Day 8 , Emp\_ID : 112916146

Task : Enum , gettters and setters (encapsulation), Arrays, (pending from yesterday)

Enum

package JavaTraining;

import java.util.HashMap;

import java.util.Map;

public enum Element {

H("Hydrogen", 1, 1.008f),

HE("Helium", 2, 4.0026f),

// ...

NE("Neon", 10, 20.180f);

private static final Map<String, Element> BY\_LABEL = new HashMap<>();

private static final Map<Integer, Element> BY\_ATOMIC\_NUMBER = new HashMap<>();

private static final Map<Float, Element> BY\_ATOMIC\_WEIGHT = new HashMap<>();

static {

for (Element e : values()) { //for each loop

BY\_LABEL.put(e.label, e);

BY\_ATOMIC\_NUMBER.put(e.atomicNumber, e);

BY\_ATOMIC\_WEIGHT.put(e.atomicWeight, e);

}

}

public final String label;

public final int atomicNumber;

public final float atomicWeight;

private Element(String label, int atomicNumber, float atomicWeight) {

this.label = label;

this.atomicNumber = atomicNumber;

this.atomicWeight = atomicWeight;

}

public static Element valueOfLabel(String label) {

return BY\_LABEL.get(label);

}

public static Element valueOfAtomicNumber(int number) {

return BY\_ATOMIC\_NUMBER.get(number);

}

public static Element valueOfAtomicWeight(float weight) {

return BY\_ATOMIC\_WEIGHT.get(weight);

}

public static void main(String[] args) {

System.out.println("Element Details for all elemnents:");

for (Element e : Element.values()) {

System.out.println("Symbol: " + e.name() + ", Name: " + e.label +

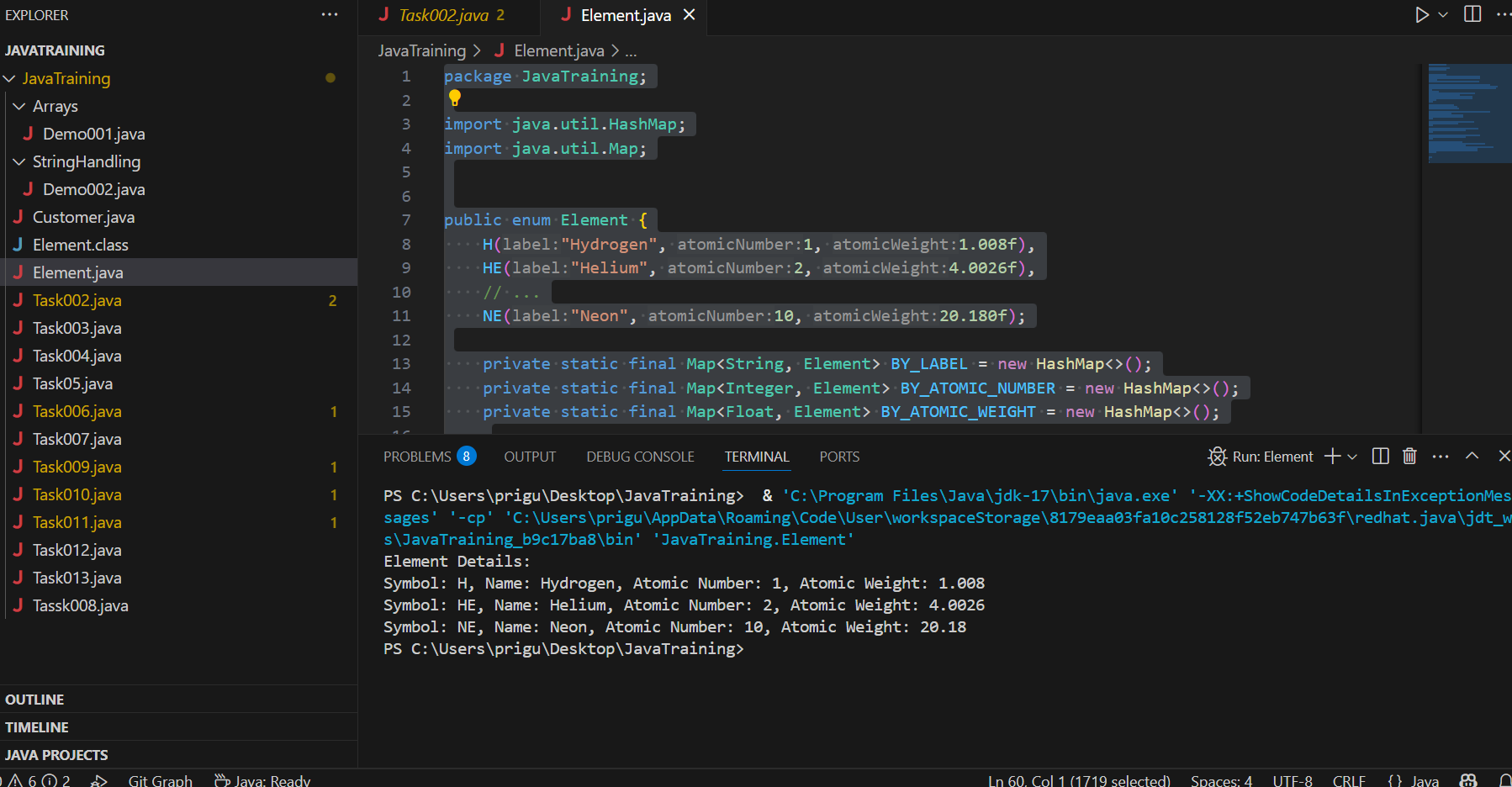
", Atomic Number: " + e.atomicNumber +

", Atomic Weight: " + e.atomicWeight);

}

}

}



Task 017:

Getter and setter

Ans: Unable to access as the variables are created Private

package JavaTraining;

public class Task017 {

public static void main(String[] args) {

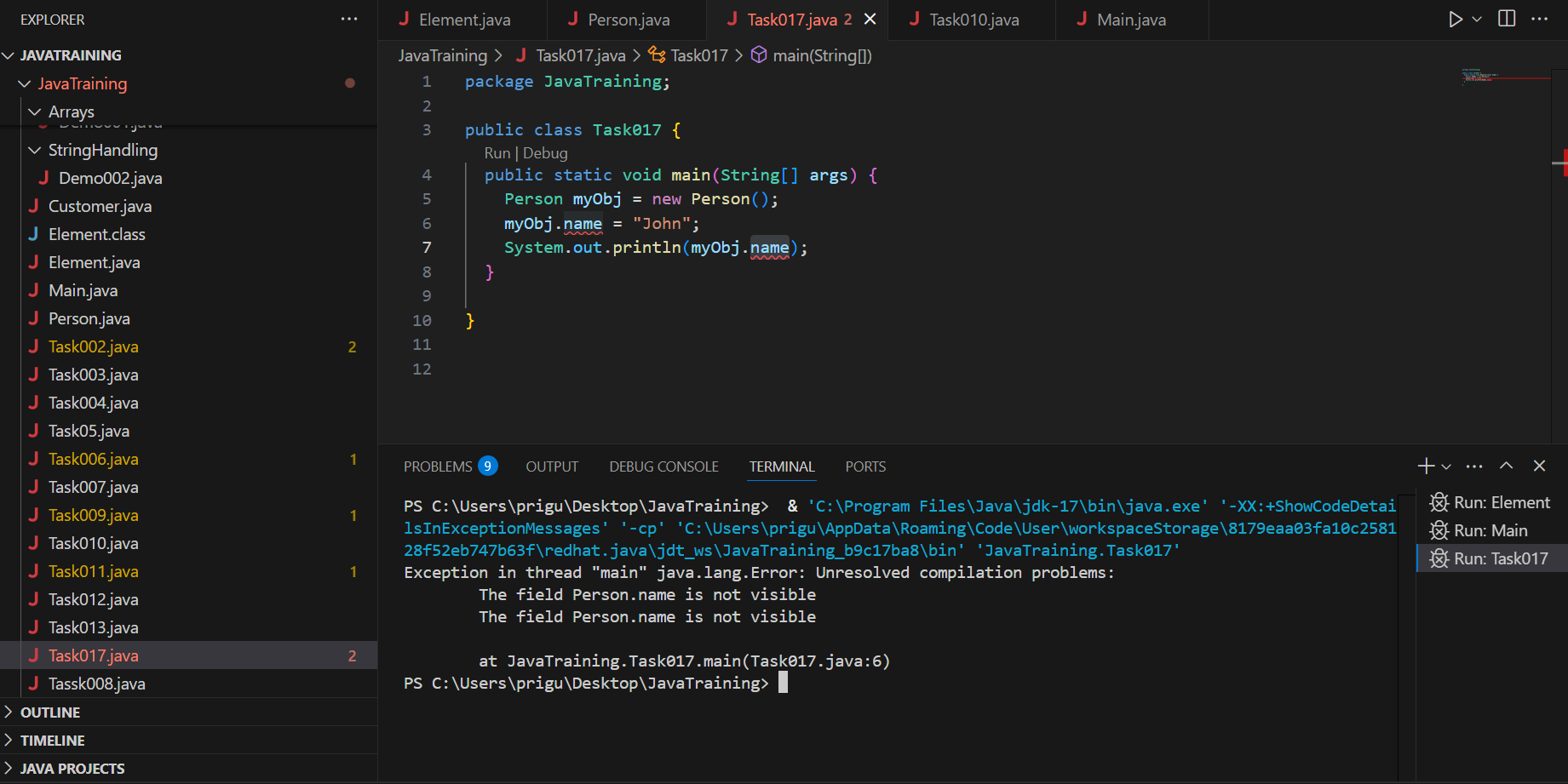
Person myObj = new Person();

myObj.name = "John";

System.out.println(myObj.name);

}

}



Task 018

Now create one more program named Task018.java

public class Main {

package JavaTraining;

public class Main {

public static void main(String[] args) {

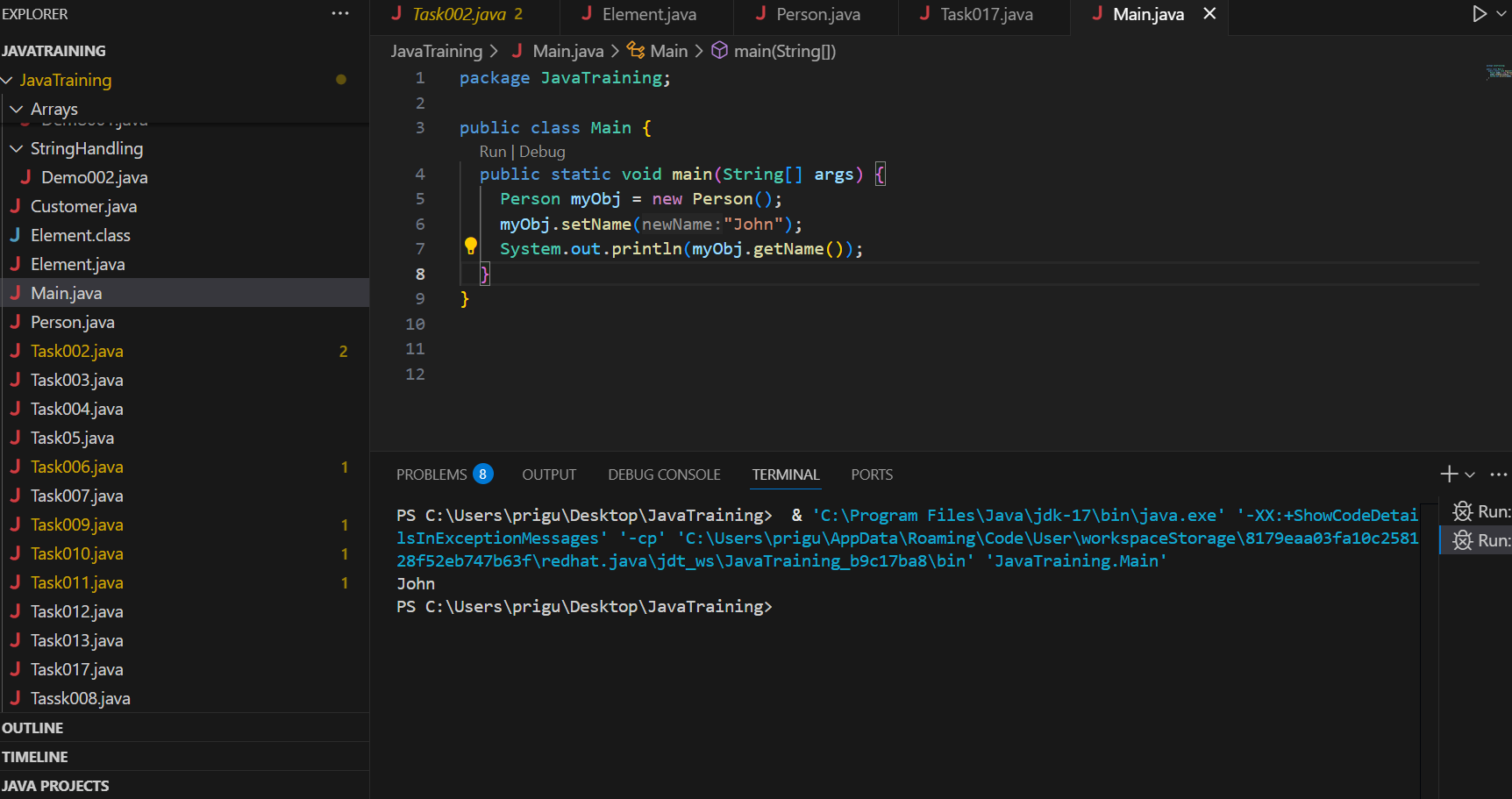
Person myObj = new Person();

myObj.setName("John");

System.out.println(myObj.getName());

}

}



Task 19 👍

// Source code is decompiled from a .class file using FernFlower decompiler.

package JavaTraining;

import java.io.PrintStream;

import java.util.HashMap;

import java.util.Map;

public enum Element {

H("Hydrogen", 1, 1.008F),

HE("Helium", 2, 4.0026F),

NE("Neon", 10, 20.18F);

private static final Map<String, Element> BY\_LABEL = new HashMap();

private static final Map<Integer, Element> BY\_ATOMIC\_NUMBER = new HashMap();

private static final Map<Float, Element> BY\_ATOMIC\_WEIGHT = new HashMap();

public final String label;

public final int atomicNumber;

public final float atomicWeight;

private Element(String var3, int var4, float var5) {

this.label = var3;

this.atomicNumber = var4;

this.atomicWeight = var5;

}

public static Element valueOfLabel(String var0) {

return (Element)BY\_LABEL.get(var0);

}

public static Element valueOfAtomicNumber(int var0) {

return (Element)BY\_ATOMIC\_NUMBER.get(var0);

}

public static Element valueOfAtomicWeight(float var0) {

return (Element)BY\_ATOMIC\_WEIGHT.get(var0);

}

public static void main(String[] var0) {

System.out.println("Element Details:");

Element[] var1 = values();

int var2 = var1.length;

for(int var3 = 0; var3 < var2; ++var3) {

Element var4 = var1[var3];

PrintStream var10000 = System.out;

String var10001 = var4.name();

var10000.println("Symbol: " + var10001 + ", Name: " + var4.label + ", Atomic Number: " + var4.atomicNumber + ", Atomic Weight: " + var4.atomicWeight);

}

}

static {

Element[] var0 = values();

int var1 = var0.length;

for(int var2 = 0; var2 < var1; ++var2) {

Element var3 = var0[var2];

BY\_LABEL.put(var3.label, var3);

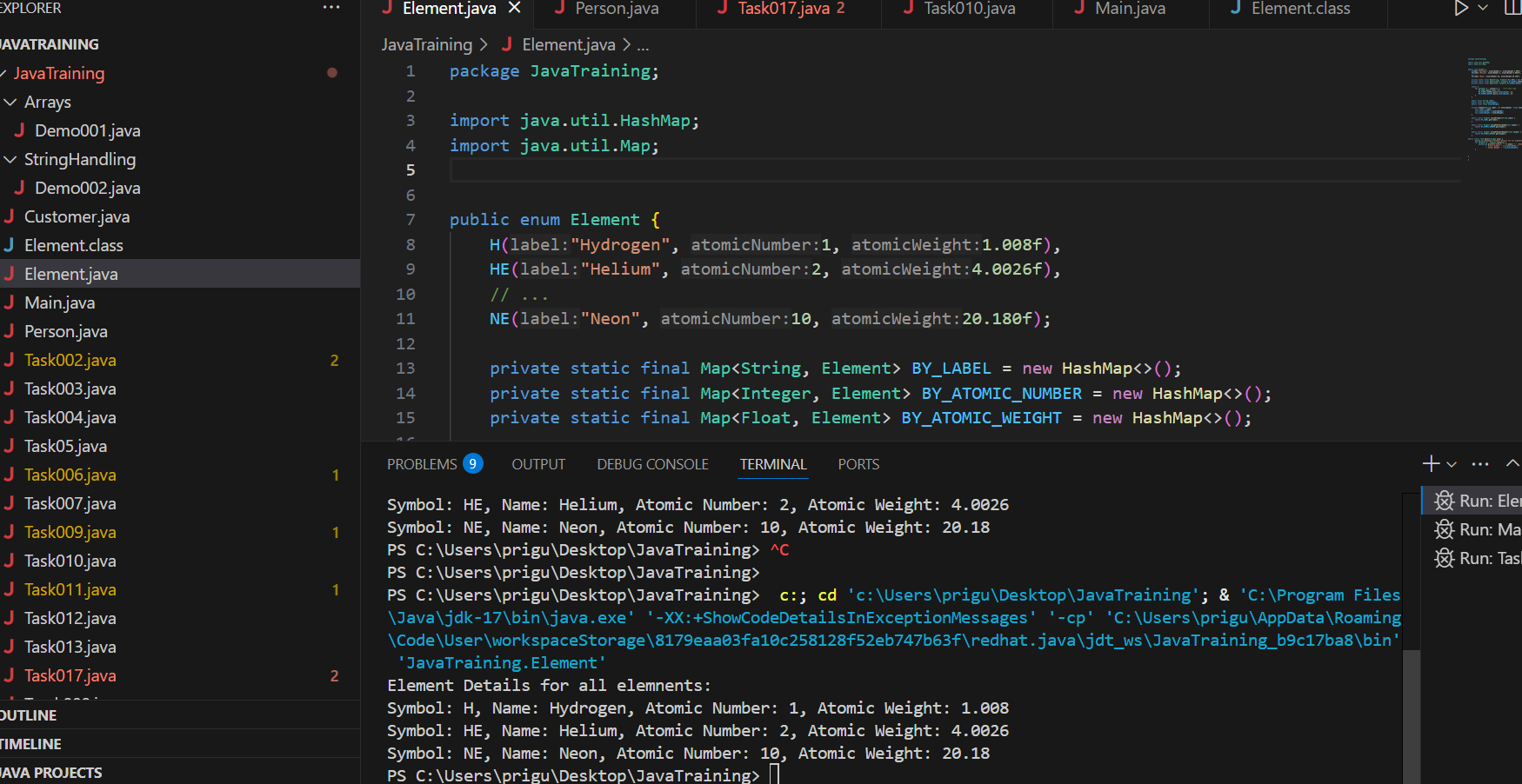
BY\_ATOMIC\_NUMBER.put(var3.atomicNumber, var3);

BY\_ATOMIC\_WEIGHT.put(var3.atomicWeight, var3);

}

}

}



Task 020:

Create an array of your name

Hint : use

package JavaTraining.Arrays;

public class Task020 {

public static void main(String[] args) {

System.out.println(" Lets print length of my name : Priya");

char [] Name = { 'P', 'R','I', 'Y','A'};

System.out.print("My name is: ");

for (int i = 0; i < Name.length; i++) {

System.out.print(Name[i]);

}

System.out.println();

int n = Name.length; // size of your name

System.out.println(" There are "+ n + " number of letters in my name");

}

}

Task 021

Task 21: Home Task

**Example:** This example demonstrates how to initialize an array and traverse it using a for loop to print each element.

public class Task021 {

public static void main(String[] args) {

// Initializing array

int[] arr = {1, 2, 3, 4, 5};

// Size of array

int n = arr.length;

// Traversing array

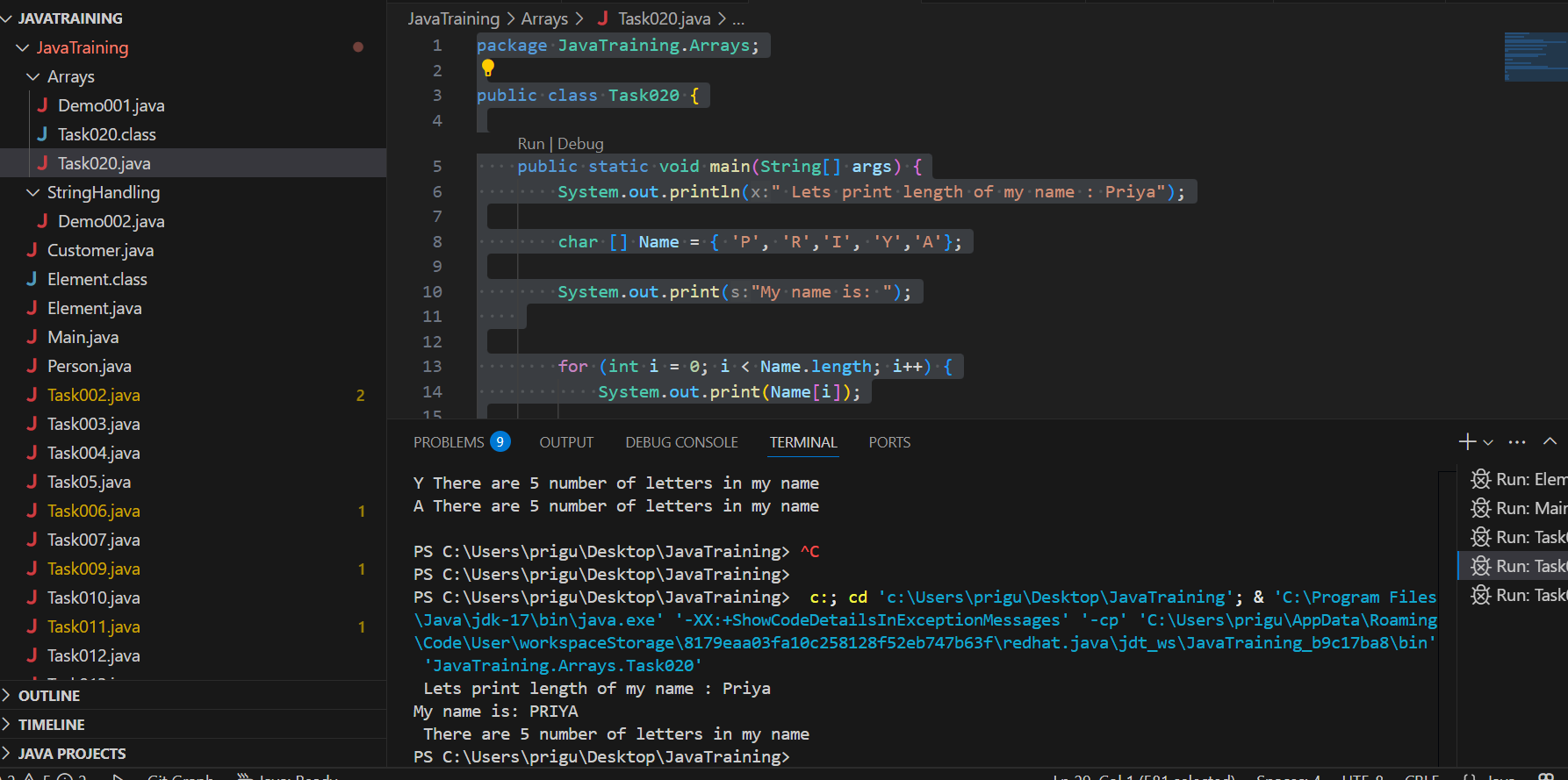
for (int i = 0; i < n; i++) {

System.out.print(arr[i] + " ");

}

}

}



Task 22

*Task 022 - home task*

#### **Implementation:**

// Java program to illustrate creating an array

// of integers, puts some values in the array,

// and prints each value to standard output.

public class Task022 {

public static void main(String[] args) {

// declares an Array of integers.

int[] arr;

// allocating memory for 5 integers.

arr = new int[5];

// initialize the elements of the array

// first to last (fifth) element

arr[0] = 10;

arr[1] = 20;

arr[2] = 30;

arr[3] = 40;

arr[4] = 50;

// accessing the elements of the specified array

for (int i = 0; i < arr.length; i++) {

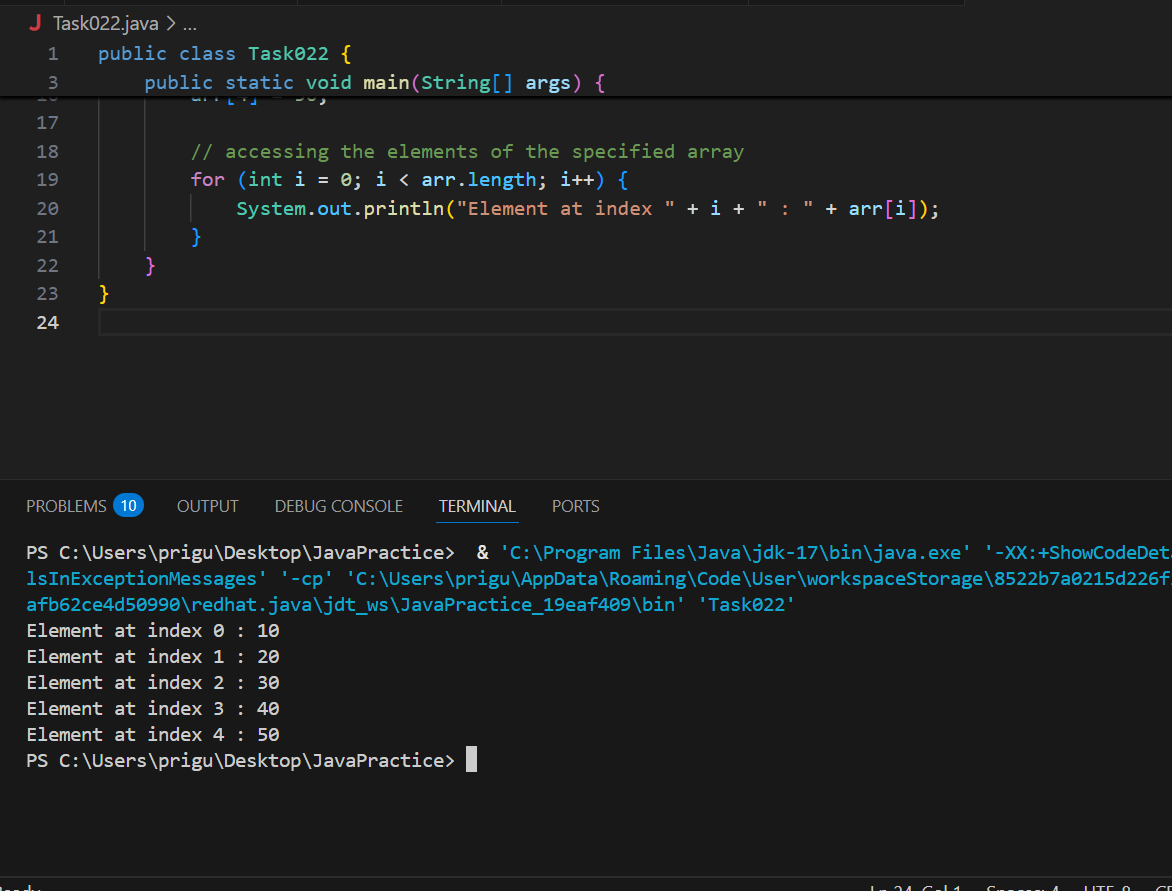
System.out.println("Element at index " + i + " : " + arr[i]);

}

}

}

​



Task 023 - home task

**Example:** Here we are taking a student class and creating an array of Student with five Student objects stored in the array. The Student objects have to be instantiated using the constructor of the Student class, and their references should be assigned to the array elements.

// Java program to illustrate creating

// an array of objects

class Student {

public int roll\_no;

public String name;

// Parameterized constructor

Student(int roll\_no, String name) {

this.roll\_no = roll\_no;

this.name = name;

}

}

public class Task023 {

public static void main(String[] args) {

// Declares and initializes an array of Student objects

Student[] arr = new Student[5];

// Initializing the elements of the array

arr[0] = new Student(1, "Aman");

arr[1] = new Student(2, "Vaibhav");

arr[2] = new Student(3, "Shikhar");

arr[3] = new Student(4, "Dharmesh");

arr[4] = new Student(5, "Mohit");

// Accessing the elements of the specified array

for (int i = 0; i < arr.length; i++) {

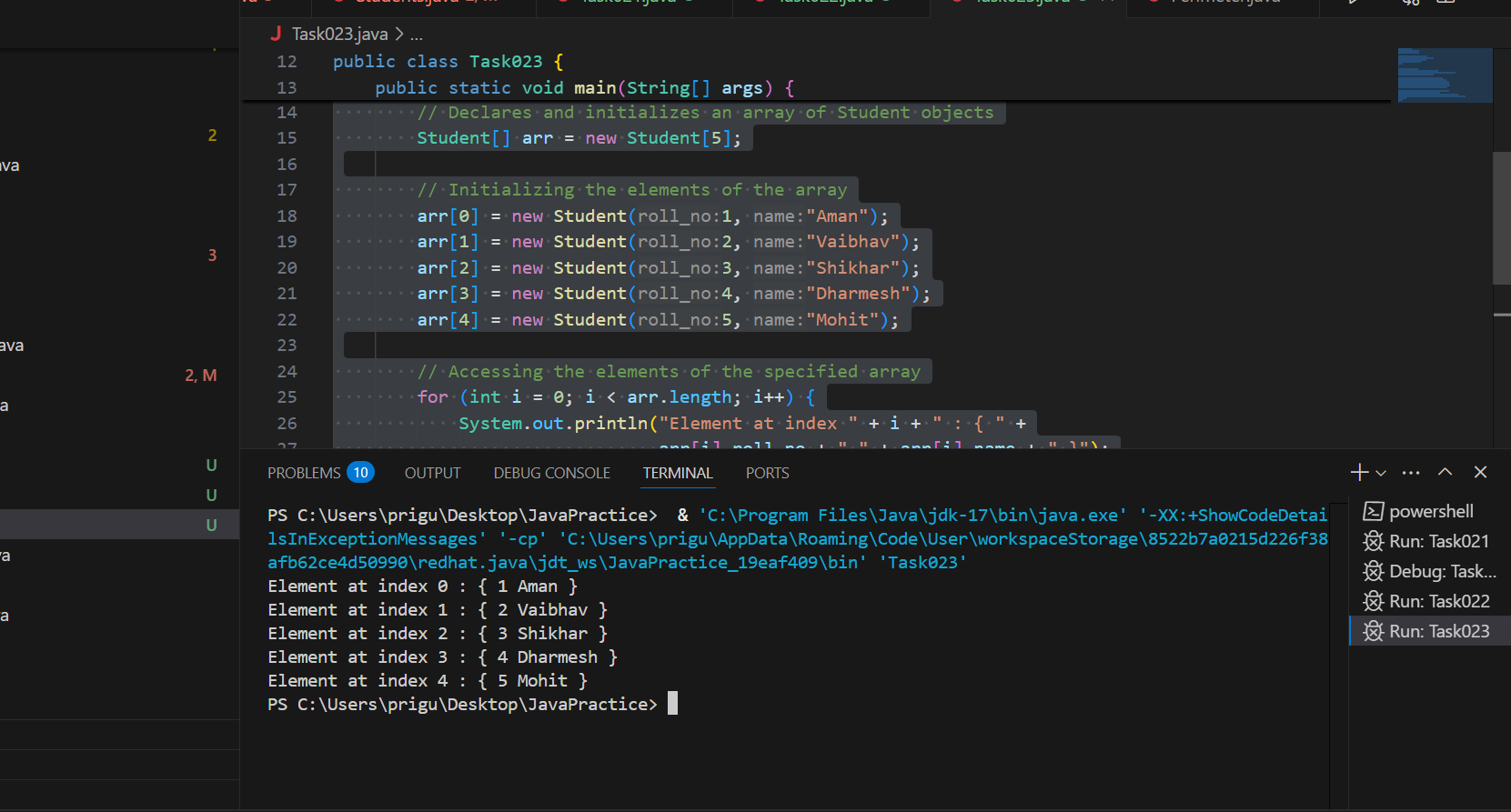
System.out.println("Element at index " + i + " : { " +

arr[i].roll\_no + " " + arr[i].name + " }");

}

}

}



Task 024 Home task

**Example:** An array of objects is also created like

// Java program to illustrate creating

// an array of objects

package Arrays;

class Student {

public String name;

Student(String name) {

this.name = name;

}

@Override

public String toString() {

return name;

}

}

public class Task024 {

public static void main(String[] args) {

Student[] myStudents = new Student[]{

new Student("Dharma"),

new Student("Sanvi"),

new Student("Rupa"),

new Student("Ajay")

};

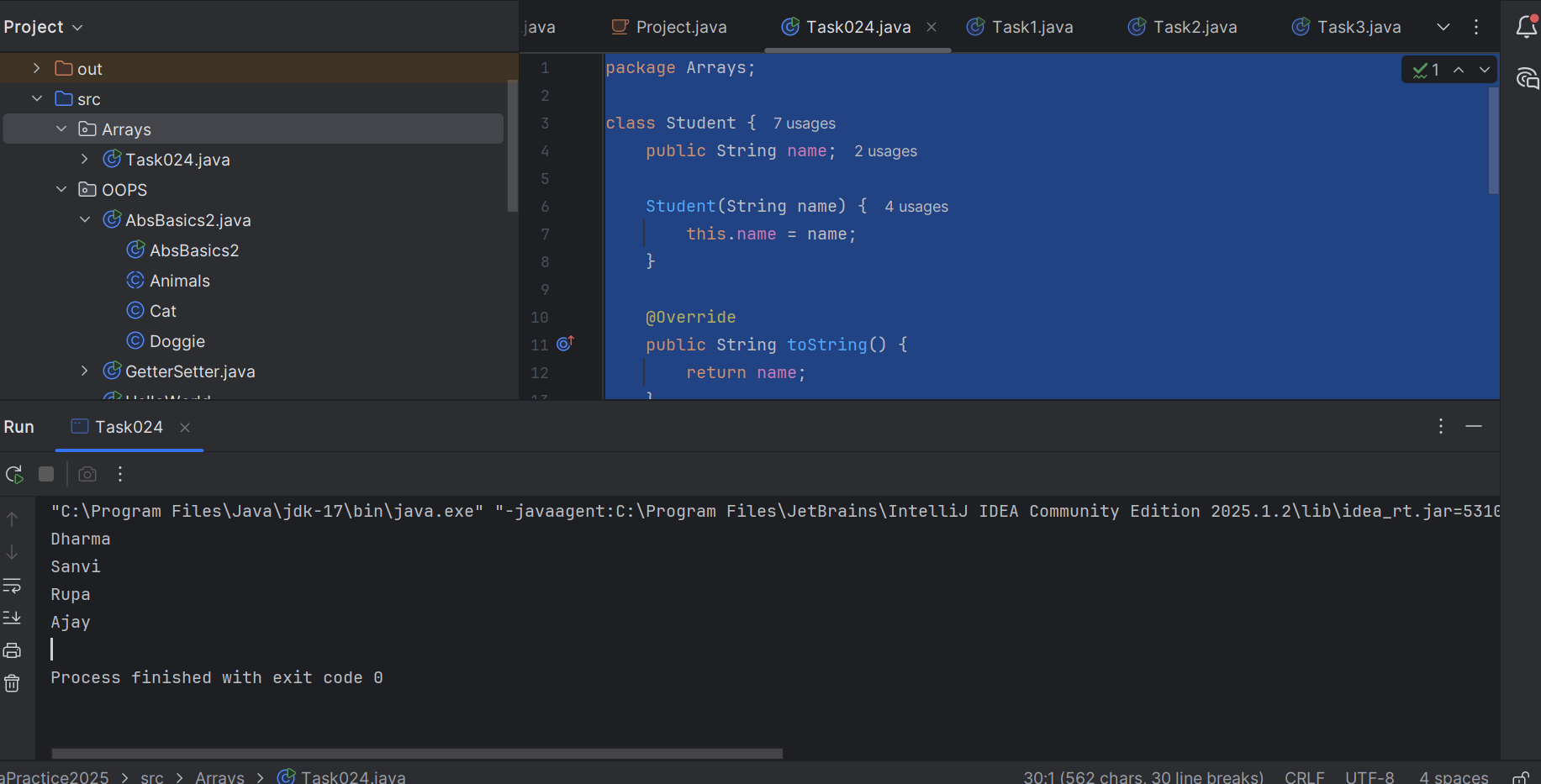
for (Student m : myStudents) {

System.*out*.println(m);

}

}

}



Task 025

// Code for showing error "ArrayIndexOutOfBoundsException"

public class Task025 {

public static void main(String[] args) {

int[] arr = new int[4];

arr[0] = 10;

arr[1] = 20;

arr[2] = 30;

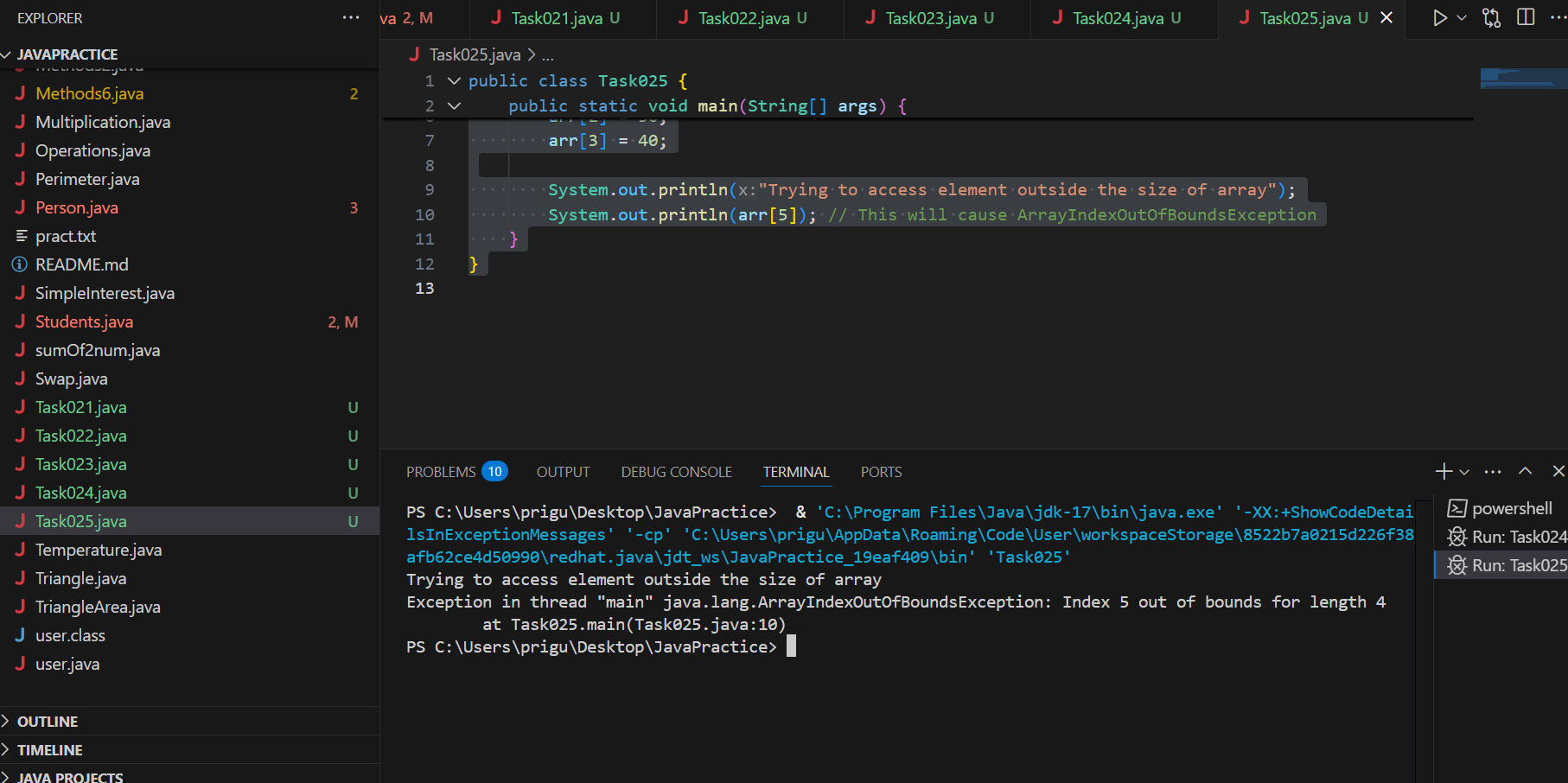
arr[3] = 40;

System.out.println("Trying to access element outside the size of array");

System.out.println(arr[5]); // This will cause ArrayIndexOutOfBoundsException

}

}



Task 025 - home task

**Example:** Let us start with basic two dimensional Array declared and initialized.

// Java Program to demonstrate

// Multidimensional Array

import java.io.\*;

public class Task025 {

public static void main(String[] args) {

// Two Dimensional Array

// Declared and Initialized

int[][] arr = new int[3][3];

// Number of Rows

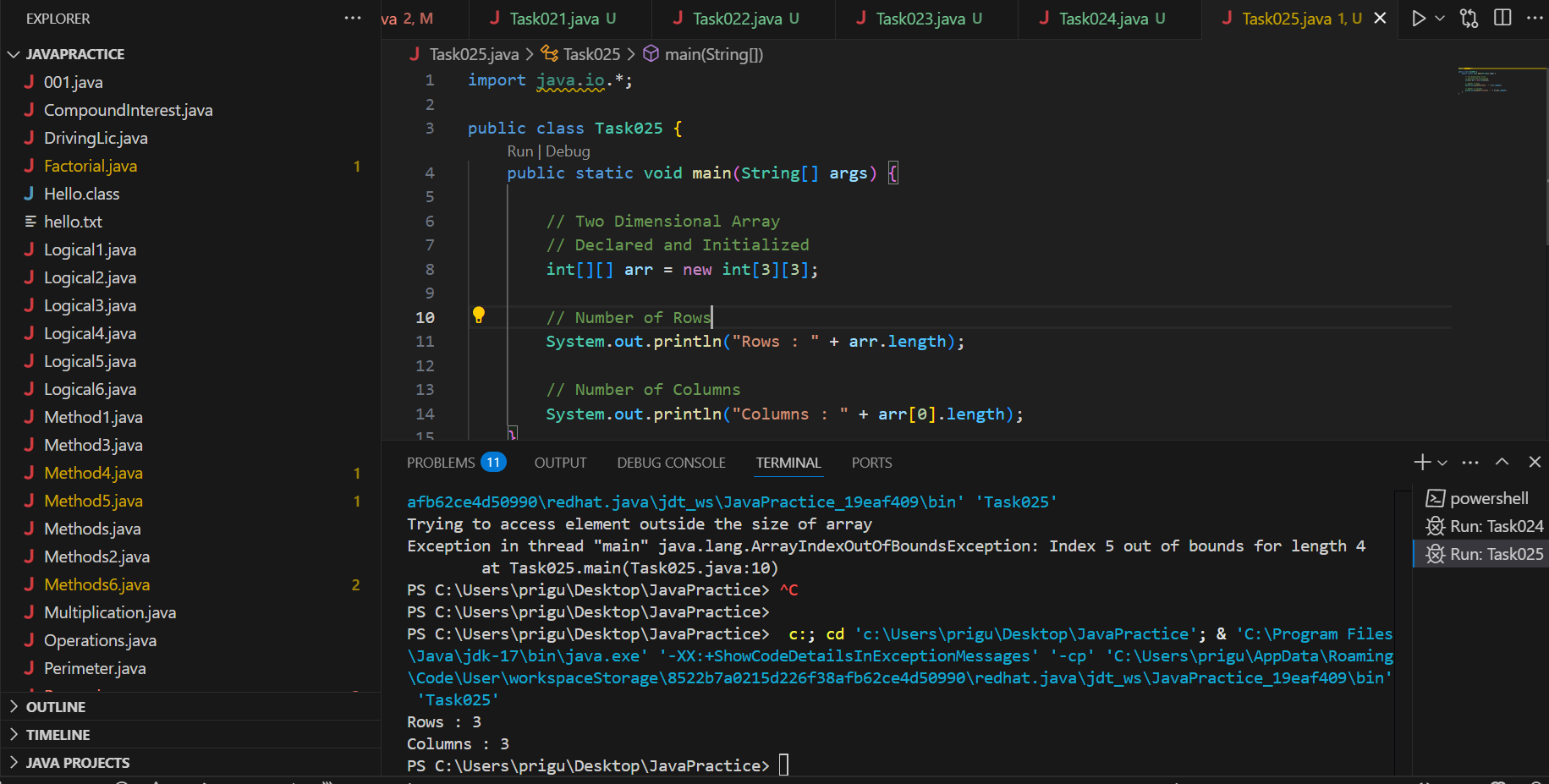
System.out.println("Rows : " + arr.length);

// Number of Columns

System.out.println("Columns : " + arr[0].length);

}

}



Task 027

public class Task026 {

// main function

public static void main(String args[]) {

// declaring and initializing 2D array

int arr[][] = { { 2, 7, 9 }, { 3, 6, 1 }, { 7, 4, 2 } };

// printing 2D array

for (int i = 0; i < 3; i++) { // rows

for (int j = 0; j < 3; j++) // columns

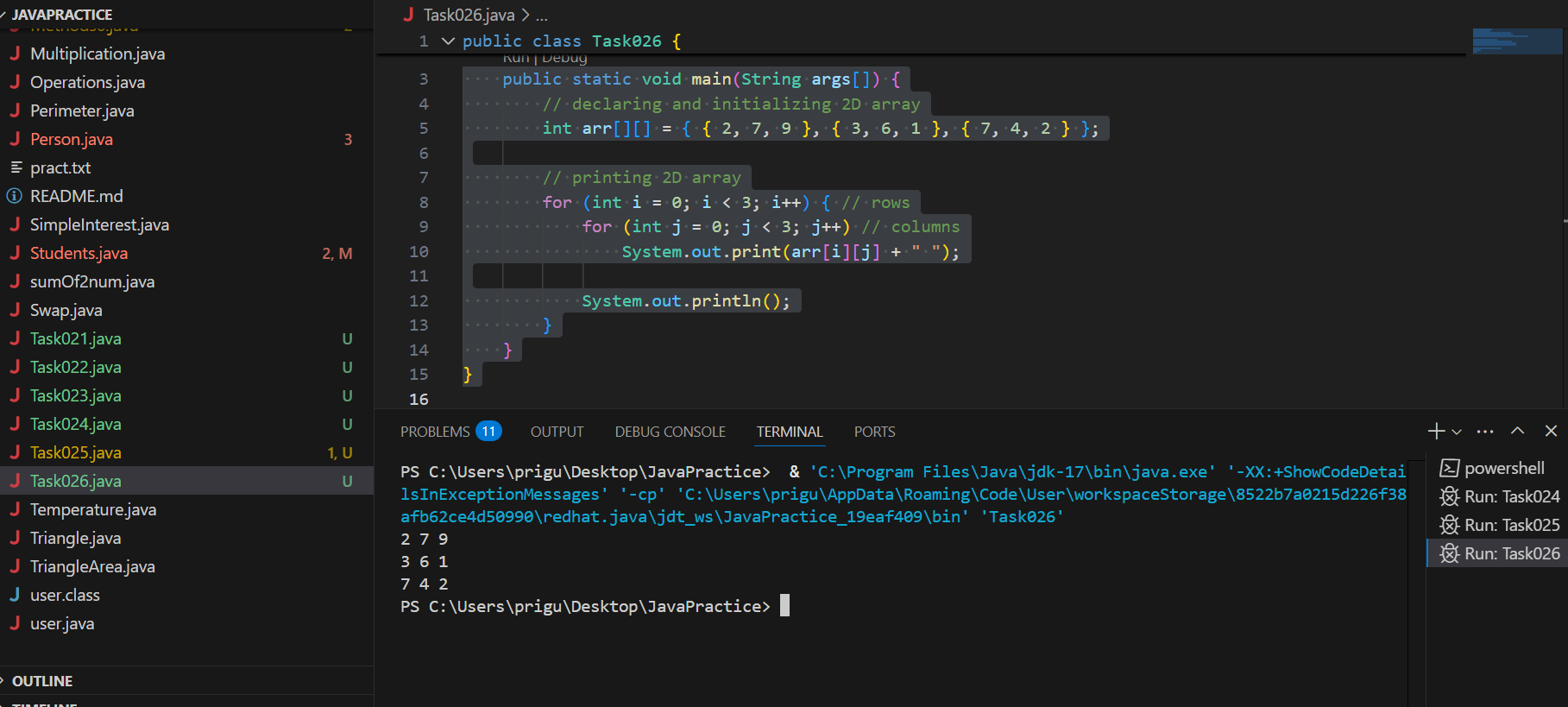
System.out.print(arr[i][j] + " ");

System.out.println();

}

}

}



Task 28

// Java program to demonstrate

// passing of array to method

public class Task027 {

// Driver method

public static void main(String args[]) {

int arr[] = { 3, 1, 2, 5, 4 };

// passing array to method sum

sum(arr);

}

public static void sum(int[] arr) {

// getting sum of array values

int sum = 0;

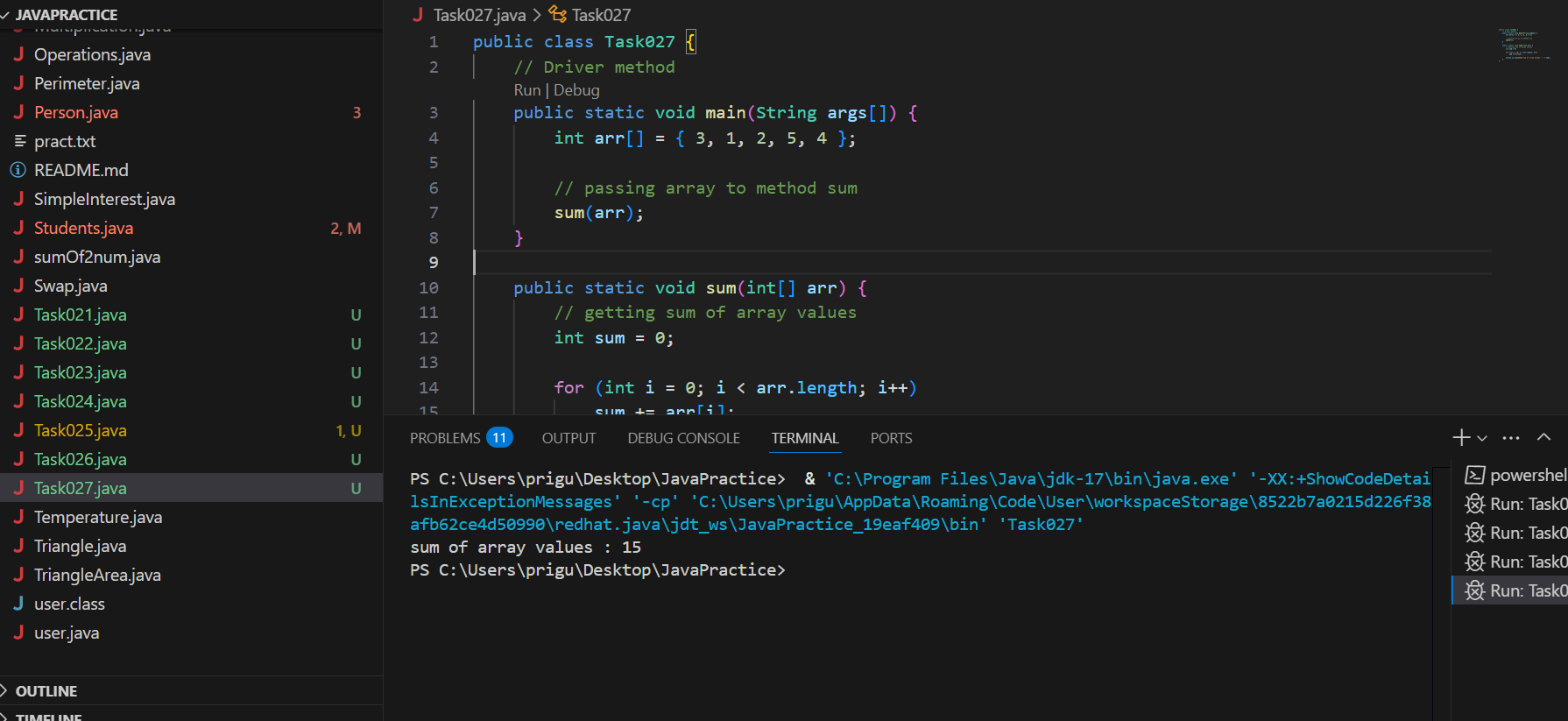
for (int i = 0; i < arr.length; i++)

sum += arr[i];

System.out.println("sum of array values : " + sum);

}

}



Task 028

Task 28 - Home Task

// Java program to demonstrate

// return of array from method

public class Task028 {

// Driver method

public static void main(String args[]) {

int arr[] = m1();

for (int i = 0; i < arr.length; i++)

System.out.print(arr[i] + " ");

}

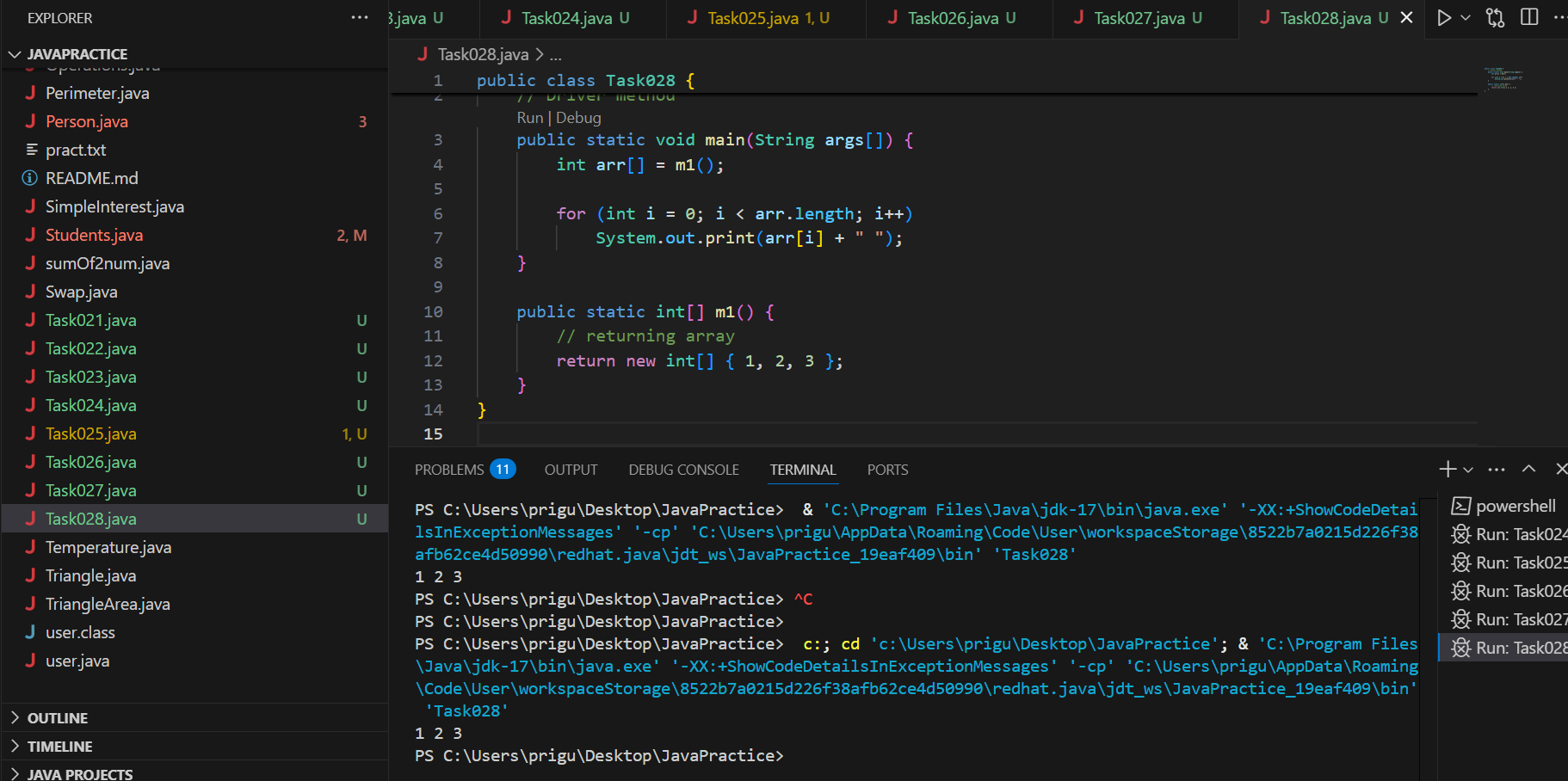
public static int[] m1() {

// returning array

return new int[] { 1, 2, 3 };

}

}



Task 29 👍**Java Array Members**

Now, as you know that arrays are objects of a class, and a direct superclass of arrays is a class Object.

package JavaTraining.Arrays;

public class Task029 {

public static void main(String args[]) {

int intArray[] = { 1, 2, 3 }; // Original array

int cloneArray[] = intArray.clone(); // Create a copy

intArray[1] = 4; // Change 2nd element in the original array

// Check if both arrays are the same (point to same memory)

System.out.println(intArray == cloneArray); // false

// Print the cloned array

for (int i = 0; i < cloneArray.length; i++) {

System.out.print(cloneArray[i] + " ");

}

System.out.println("\nOriginal Array");

// Print the original array

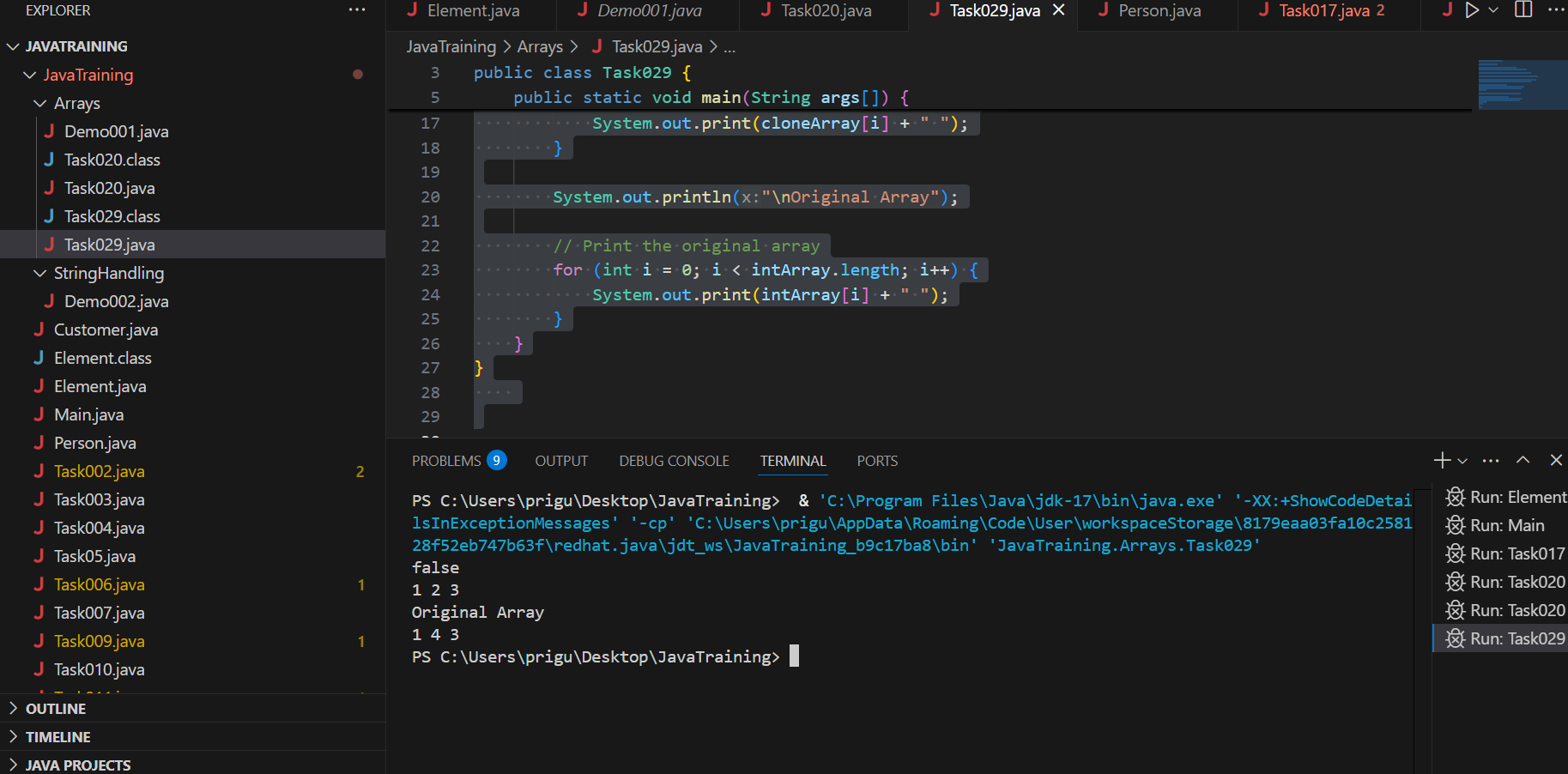
for (int i = 0; i < intArray.length; i++) {

System.out.print(intArray[i] + " ");

}

}

}



## **What’s Happening in Easy Words:**

1. You create an array:  
    👉 intArray = {1, 2, 3}
2. Then you **clone** it:  
    👉 cloneArray = intArray.clone()  
    → It makes a **separate copy** of the array (new memory).
3. Then you change:  
    👉 intArray[1] = 4  
    → You updated the original array's **second number** from 2 → 4
4. System.out.println(intArray == cloneArray);  
    👉 This checks **"Are both arrays the same object?"** ❌ Answer: **No**, because clone() makes a new array.
5. You print both arrays:  
   * cloneArray → still shows: 1 2 3
   * intArray → changed: 1 4 3

Task 031

class Calculation {

int z;

public void addition(int x, int y) {

package JavaTraining;

public class 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 My\_Calculation extends Calculation {

public void multiplication(int x, int y) {

z = x \* y;

System.out.println("The product of the given numbers:"+z);

}

public static void main(String args[]) {

int a = 20, b = 10;

My\_Calculation demo = new My\_Calculation();

demo.addition(a, b);

demo.Subtraction(a, b);

demo.multiplication(a, b);

}

}

}

Task 31

package JavaTraining;

// Parent class

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);

}

}

// Child class

public class Task031 extends Calculation {

public void multiplication(int x, int y) {

z = x \* y;

System.out.println("The product of the given numbers: " + z);

}

public static void main(String args[]) {

int a = 20, b = 10;

Task031 demo = new Task031();

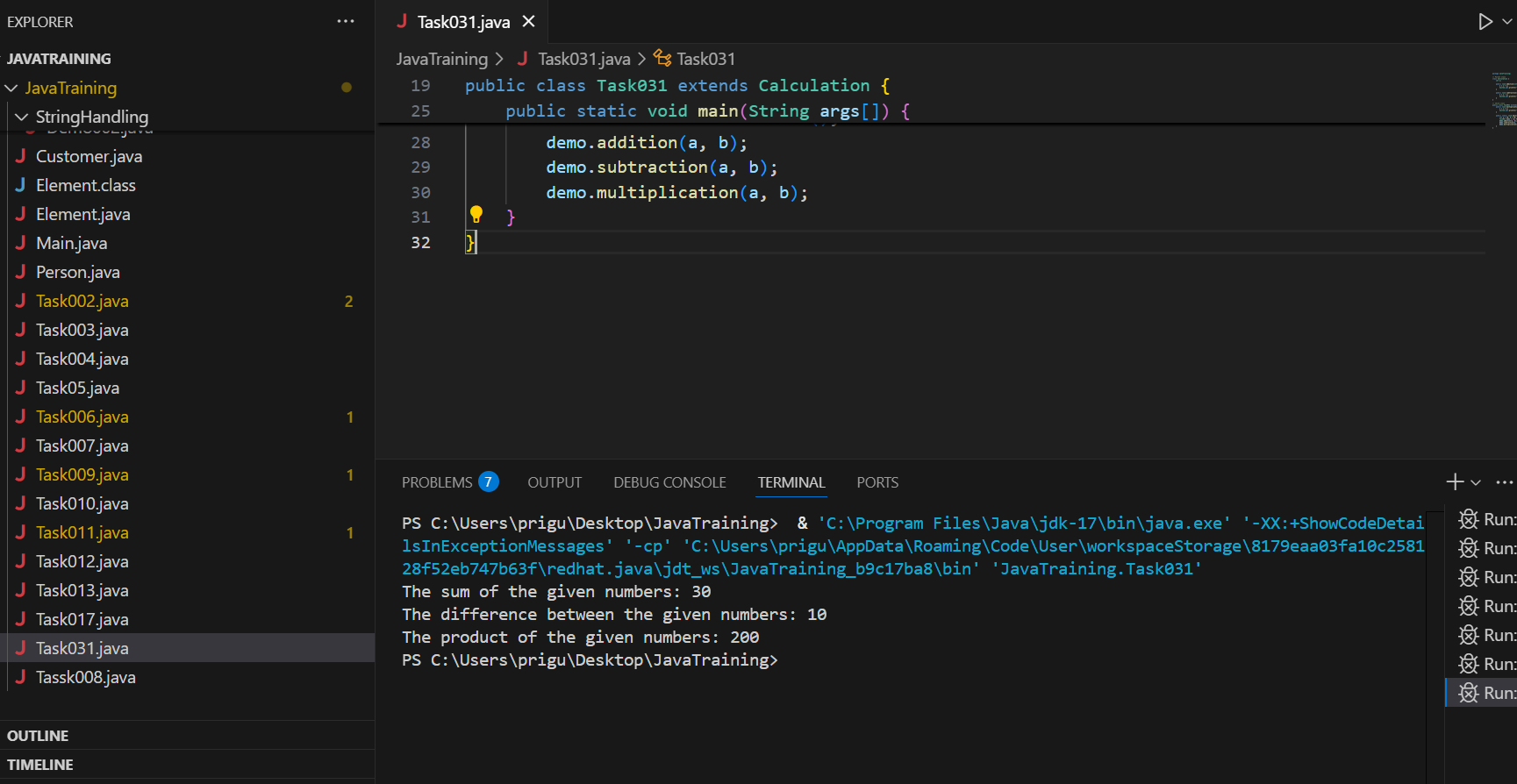
demo.addition(a, b);

demo.subtraction(a, b);

demo.multiplication(a, b);

}

}



Task 35 👍

package JavaTraining;

public class Task35 {

void add(char x, char y) {

System.out.println("char: " + x + ", " + y);

}

void add(int x, int y) {

System.out.println("int: " + x + ", " + y);

}

public static void main(String[] args) {

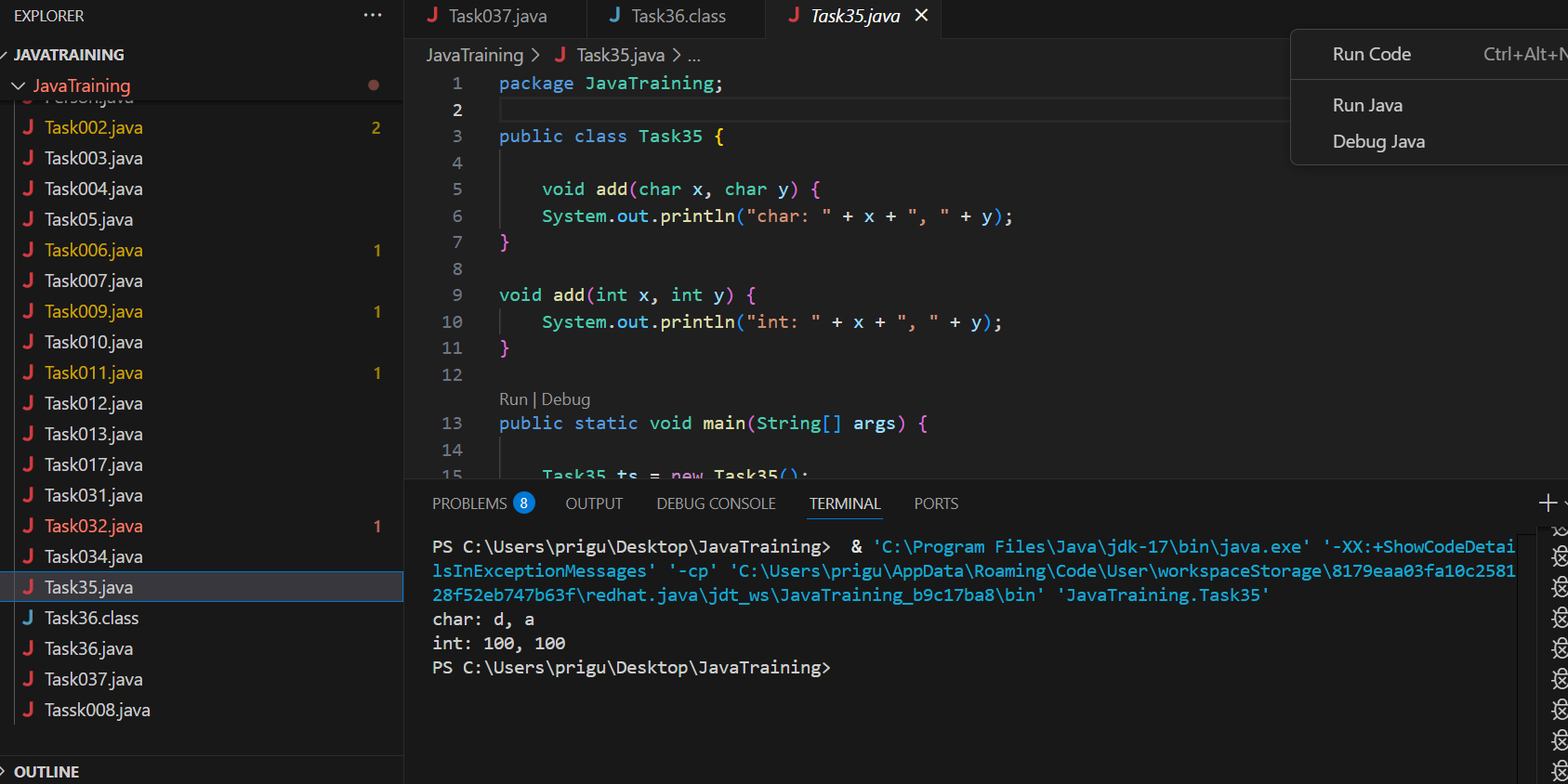
Task35 ts = new Task35();

ts.add('d', 'a'); // Calls add(char, char)

ts.add(100, 100); // Calls add(int, int)

}

}



Task 36

package JavaTraining;

public class Task36 {

void add (int x , float yf)

{

System.out.println("int x : " +x + " float y : "+ yf);

}

void add (float xf , int y)

{

System.out.println("int y : " +y + " float xf : "+ xf);

}

public static void main(String[] args) {

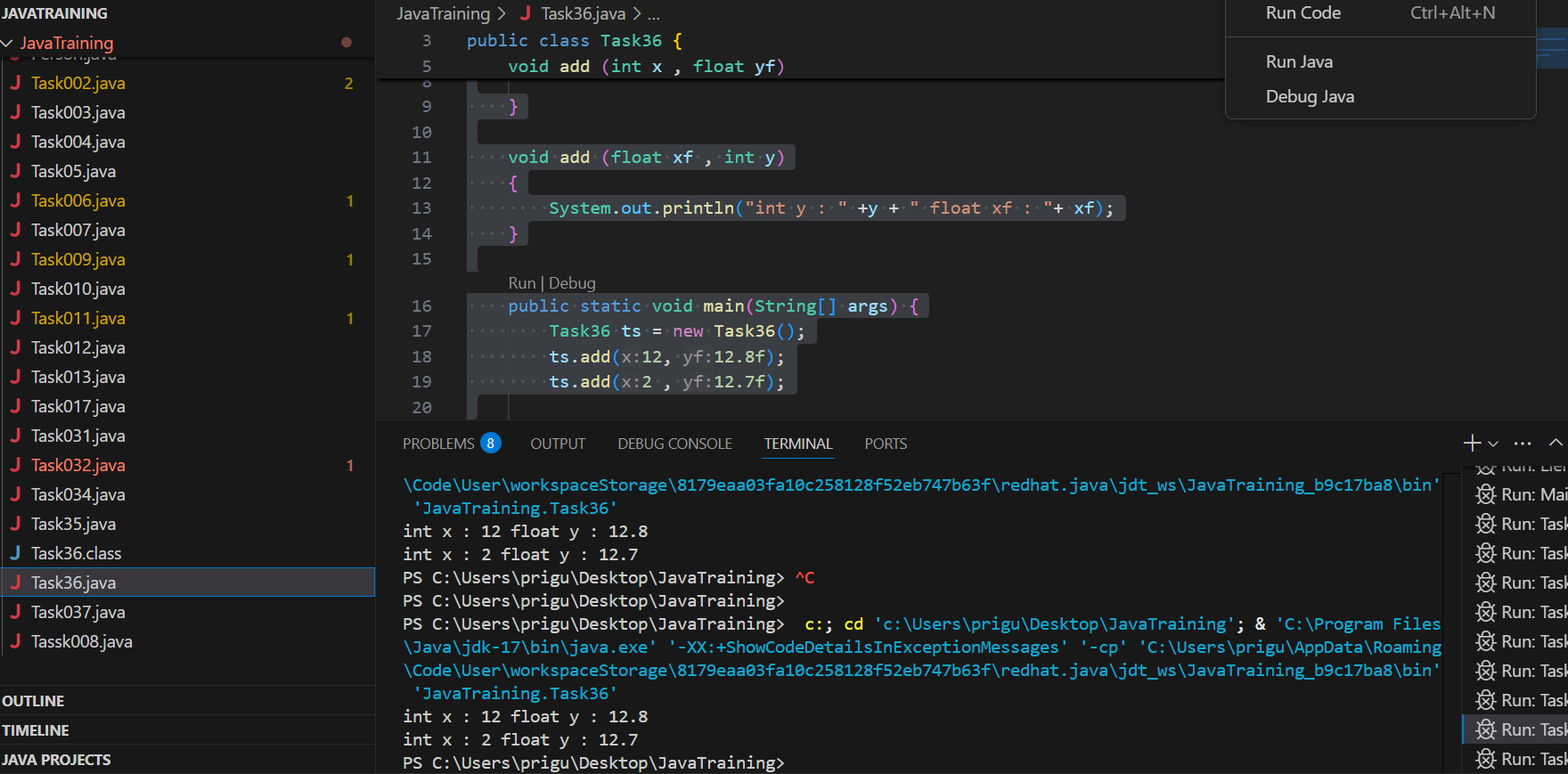
Task36 ts = new Task36();

ts.add(12, 12.8f);

ts.add(2 , 12.7f);

}

}



Task 37

package JavaTraining;

public class Task037 {

private int pwd;

protected int salary;

public int empid;

public Task037() {

pwd = 1234;

salary = 30000;

empid = 101;

}

public int getPwd() {

return pwd;

}

public void setPwd(int pwd) {

this.pwd = pwd;

}

public static class Hr extends Task037 {

public Hr() {

setPwd(1254); //for protected

salary = 50000; // accessible

empid = 10001; // accessible

}

public static void main(String[] args) {

Hr hr = new Hr();

System.out.println("EmpID: " + hr.empid);

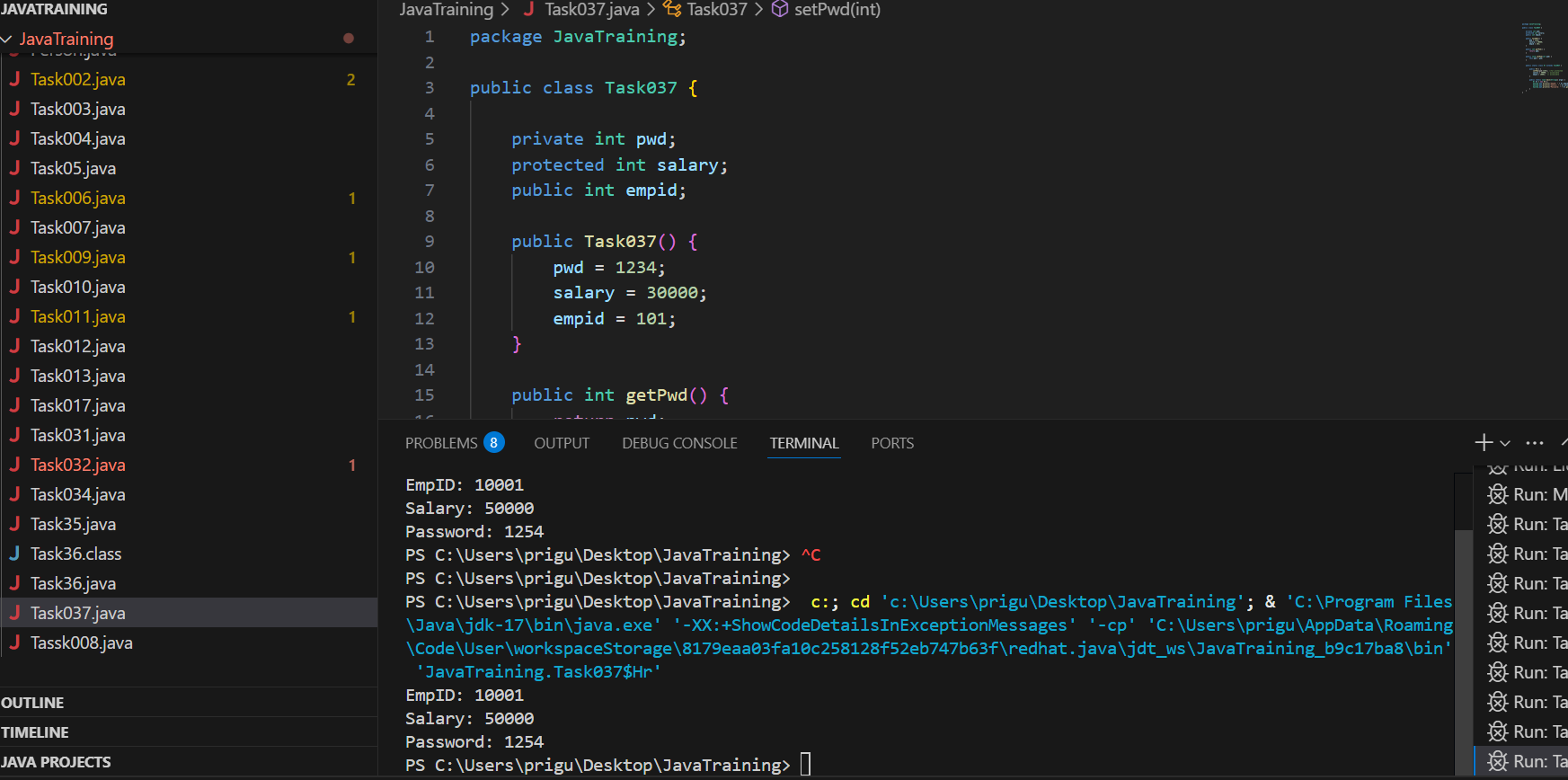
System.out.println("Salary: " + hr.salary);

System.out.println("Password: " + hr.getPwd());

}

}

}



Task 38 👍

Error :

package JavaTraining;

public class Abstraction {

public static void main(String [] args) {

/\* Following is not allowed and would raise error \*/

Employee e = new Employee("George W.", "Houston, TX", 43);

System.out.println("\n Call mailCheck using Employee reference--");

e.mailCheck();

}

}

abstract class Employee {

private String name;

private String address;

private int number;

public Employee(String name, String address, int number) {

System.out.println("Constructing an Employee");

this.name = name;

this.address = address;

this.number = number;

}

public double computePay() {

System.out.println("Inside Employee computePay");

return 0.0;

}

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;

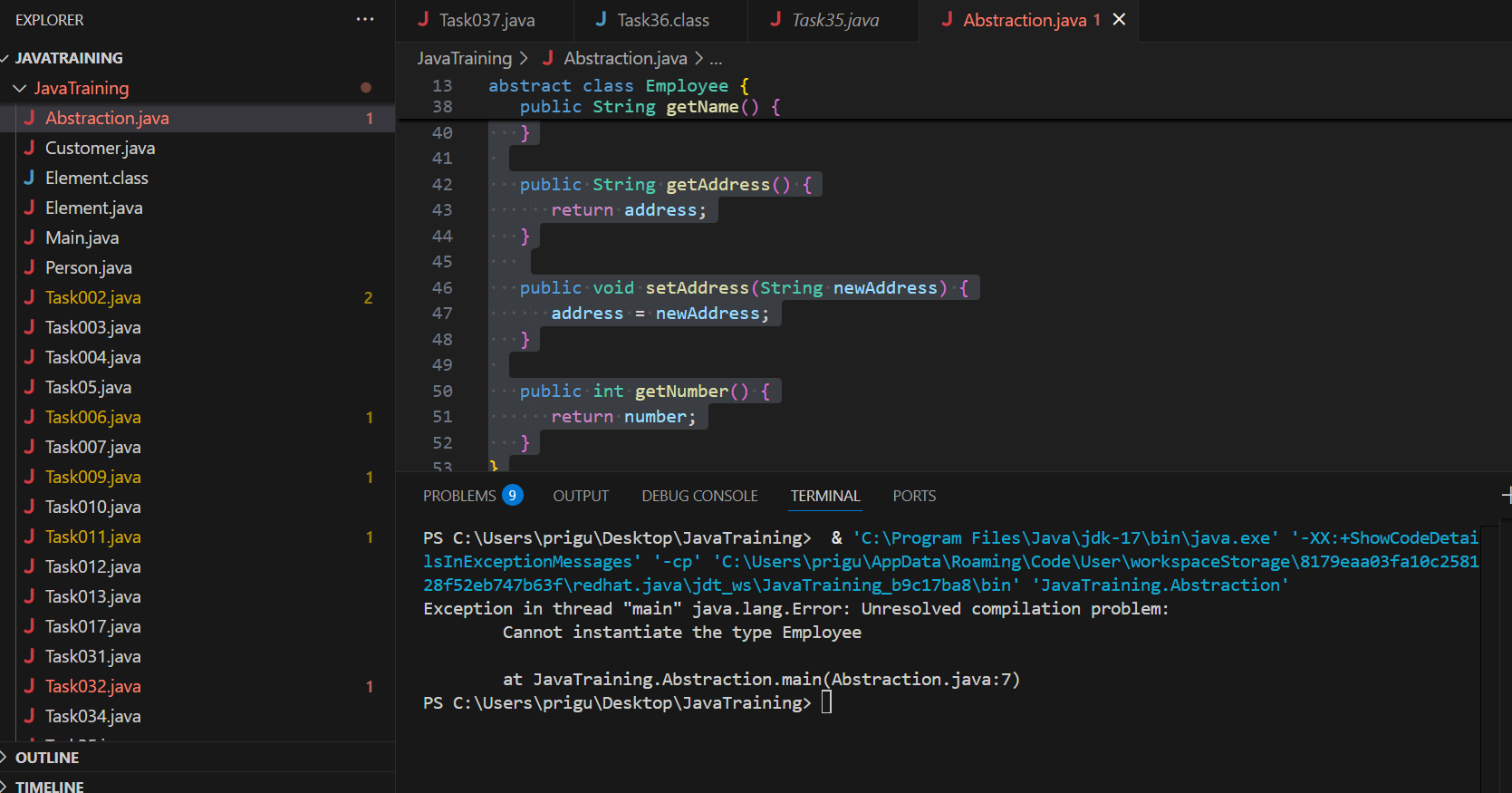
}

public int getNumber() {

return number;

}

}



After resolving error :

Task 039

Rewrite the above code to give the output without errors..

package JavaTraining;

public class Task039 {

public static void main(String[] args) {

// Create a Salary object

Employee e = new Salary("George W.", "Houston, TX", 43, 120000.0);

System.out.println("\nCall mailCheck using Employee reference--");

e.mailCheck();

System.out.println("Monthly Pay: Rs" + e.computePay());

}

}

// Abstract class

abstract class Employee {

private String name;

private String address;

private int number;

public Employee(String name, String address, int number) {

System.out.println("Creating an Employee document");

this.name = name;

this.address = address;

this.number = number;

}

public double computePay() {

System.out.println("Now we are inside Employee computePay");

return 0.0;

}

public void mailCheck() {

System.out.println("Mailing a check to " + name + " at " + 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;

}

}

// Concrete subclass

class Salary extends Employee {

private double annualSalary;

public Salary(String name, String address, int number, double salary) {

super(name, address, number); // Call constructor of Employee

this.annualSalary = salary;

}

// Override computePay method

@Override

public double computePay() {

System.out.println("Computing salary for " + getName());

return annualSalary / 12; // Monthly salary

}

// Override mailCheck method

@Override

public void mailCheck() {

System.out.println("Mailing paycheck to " + getName() + " with salary $" + annualSalary);

}

}

