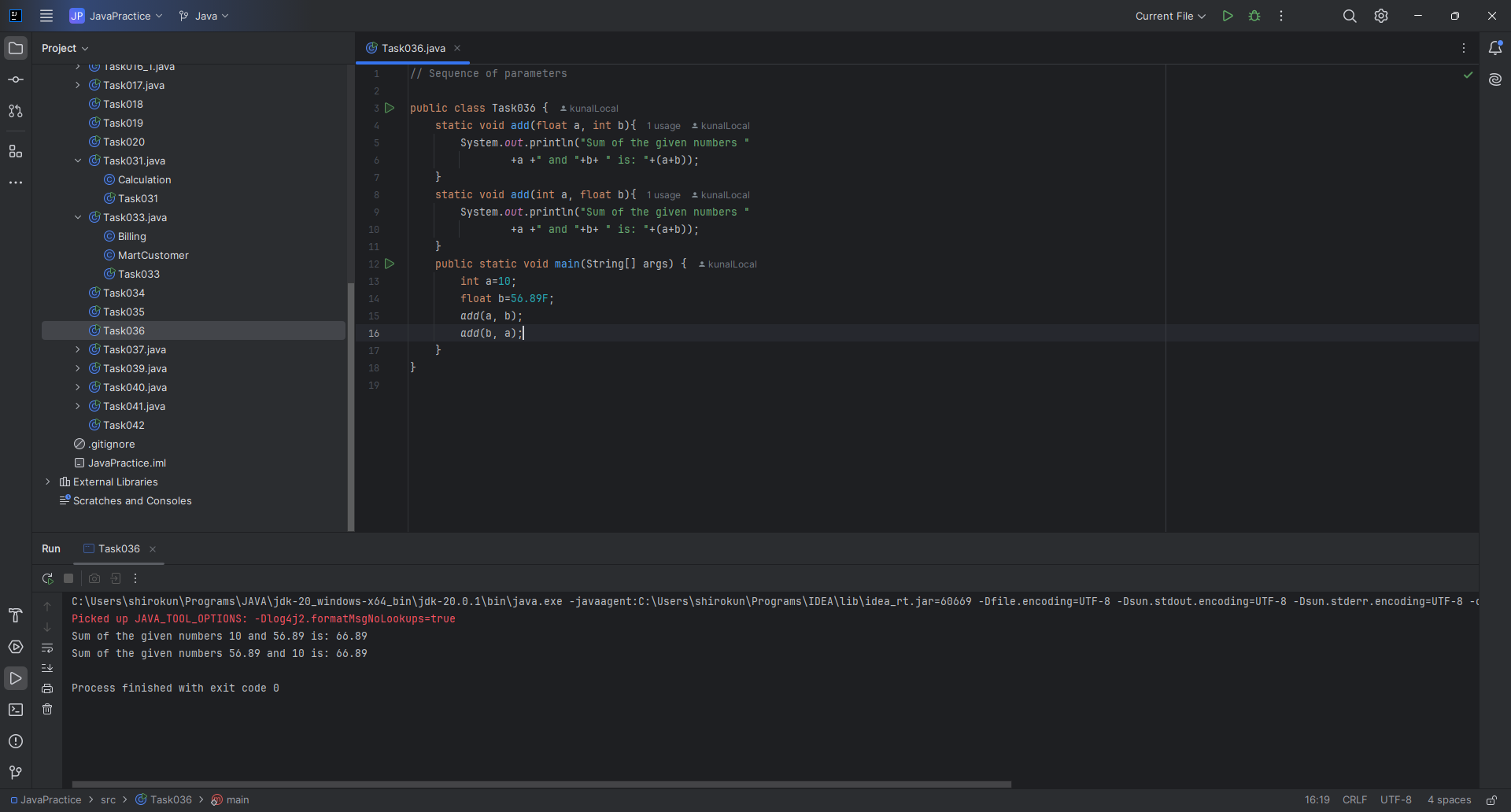
// Task034: No. of parameters  
  
public class Task034 {  
 static void add(int a, int b){  
 System.*out*.println("Sum of given numbers: "+ (a+b));  
 }  
  
 static void add(int a, int b, int c){  
 System.*out*.println("Sum of given numbers: "+ (a+b+c));  
 }  
  
 public static void main(String[] args) {  
 *add*(1, 2);  
 *add*(1, 5, 6);  
 }  
}



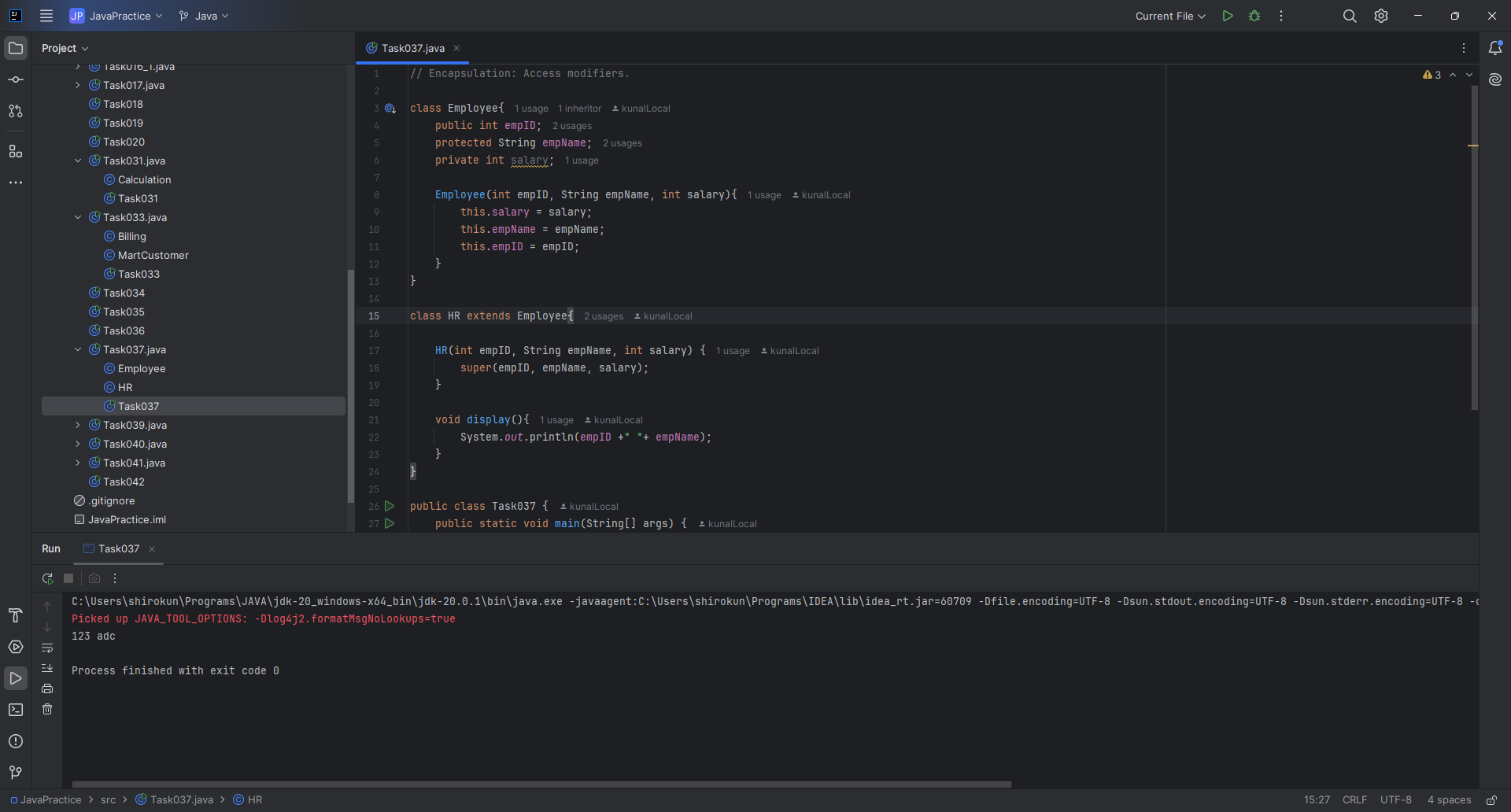
// Task035: Type of parameters  
  
import java.util.Scanner;  
  
public class Task035 {  
 static void add(double a, double b){  
 System.*out*.println("Sum of given numbers: "+ (a+b));  
 }  
 static void add(char a, char b){  
 System.*out*.println("Sum of given numbers: "+ (a+b));  
 }  
 static void add(int a, int b){  
 System.*out*.println("Sum of given numbers: "+ (a+b));  
 }  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
 System.*out*.println("Enter two numbers: ");  
 int a = scanner.nextInt();  
 int b = scanner.nextInt();  
 char ca = 'a', cb='b';  
 *add*(a, b);  
 *add*((double) a, b);  
 *add*(ca, cb);  
  
 scanner.close();  
 }  
}



// Task036: Sequence of parameters  
  
public class Task036 {  
 static void add(float a, int b){  
 System.*out*.println("Sum of the given numbers "  
 +a +" and "+b+ " is: "+(a+b));  
 }  
 static void add(int a, float b){  
 System.*out*.println("Sum of the given numbers "  
 +a +" and "+b+ " is: "+(a+b));  
 }  
 public static void main(String[] args) {  
 int a=10;  
 float b=56.89F;  
 *add*(a, b);  
 *add*(b, a);  
 }  
}

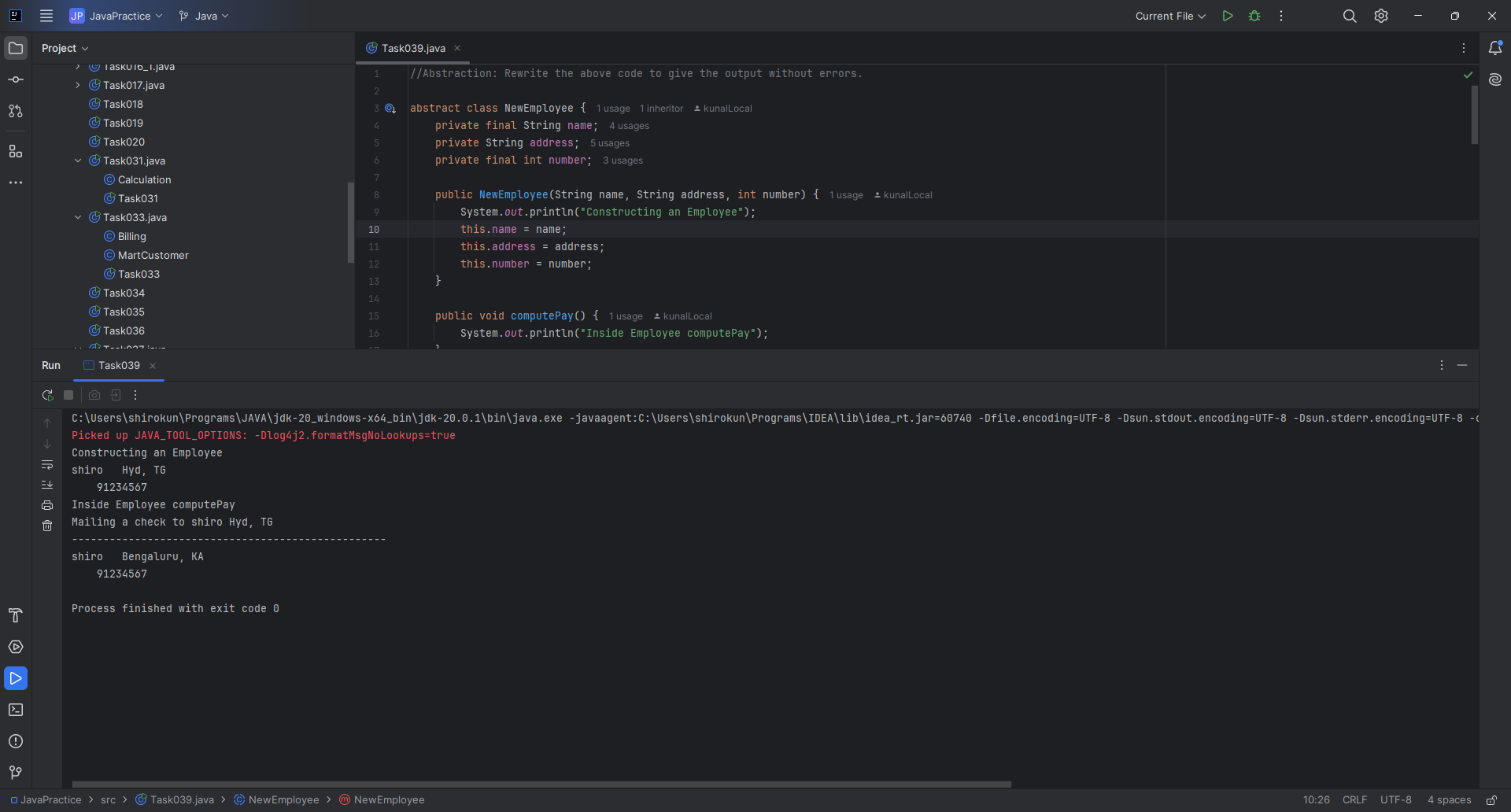


// Task037: Encapsulation: Access modifiers.  
  
class Employee{  
 public int empID;  
 protected String empName;  
 private int salary;  
  
 Employee(int empID, String empName, int salary){  
 this.salary = salary;  
 this.empName = empName;  
 this.empID = empID;  
 }  
}  
  
class HR extends Employee{  
  
 HR(int empID, String empName, int salary) {  
 super(empID, empName, salary);  
 }  
  
 void display(){  
 System.*out*.println(empID +" "+ empName);  
 }  
}  
  
public class Task037 {  
 public static void main(String[] args) {  
 HR emp1 = new HR(123, "adc", 90000);  
 emp1.display();  
 }  
}



//Abstraction: Rewrite the above code to give the output without errors.  
  
abstract class NewEmployee {  
 private final String name;  
 private String address;  
 private final int number;  
  
 public NewEmployee(String name, String address, int number) {  
 System.*out*.println("Constructing an Employee");  
 this.name = name;  
 this.address = address;  
 this.number = number;  
 }  
  
 public void computePay() {  
 System.*out*.println("Inside Employee computePay");  
 }  
  
 public void mailCheck() {  
 System.*out*.println("Mailing a check to " + this.name + " " + this.address);  
 }  
  
 public String toString() {  
 return name + " " + address + " " + number;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public String getAddress() {  
 return address;  
 }  
  
 public void setAddress(String newAddress) {  
 address = newAddress;  
 }  
  
 public int getNumber() {  
 return number;  
 }  
}  
  
class conEmployee extends NewEmployee{  
  
 public conEmployee(String name, String address, int number) {  
 super(name, address, number);  
 }  
}

public class Task039 {  
 public static void main(String[] args) {  
 String newEmpName = "shiro";  
 int mobileNum = 91234567;  
 String address = "Hyd, TG";  
 conEmployee employee = new conEmployee(newEmpName, address, mobileNum);  
 System.*out*.println(employee.getName()+"\t"+employee.getAddress()+"\n\t"+  
 employee.getNumber());  
 employee.computePay();  
 employee.mailCheck();  
 System.*out*.println("-".repeat(50));  
 employee.setAddress("Bengaluru, KA");  
 System.*out*.println(employee.getName()+"\t"+employee.getAddress()+"\n\t"+  
 employee.getNumber());  
 }  
}

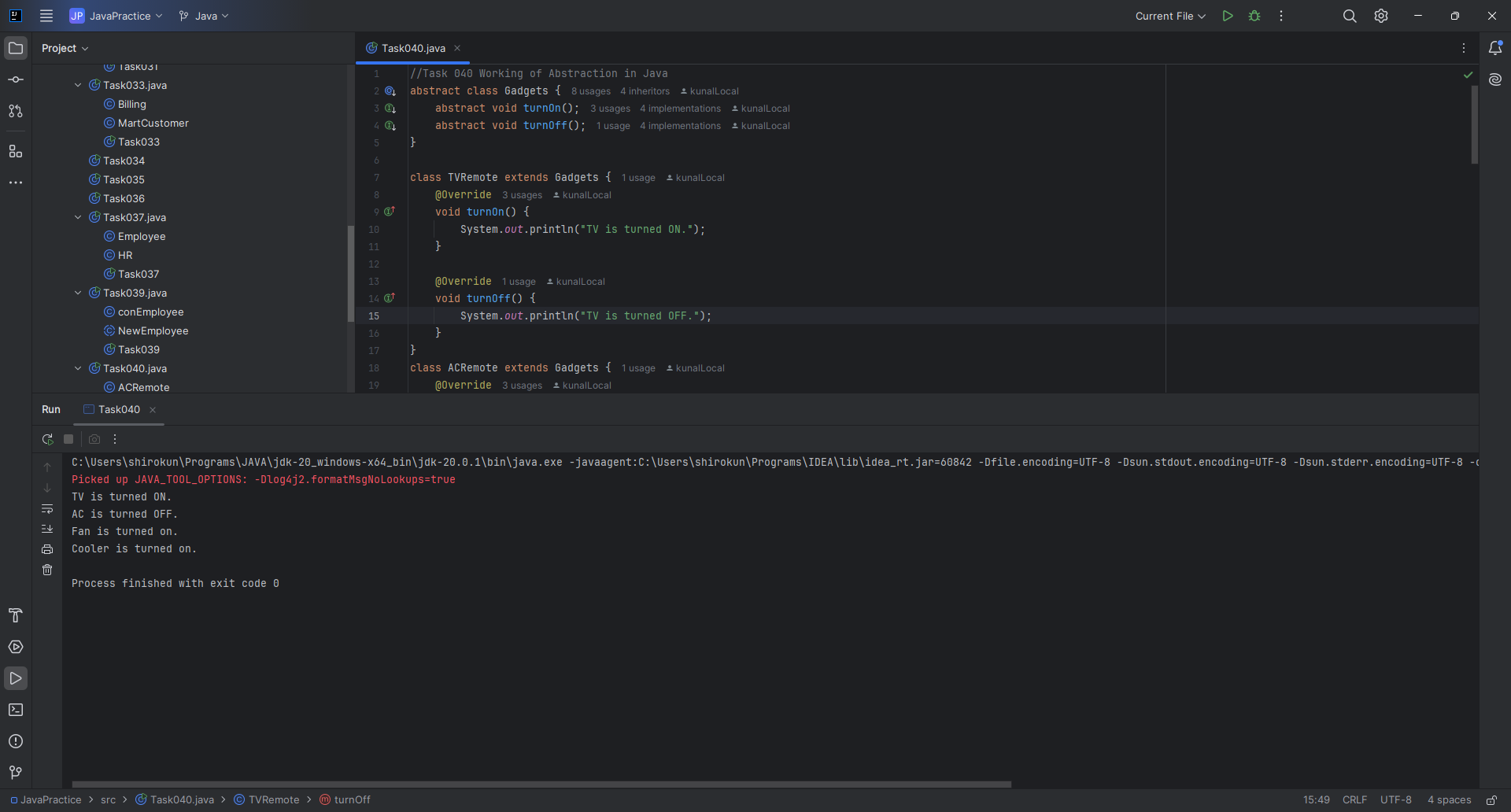


//Task 040 Working of Abstraction in Java

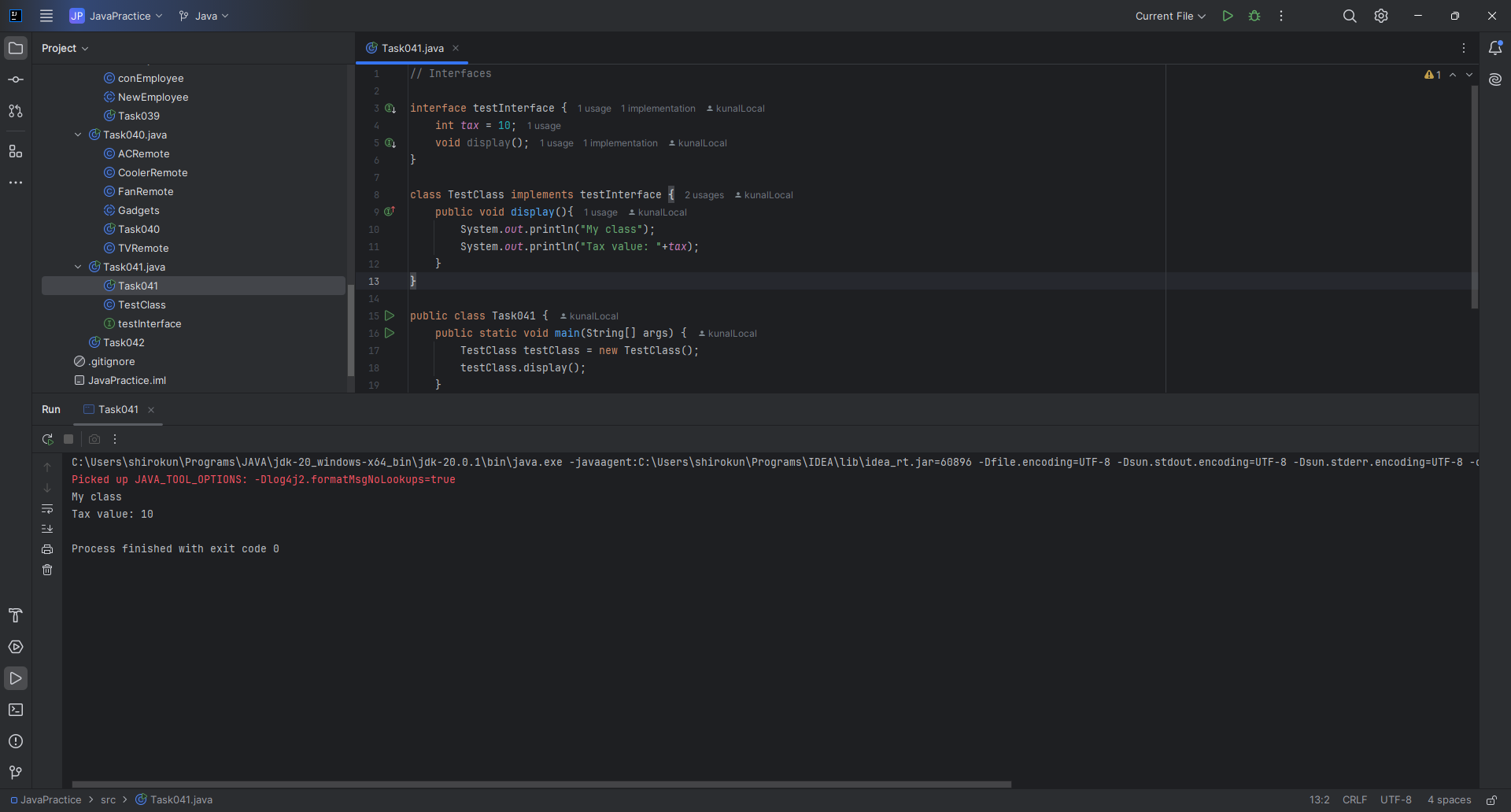
abstract class Gadgets {  
 abstract void turnOn();  
 abstract void turnOff();  
}  
  
class TVRemote extends Gadgets {

@Override  
 void turnOn() {  
 System.*out*.println("TV is turned ON.");  
 }  
  
 @Override  
 void turnOff() {  
 System.*out*.println("TV is turned OFF.");  
 }  
}

class ACRemote extends Gadgets {  
 @Override  
 void turnOn() {  
 System.*out*.println("AC is turned ON.");  
 }  
  
 @Override  
 void turnOff() {  
 System.*out*.println("AC is turned OFF.");  
 }  
}  
  
class FanRemote extends Gadgets{  
  
 @Override  
 void turnOn() {  
 System.*out*.println("Fan is turned on.");  
 }  
  
 @Override  
 void turnOff() {  
 System.*out*.println("Fan is turned off.");  
 }  
}  
  
class CoolerRemote extends Gadgets{  
  
 @Override  
 void turnOn() {  
 System.*out*.println("Cooler is turned on.");  
 }  
  
 @Override  
 void turnOff() {  
 System.*out*.println("Cooler is turned off.");  
 }  
}  
  
public class Task040 {  
 public static void main(String[] args) {  
 Gadgets acRemote = new ACRemote();  
 Gadgets tvRemote = new TVRemote();  
 tvRemote.turnOn();  
 acRemote.turnOff();  
 Gadgets fanRemote = new FanRemote();  
 Gadgets coolerRemote = new CoolerRemote();  
 fanRemote.turnOn();  
 coolerRemote.turnOn();  
 }  
}

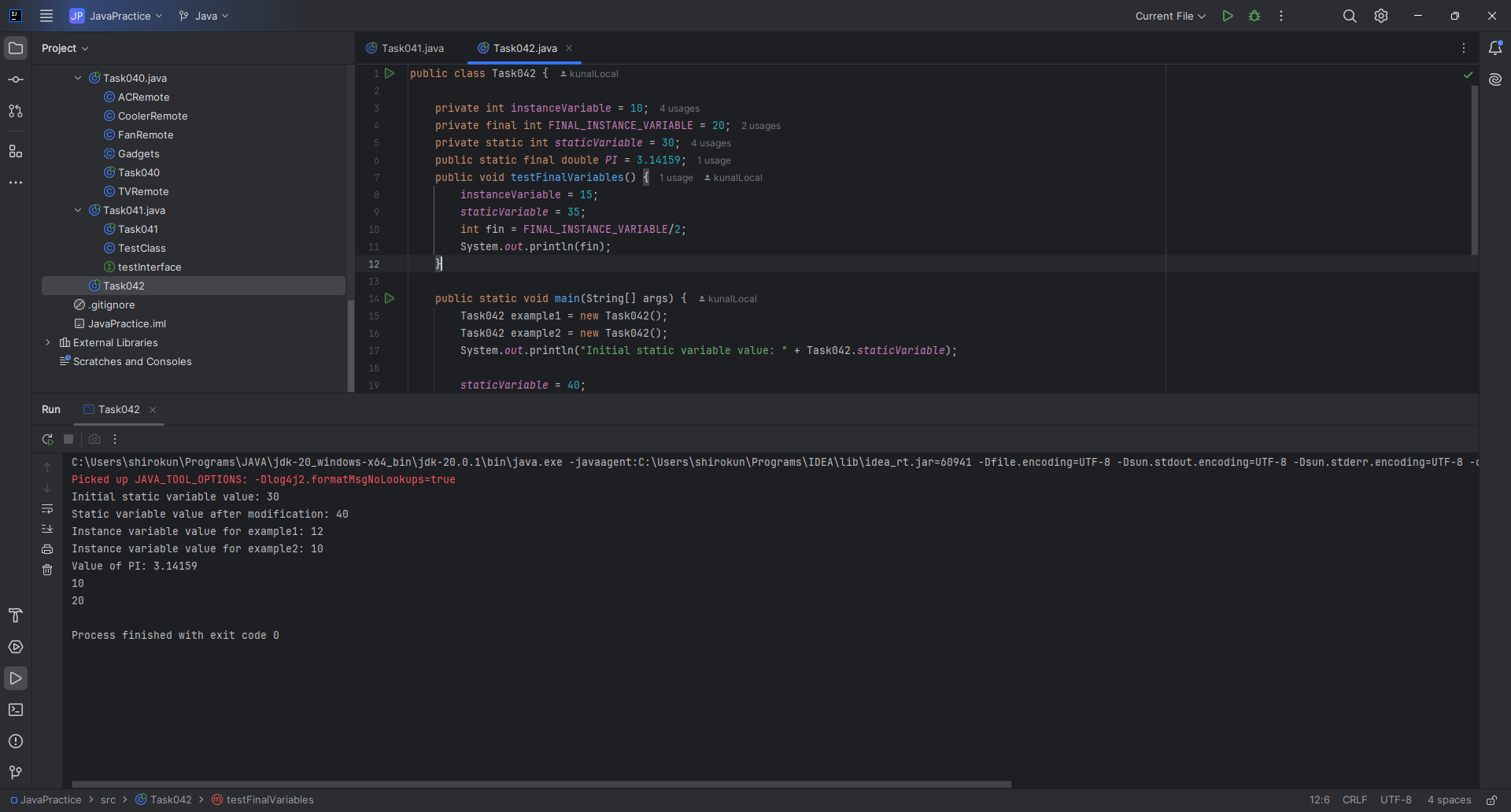


// Task041: Interfaces  
  
interface testInterface {  
 int *tax* = 10;  
 void display();  
}  
  
class TestClass implements testInterface {  
 public void display(){  
 System.*out*.println("My class");  
 System.*out*.println("Tax value: "+*tax*);  
 }  
}  
  
public class Task041 {  
 public static void main(String[] args) {  
 TestClass testClass = new TestClass();  
 testClass.display();  
 }  
}



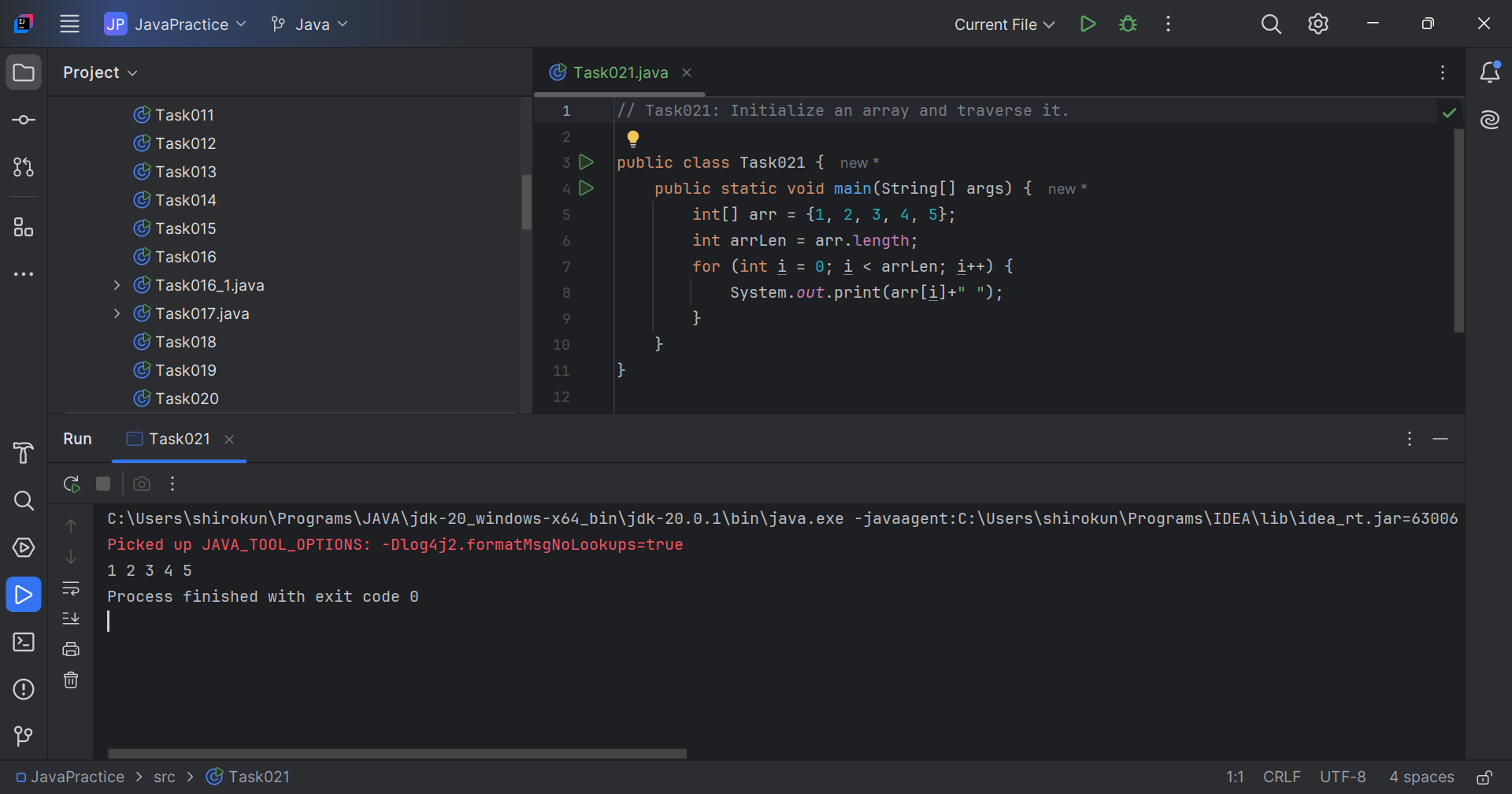
// Task042: Final keyword.

public class Task042 {  
  
 private int instanceVariable = 10;  
 private final int FINAL\_INSTANCE\_VARIABLE = 20;  
 private static int *staticVariable* = 30;  
 public static final double *PI* = 3.14159;  
 public void testFinalVariables() {  
 instanceVariable = 15;  
 *staticVariable* = 35;  
 int fin = FINAL\_INSTANCE\_VARIABLE/2;  
 System.*out*.println(fin);  
 }  
  
 public static void main(String[] args) {  
 Task042 example1 = new Task042();  
 Task042 example2 = new Task042();  
 System.*out*.println("Initial static variable value: " + Task042.*staticVariable*);  
  
 *staticVariable* = 40;  
 System.*out*.println("Static variable value after modification: " + *staticVariable*);  
 example1.instanceVariable = 12;  
 System.*out*.println("Instance variable value for example1: " + example1.instanceVariable);  
 System.*out*.println("Instance variable value for example2: " + example2.instanceVariable);  
 System.*out*.println("Value of PI: " + Task042.*PI*);  
 example2.testFinalVariables();  
 System.*out*.println(example2.FINAL\_INSTANCE\_VARIABLE);  
 }  
}

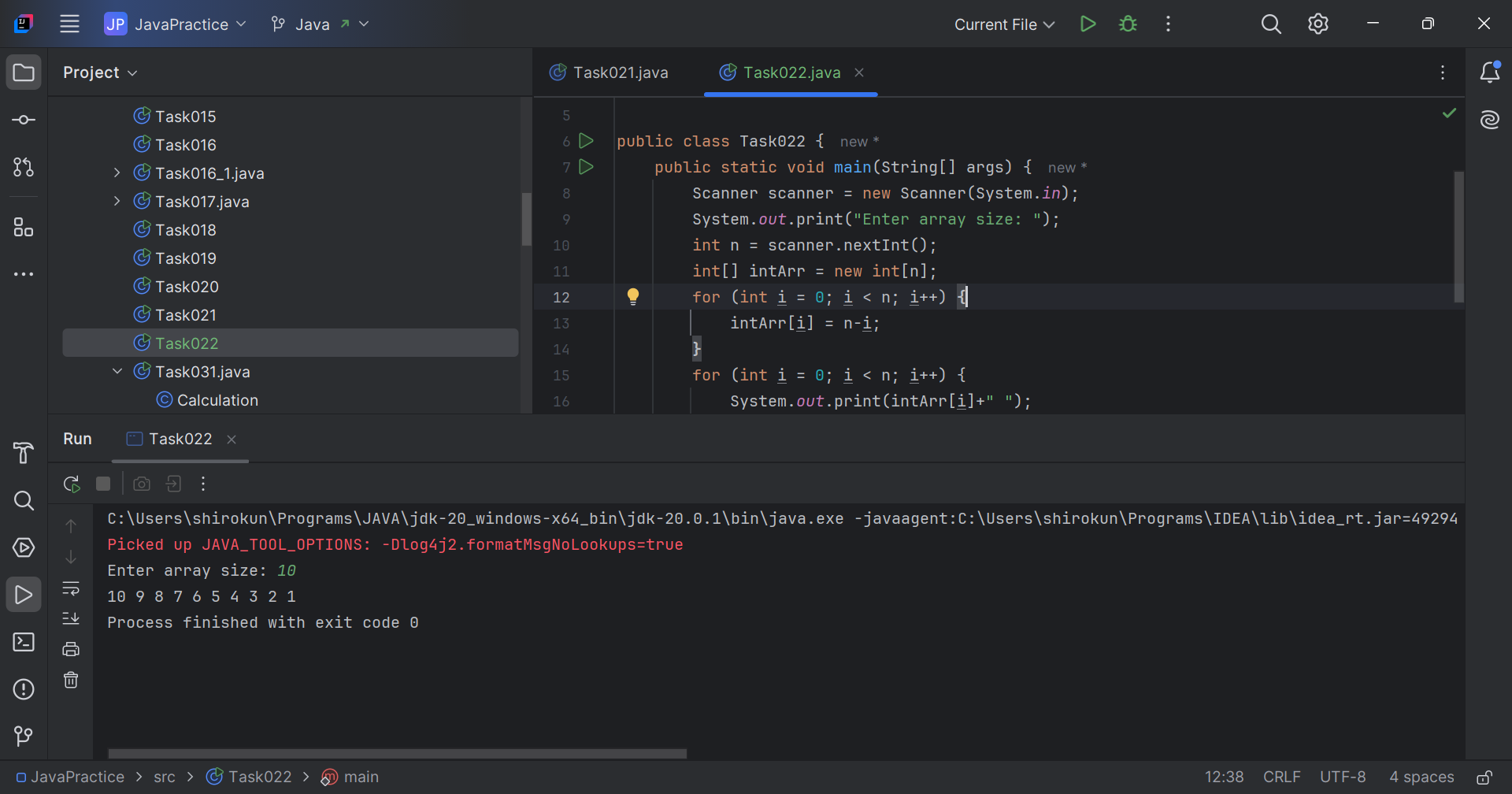


Home works tasks:

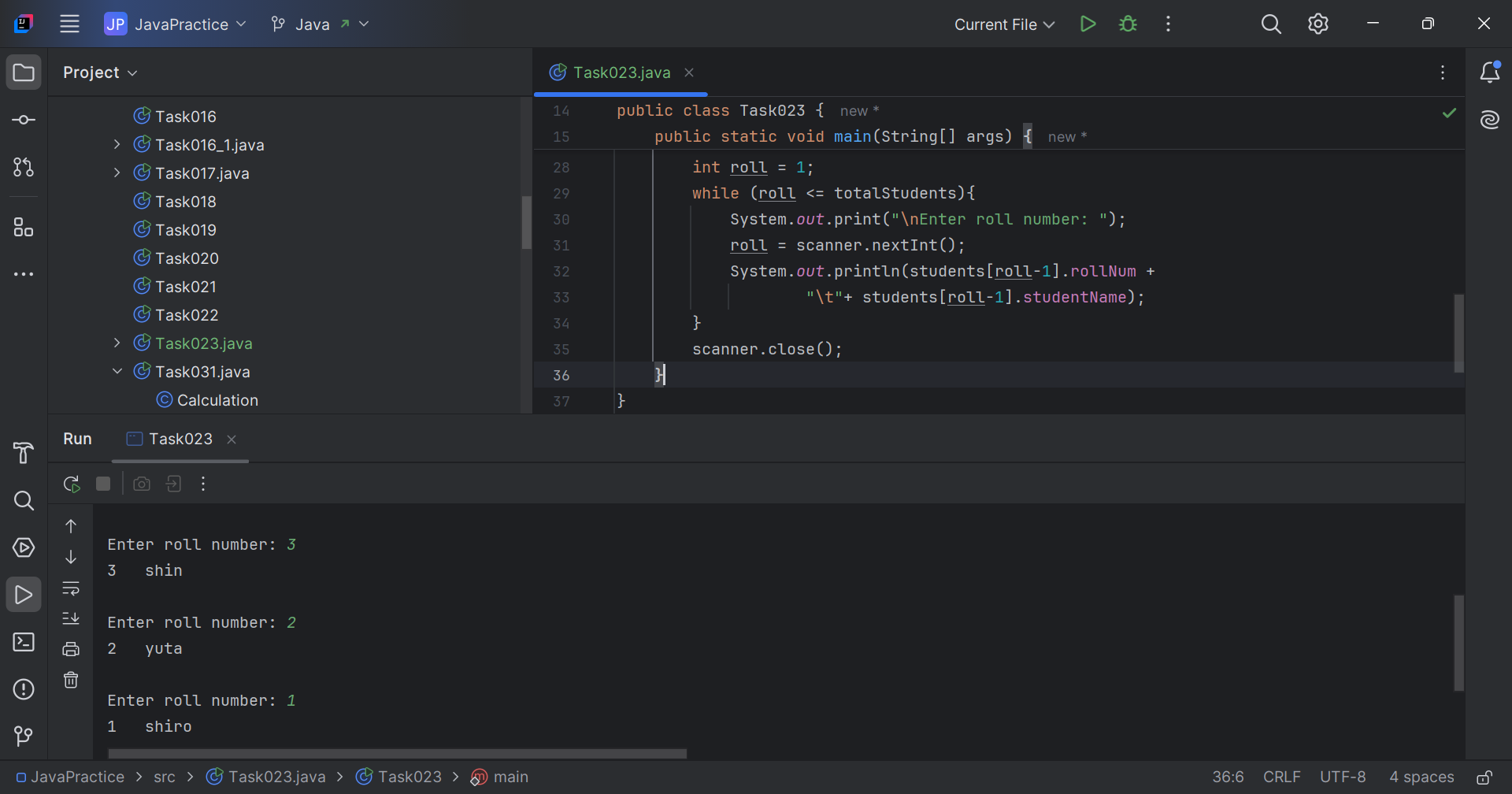
// Task021: Initialize an array and traverse it.  
  
public class Task021 {  
 public static void main(String[] args) {  
 int[] arr = {1, 2, 3, 4, 5};  
 int arrLen = arr.length;  
 for (int i = 0; i < arrLen; i++) {  
 System.*out*.print(arr[i]+" ");  
 }  
 }  
}



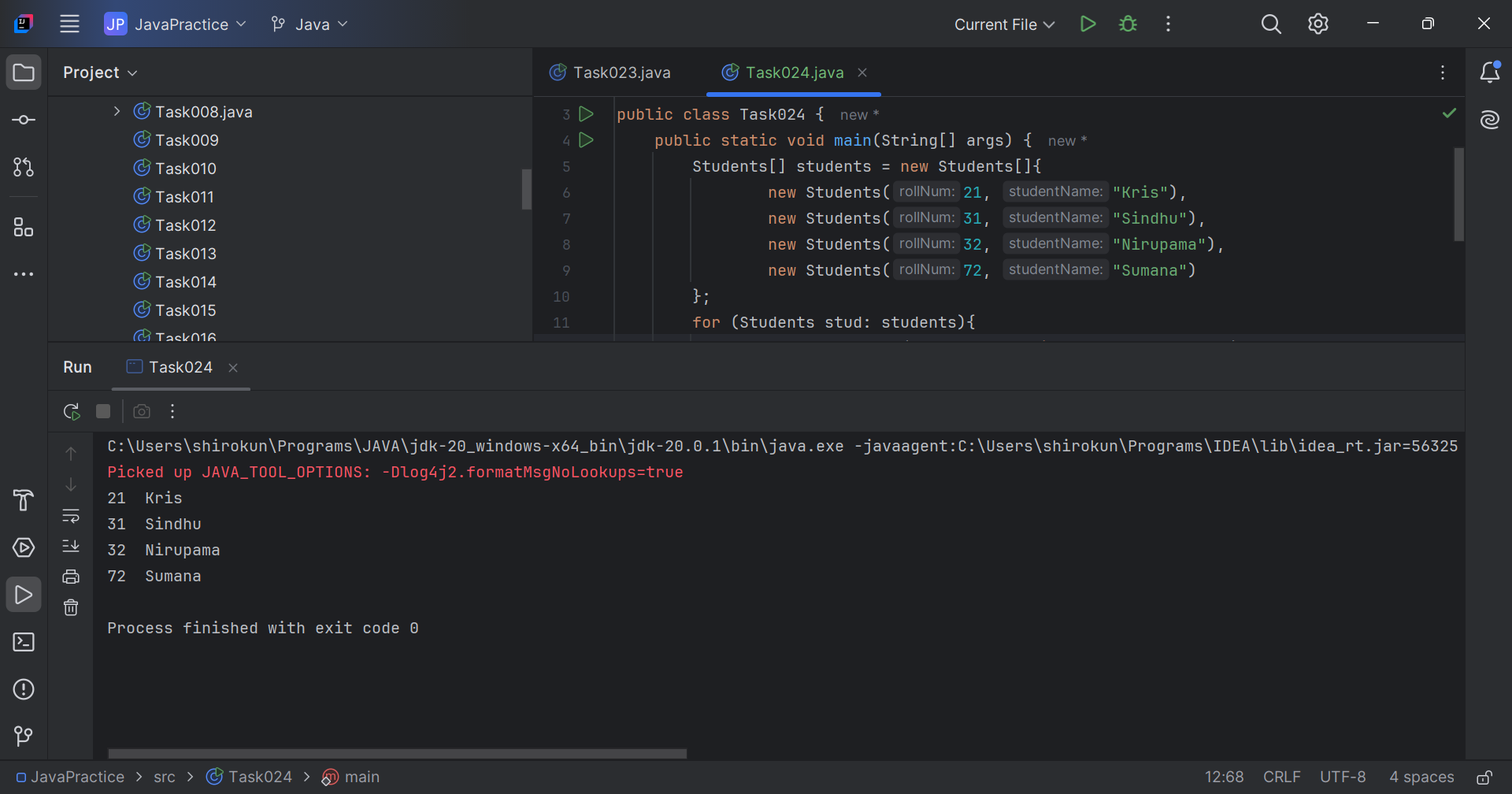
// Task 022: Create an array of integers,  
// read values into the array and print.  
  
import java.util.Scanner;  
  
public class Task022 {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
 System.*out*.print("Enter array size: ");  
 int n = scanner.nextInt();  
 int[] intArr = new int[n];  
 for (int i = 0; i < n; i++) {  
 intArr[i] = n-i;  
 }  
 for (int i = 0; i < n; i++) {  
 System.*out*.print(intArr[i]+" ");  
 }  
 scanner.close();  
 }  
}



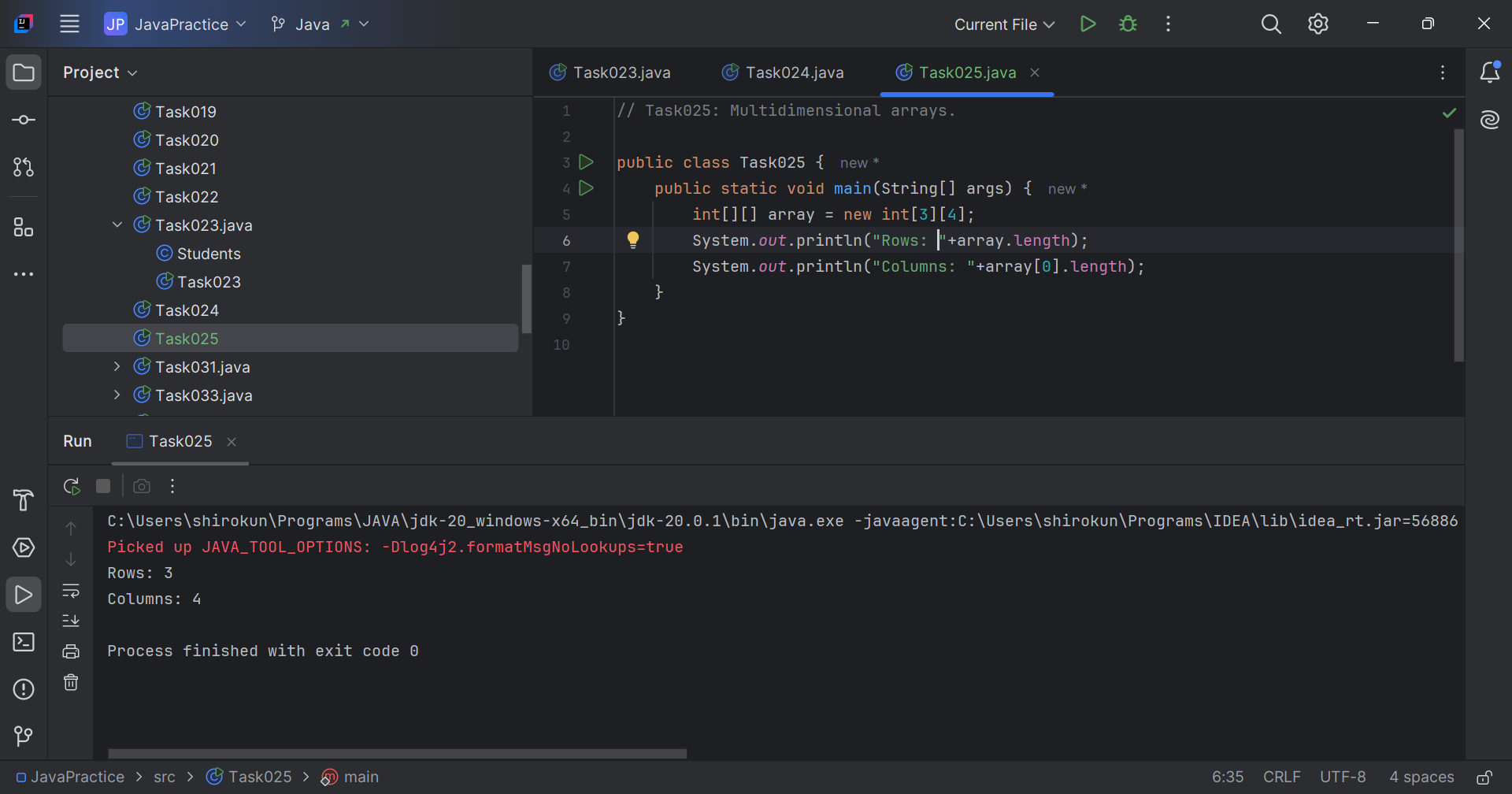
// Task023: Illustrate creating an array of objects.  
  
import java.util.Scanner;  
  
class Students{  
 public String studentName;  
 public int rollNum;  
 Students(int rollNum, String studentName){  
 this.studentName = studentName;  
 this.rollNum = rollNum;  
 }  
}  
public class Task023 {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
 System.*out*.print("Enter no. of students: ");  
 int totalStudents = scanner.nextInt();  
 Students[] students = new Students[totalStudents];  
 for (int i = 0; i < totalStudents; i++) {  
 System.*out*.print("Roll no: ");  
 int roll = scanner.nextInt();  
 System.*out*.print("Name: ");  
 String name = scanner.next();  
 students[i] = new Students(roll, name);  
 }  
 int roll = 1;  
 while (roll <= totalStudents){  
 System.*out*.print("\nEnter roll number: ");  
 roll = scanner.nextInt();  
 System.*out*.println(students[roll-1].rollNum +  
 "\t"+ students[roll-1].studentName);  
 }  
 scanner.close();  
 }  
}



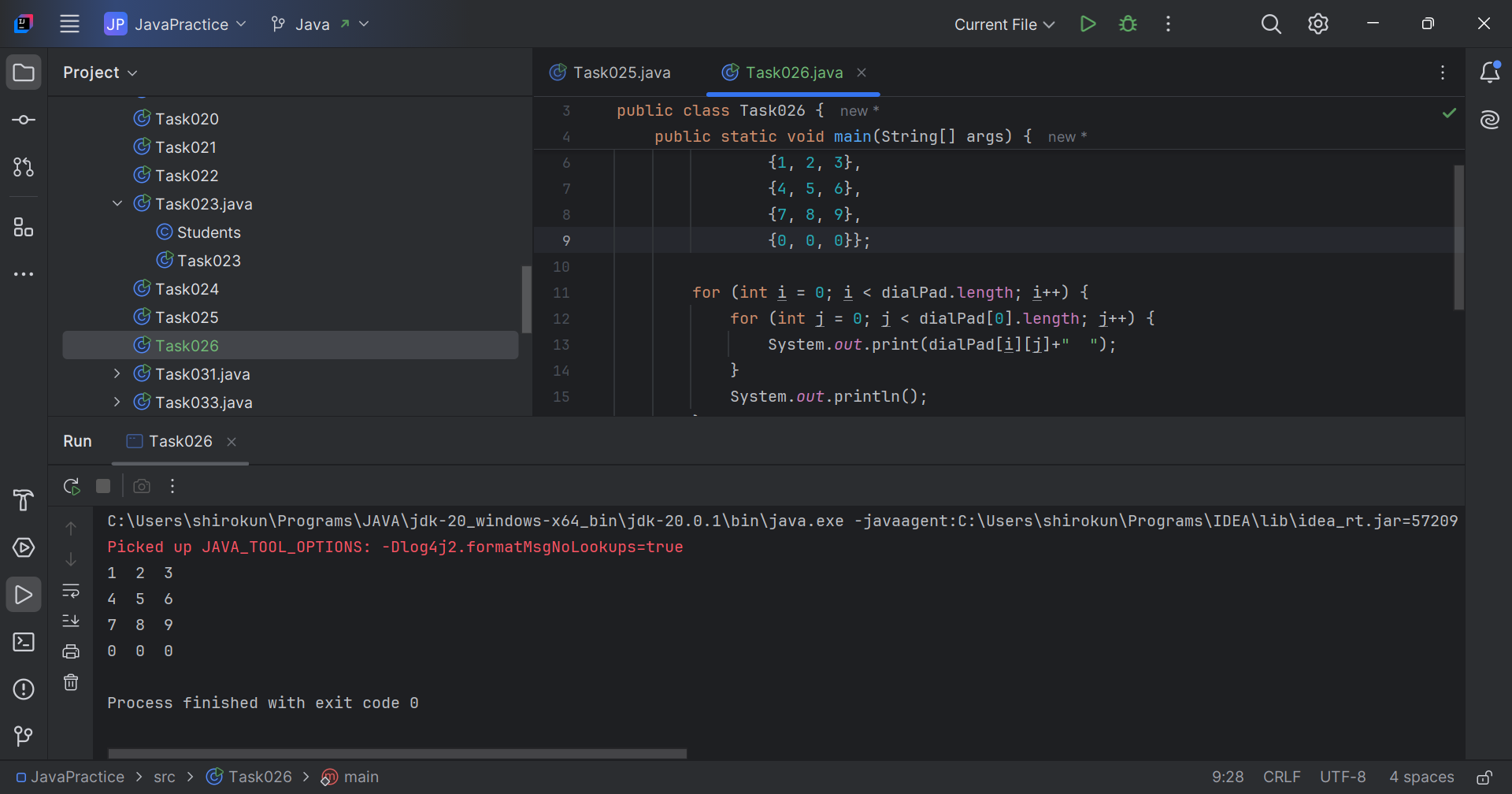
// Task024: Another way of creating array of objects.  
  
public class Task024 {  
 public static void main(String[] args) {  
 Students[] students = new Students[]{  
 new Students(21, "Kris"),  
 new Students(31, "Sindhu"),  
 new Students(32, "Nirupama"),  
 new Students(72, "Sumana")  
 };  
 for (Students stud: students){  
 System.*out*.println(stud.rollNum+"\t"+stud.studentName);  
 }  
 }  
}



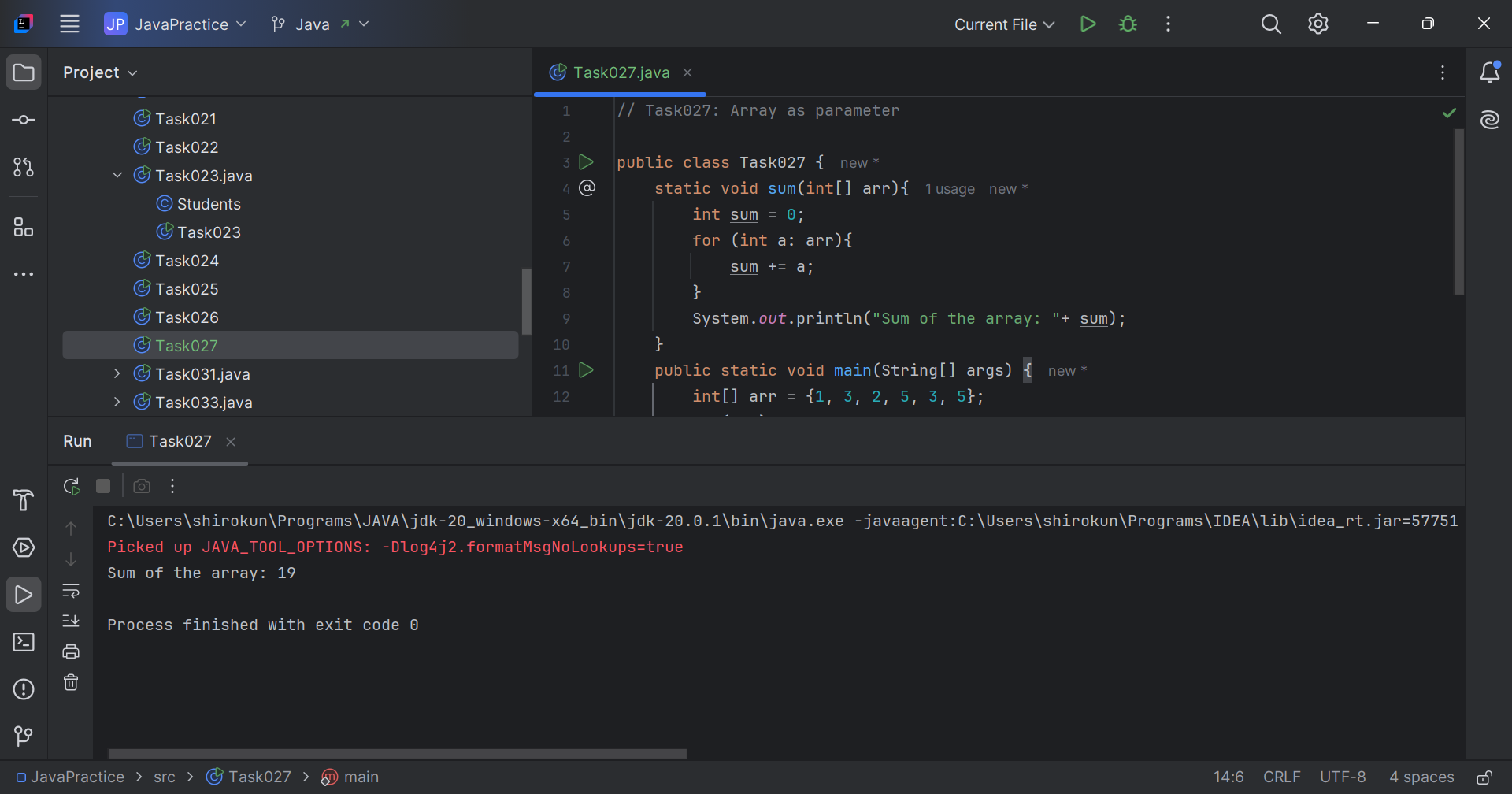
// Task025: Multidimensional arrays.  
  
public class Task025 {  
 public static void main(String[] args) {  
 int[][] array = new int[3][4];  
 System.*out*.println("Rows: "+array.length);  
 System.*out*.println("Columns: "+array[0].length);  
 }  
}



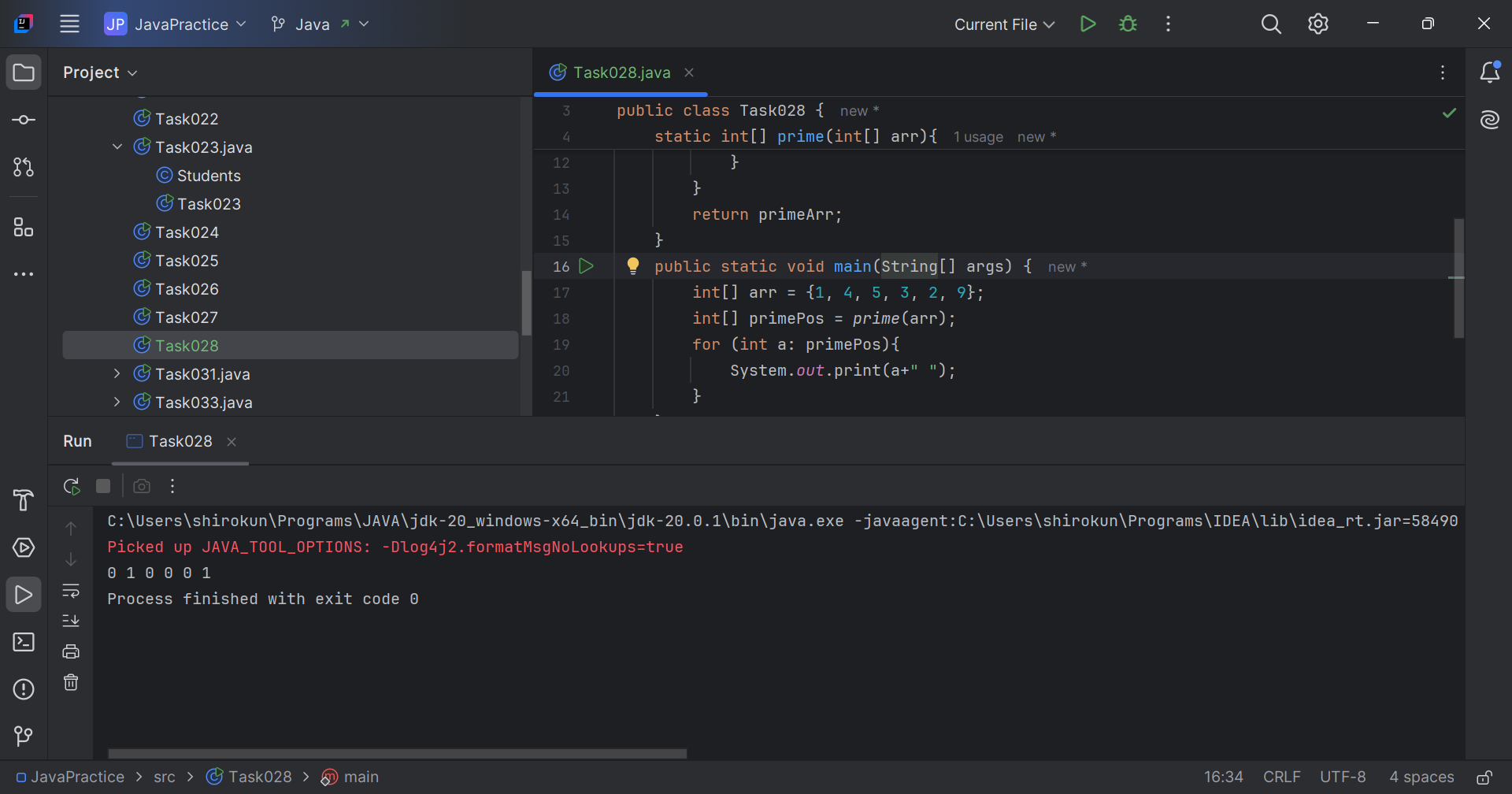
// Task026: Traverse multidimensional array.  
  
public class Task026 {  
 public static void main(String[] args) {  
 int[][] dialPad = {  
 {1, 2, 3},  
 {4, 5, 6},  
 {7, 8, 9},  
 {0, 0, 0}};  
  
 for (int i = 0; i < dialPad.length; i++) {  
 for (int j = 0; j < dialPad[0].length; j++) {  
 System.*out*.print(dialPad[i][j]+" ");  
 }  
 System.*out*.println();  
 }  
 }  
}



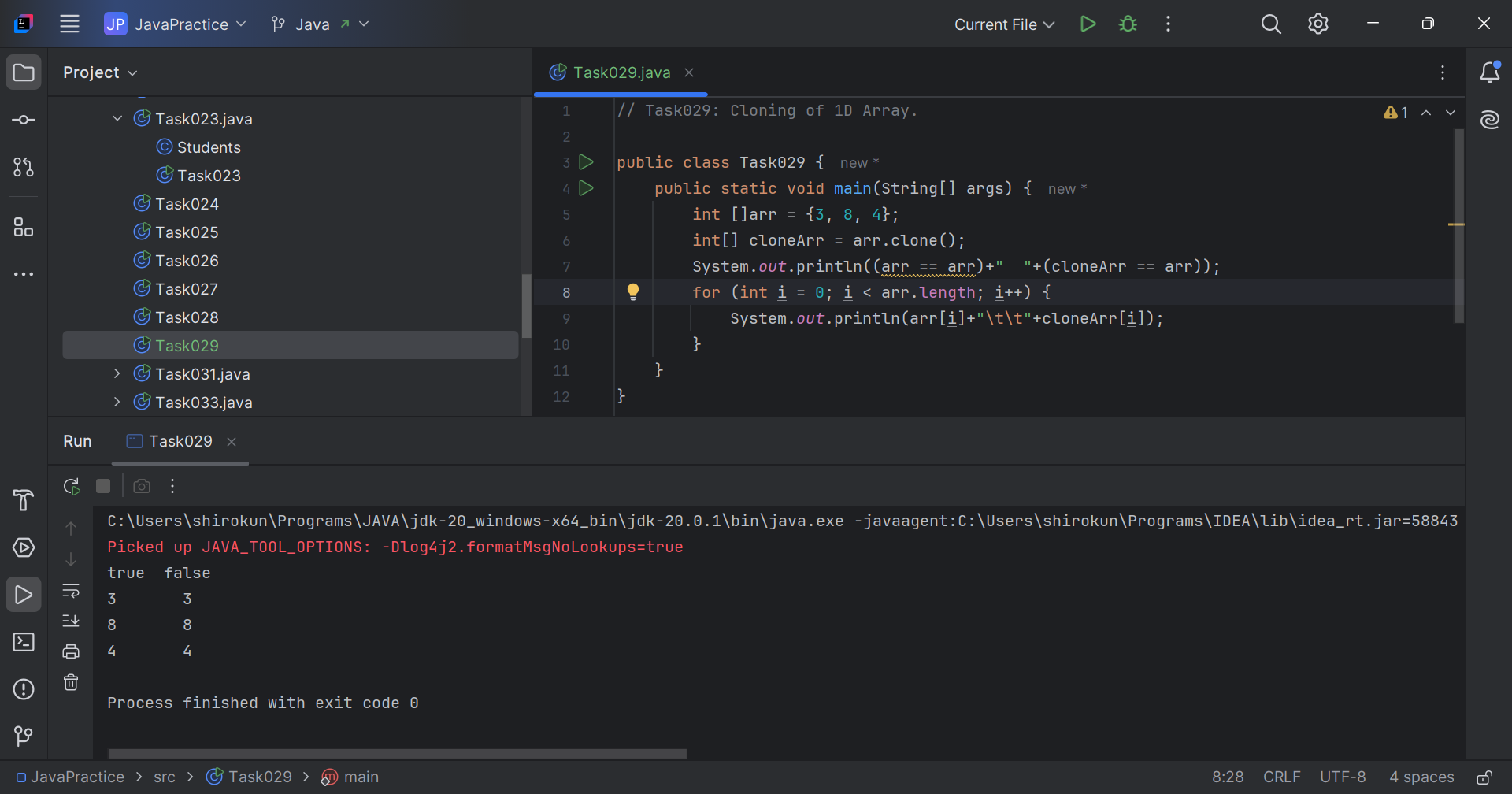
// Task027: Array as parameter  
  
public class Task027 {  
 static void sum(int[] arr){  
 int sum = 0;  
 for (int a: arr){  
 sum += a;  
 }  
 System.*out*.println("Sum of the array: "+ sum);  
 }  
 public static void main(String[] args) {  
 int[] arr = {1, 3, 2, 5, 3, 5};  
 *sum*(arr);  
 }  
}



// Task028: Array return type.  
  
public class Task028 {  
 static int[] prime(int[] arr){  
 int[] primeArr = new int[arr.length];  
 for (int j=0; j<arr.length; j++){  
 for (int i = 2; i <= arr[j]/2; i++) {  
 if (arr[j]%i == 0){  
 primeArr[j] = 1;  
 break;  
 }  
 }  
 }  
 return primeArr;  
 }  
 public static void main(String[] args) {  
 int[] arr = {1, 4, 5, 3, 2, 9};  
 int[] primePos = *prime*(arr);  
 for (int a: primePos){  
 System.*out*.print(a+" ");  
 }  
 }  
}



// Task029: Cloning of 1D Array.  
  
public class Task029 {  
 public static void main(String[] args) {  
 int []arr = {3, 8, 4};  
 int[] cloneArr = arr.clone();  
 System.*out*.println((arr == arr)+" "+(cloneArr == arr));  
 for (int i = 0; i < arr.length; i++) {  
 System.*out*.println(arr[i]+"\t\t"+cloneArr[i]);  
 }  
 }  
}



// Task030: Cloning 2D array.  
  
public class Task030 {  
 public static void main(String[] args) {  
 int[][] arr = {  
 {1, 2, 3},  
 {4, 5, 6},  
 {7, 8, 9}};  
 int[][] cloneArr = arr.clone();  
 System.*out*.println(arr == cloneArr);  
 System.*out*.println(arr[0] == cloneArr[0]);  
 }  
}

