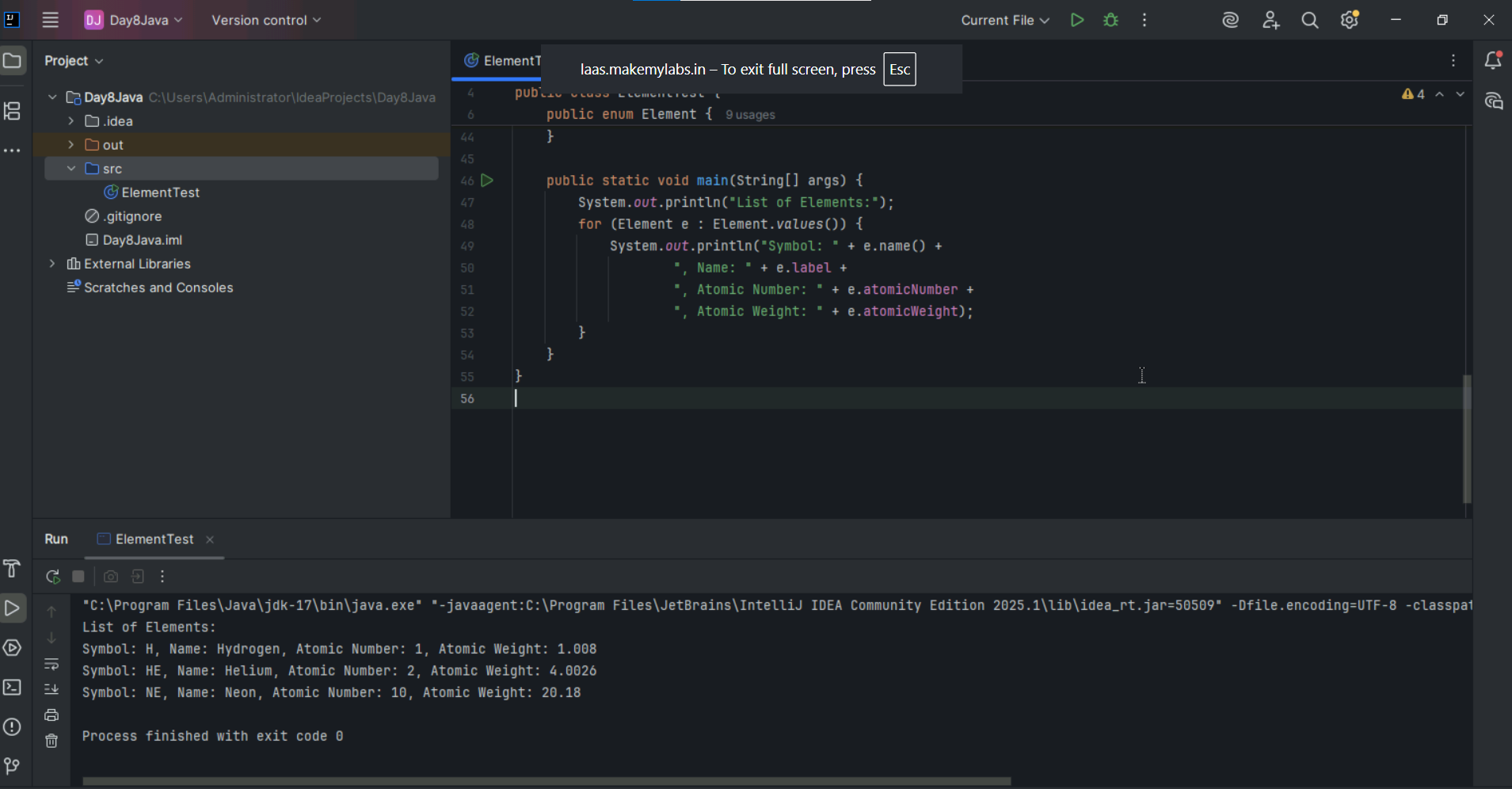
Task 19:

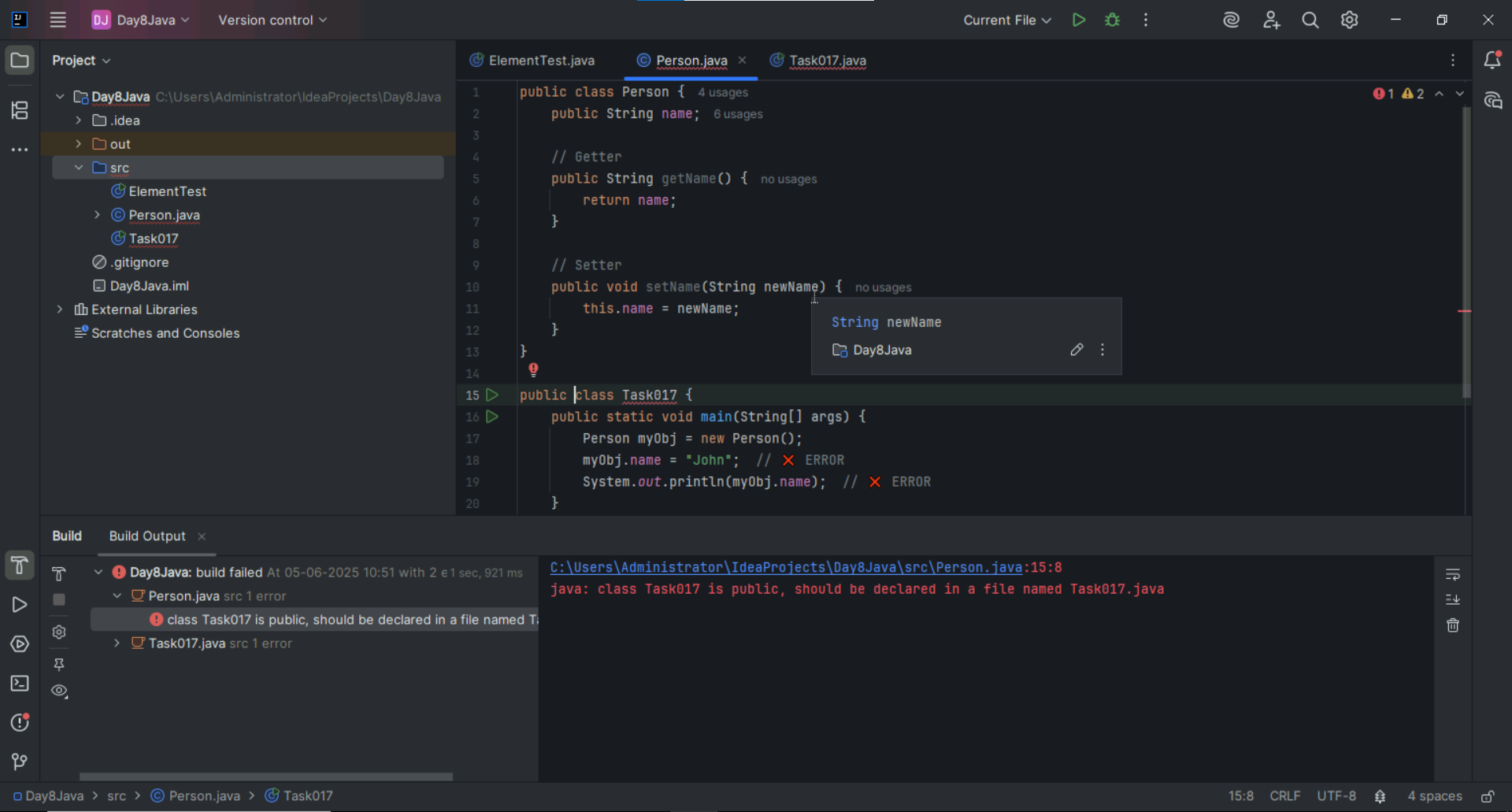
Wap to display the content of the above enum.. (main needs to be added)



Task 017:

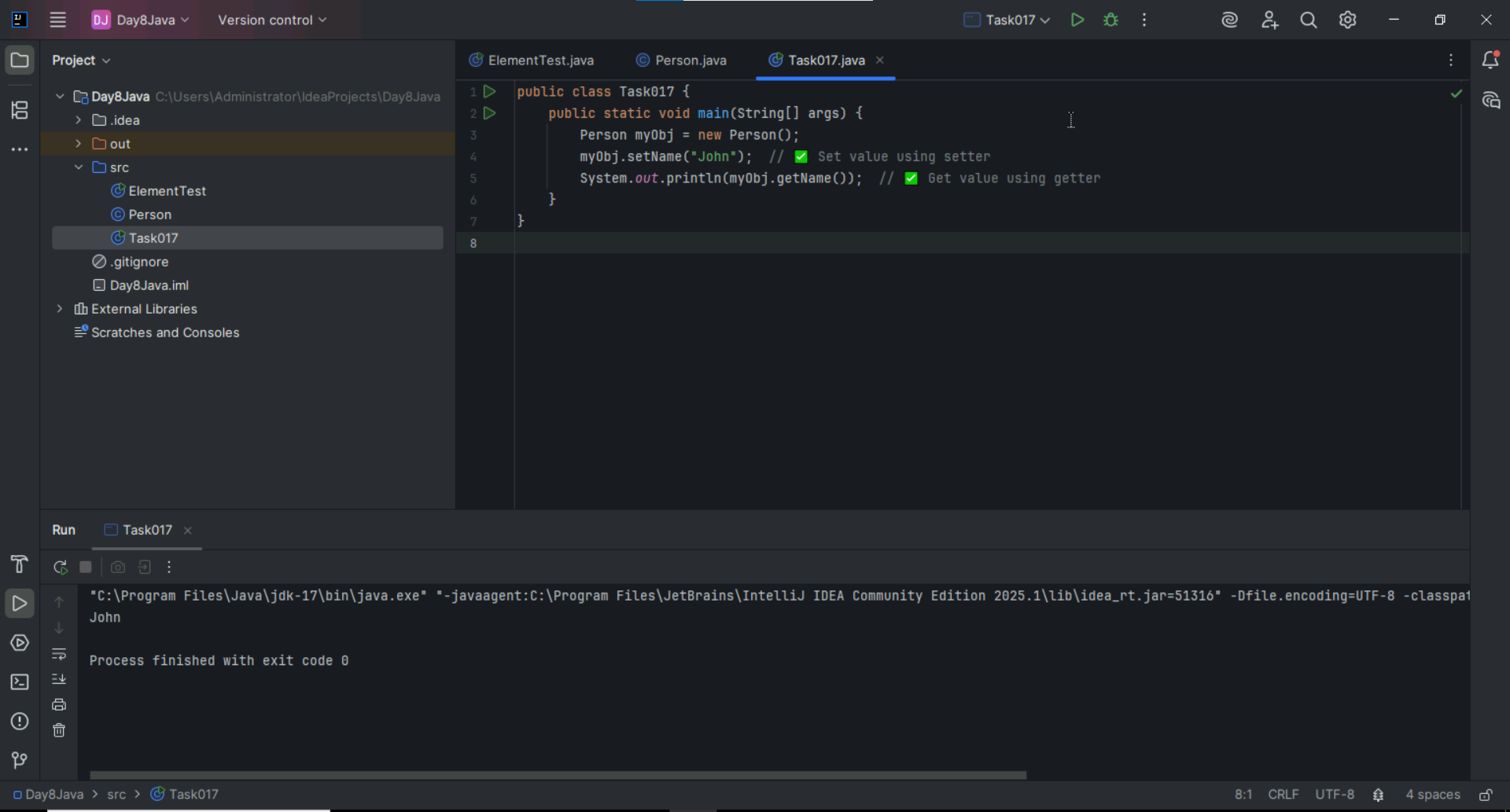
Getter and setter

Name is declared as private class, which and we are accessing it directly in task 17 which is public which is not allowed due to encapsulation.



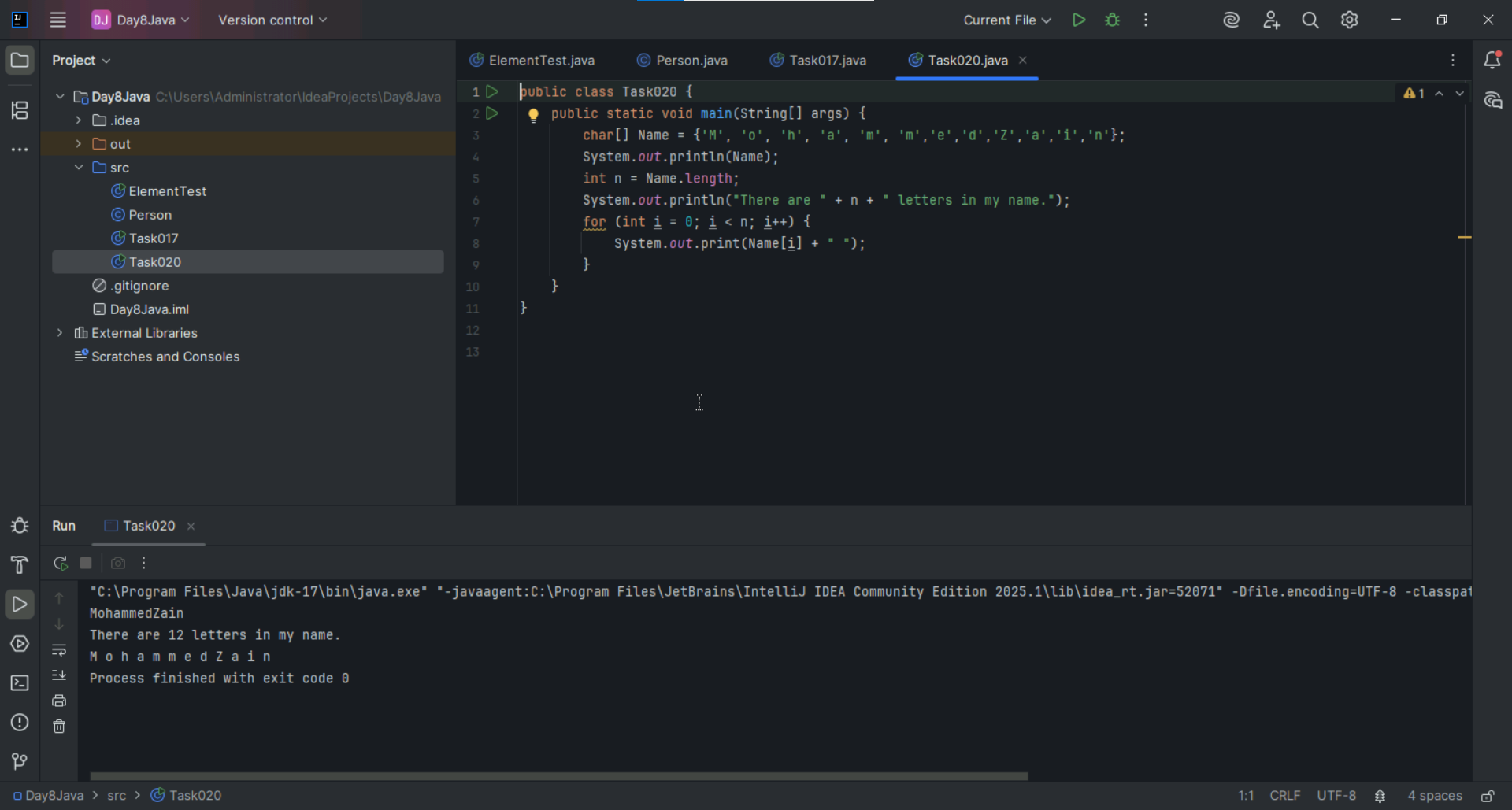
Task 18 :

Now create one more program named Task018.java



Task 020:

Create an array of your name



Task 031

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();

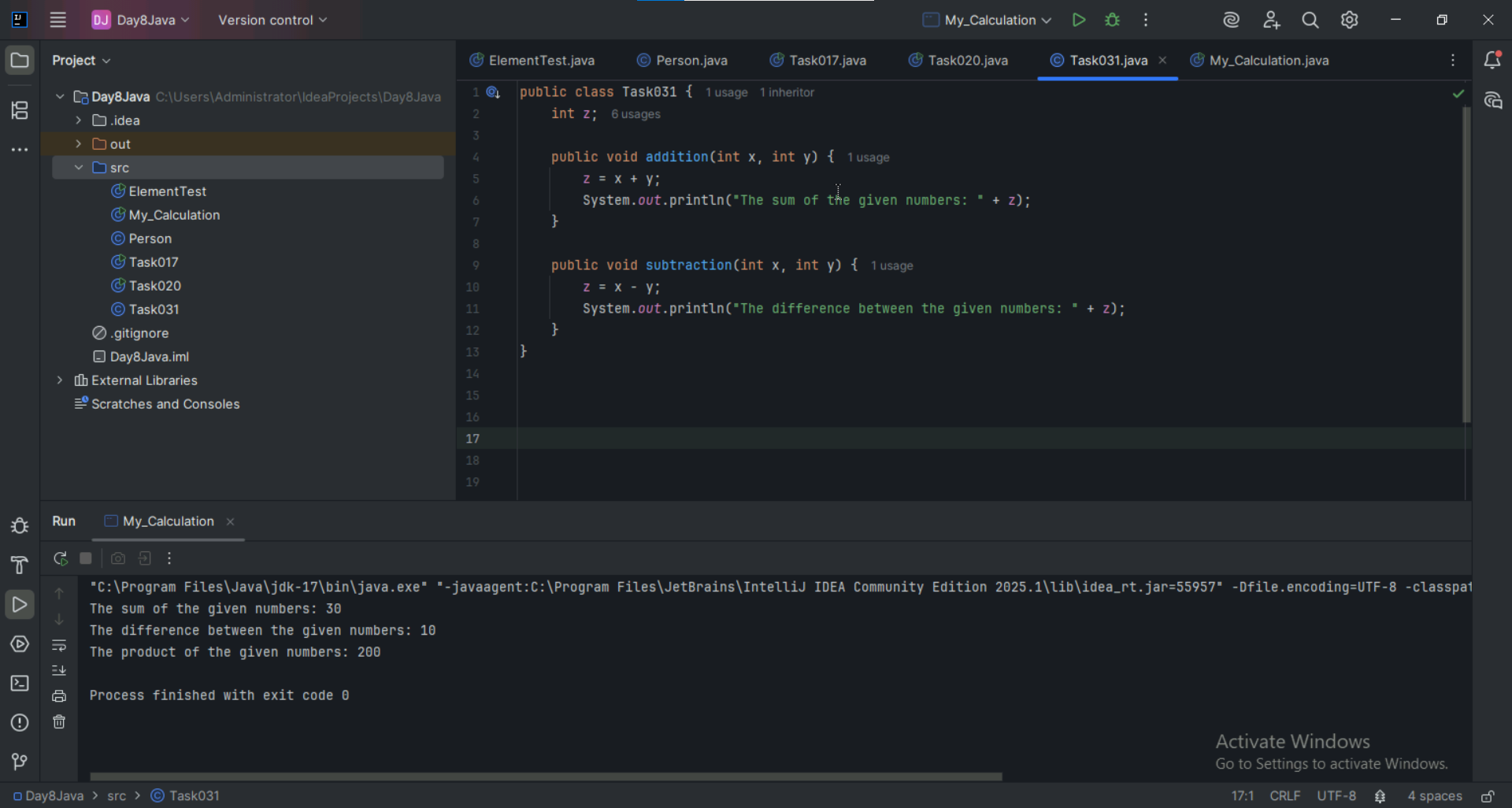
demo.addition(a, b);

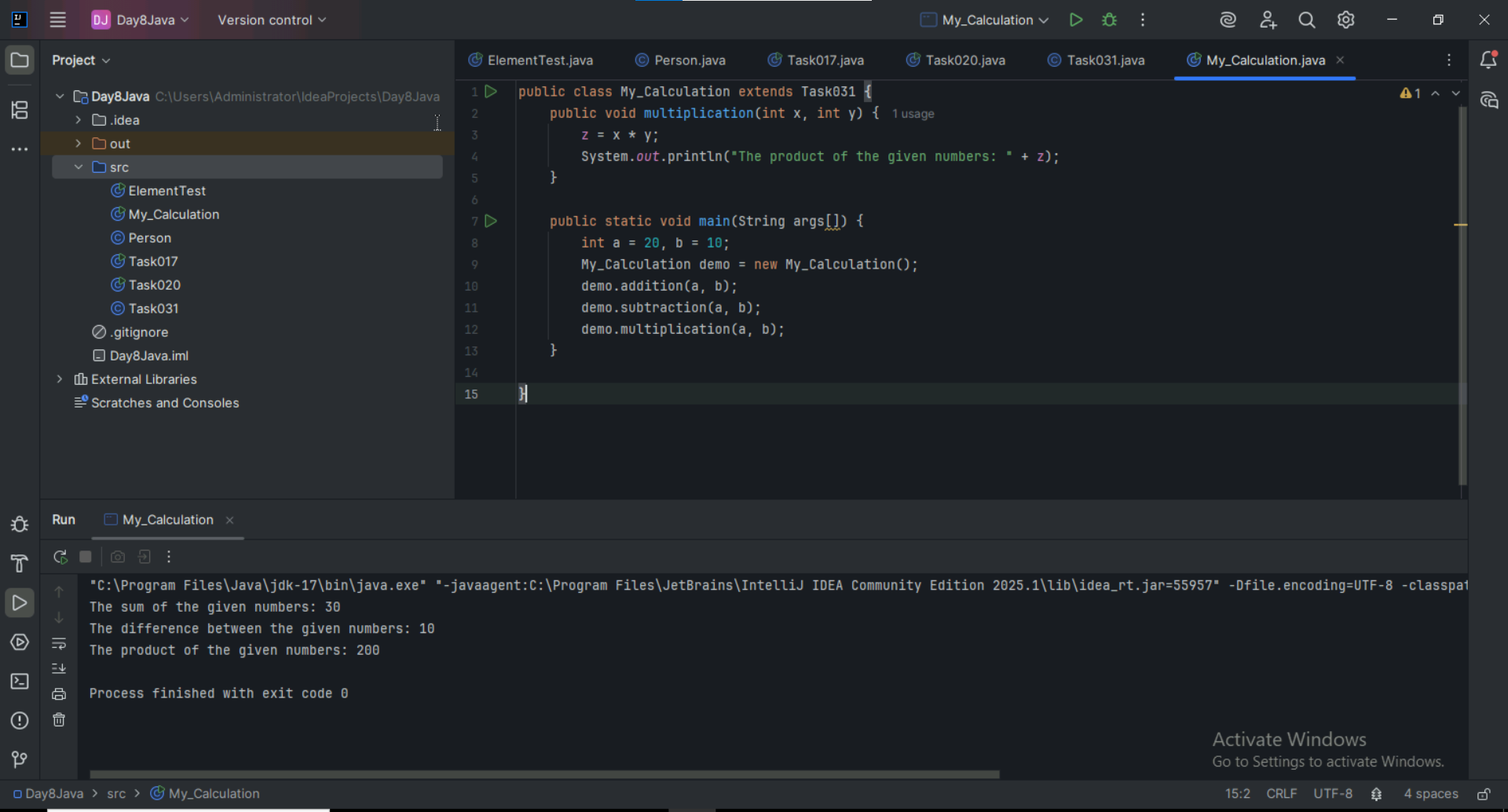
demo.Subtraction(a, b);

demo.multiplication(a, b);

}

}





Task 035 :

Type of parameters

Task 035

Void add(char x, char y){

Sout —-> x, y values

}

Void add(int x, int y) {

Sout —> x, y values

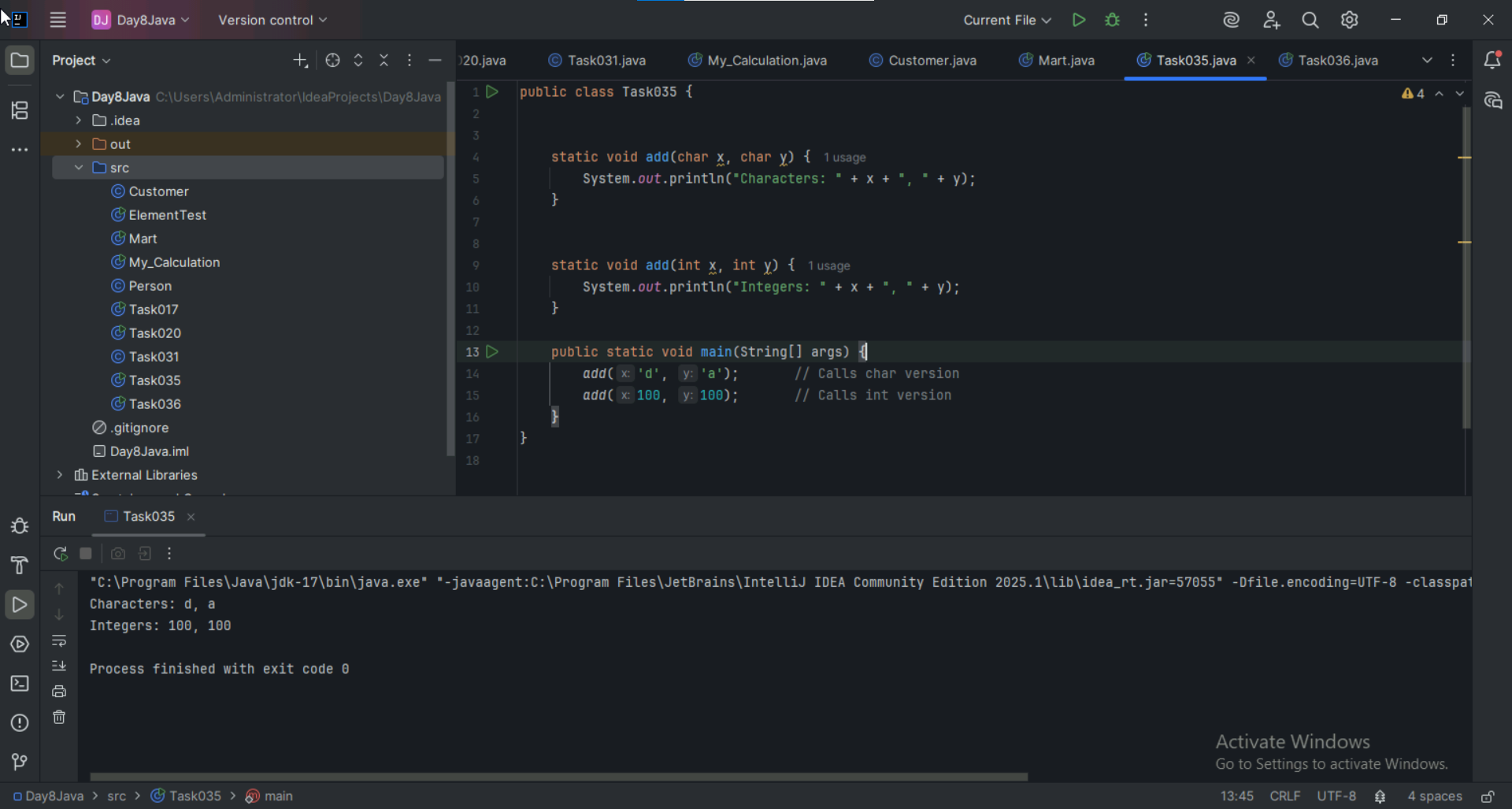
}

psvm(){

add(‘d’, ‘a’);

add(100, 100);

}



Sequence of Parameters

Task 036

Void add(int x, float y){

Sout → x, y values

}

Void add(float x, int y){

Sout → x, y

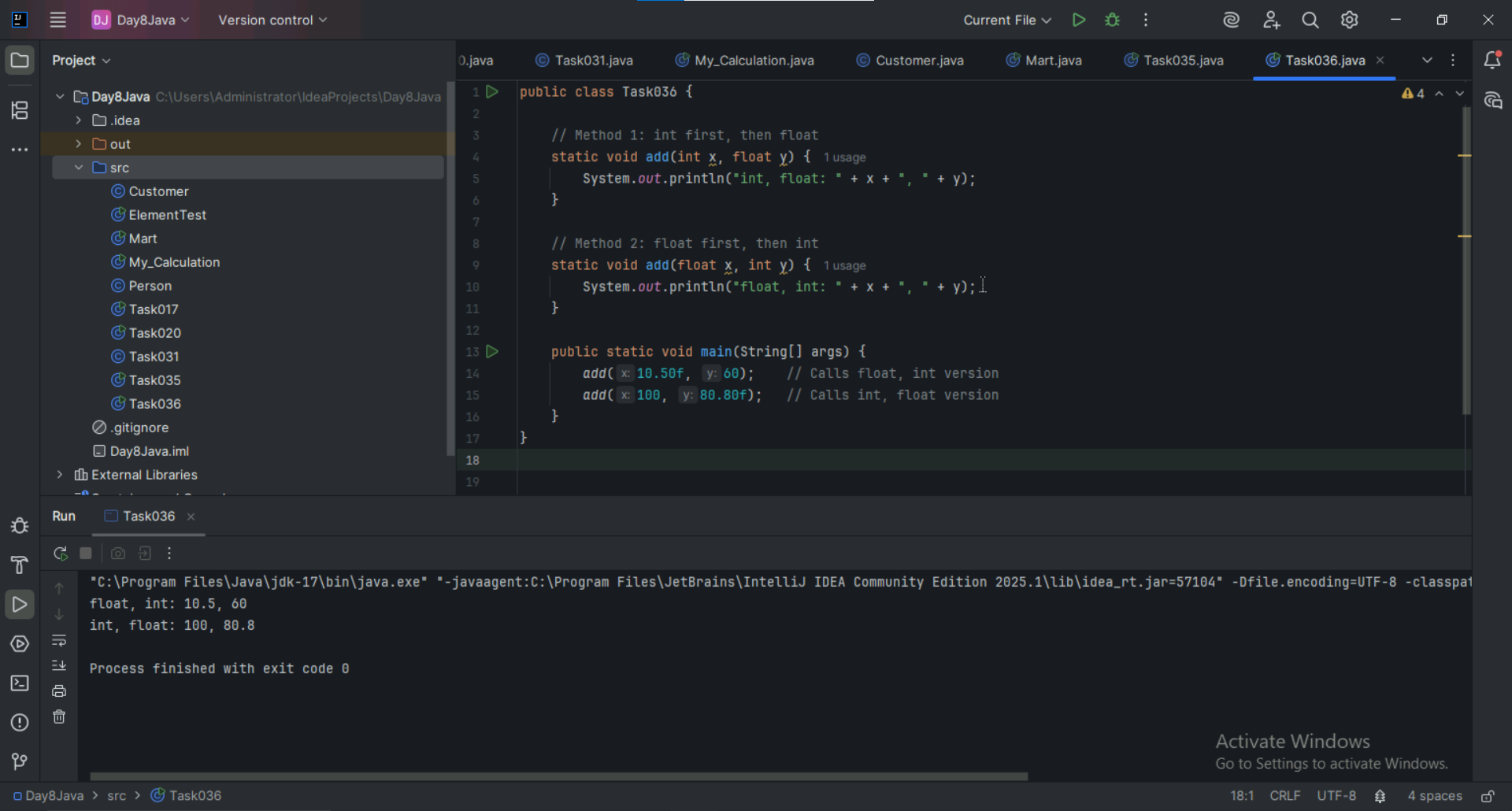
}

psvm(){

add(10.50f, 60);

add(100, 80.80f)

}



Task 034

Void add(int x, int y){

Sout —> x and y values

}

Void add(int x, int y, int z){

Sout —-> x, y, z values

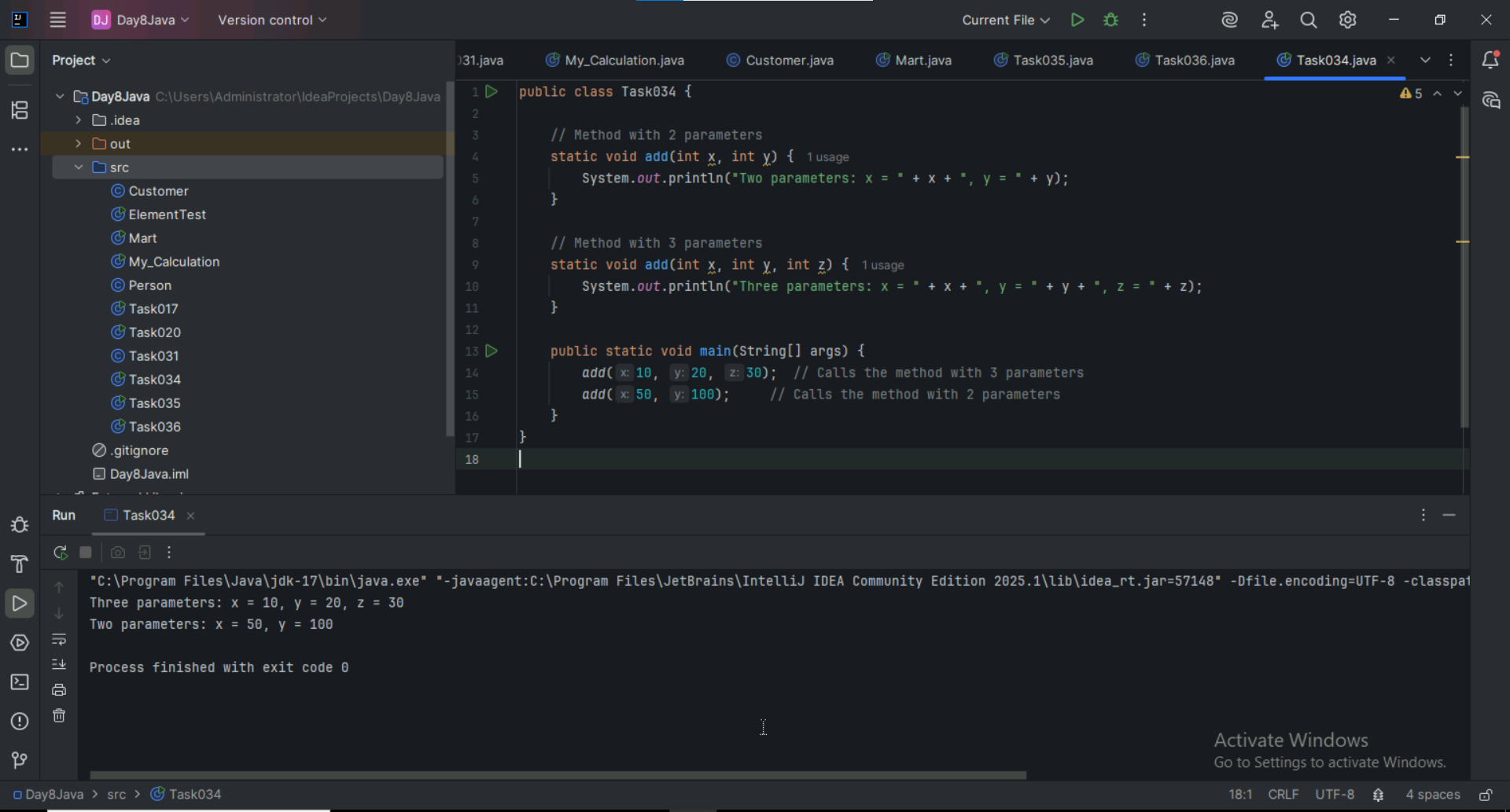
}

psvm(){

add(10,20,30);

add(50,100);

}



Task 037:

Class Employee{

Private int pwd;

Protected int Salary;

Public int empid:

employee(){ // constructors are methods having same name as class name (we have in c++)

}

~employee(){// destructors used in c++ but not in java

}

}

Class Hr extends Employee {

super.pwd = 1254; //===============> ??????

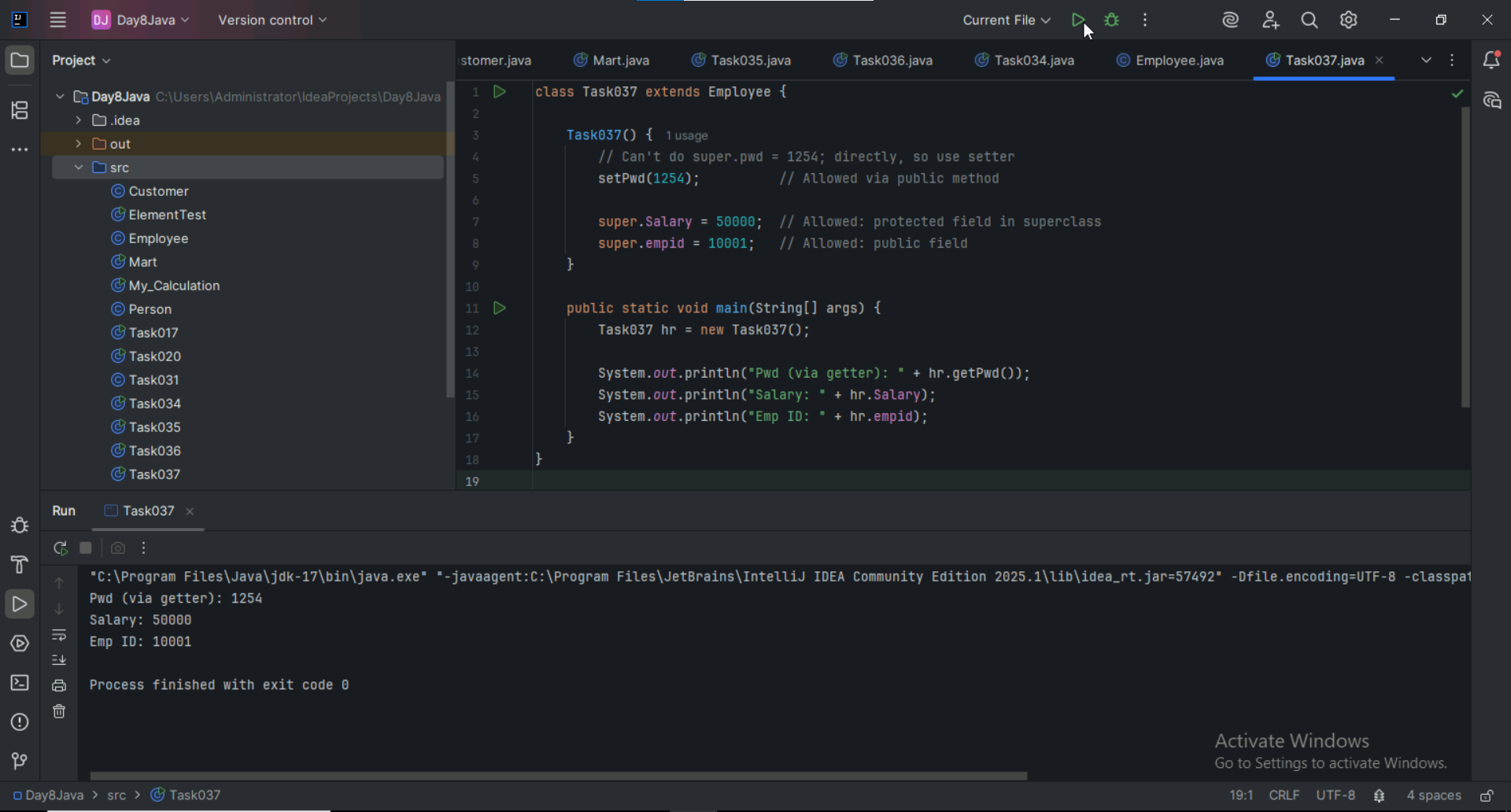
super.Salary = 50000; //==================> ?

Super.empid = 10001; // ======================>?

psvm(){

}

}



Task 038

/\* File name : AbstractDemo.java \*/

Public class AbstractDemo {

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() {

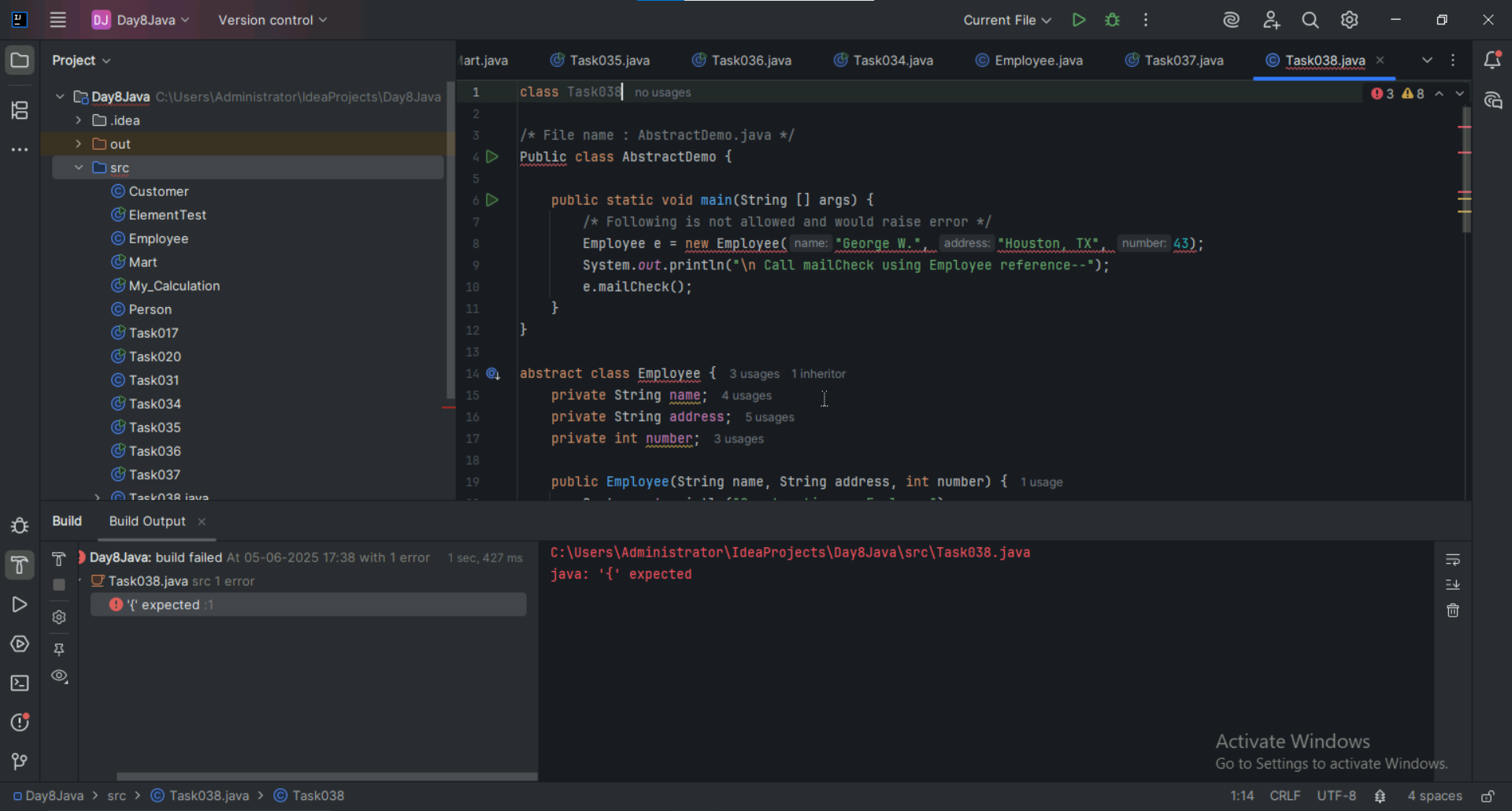
return number;

}

}

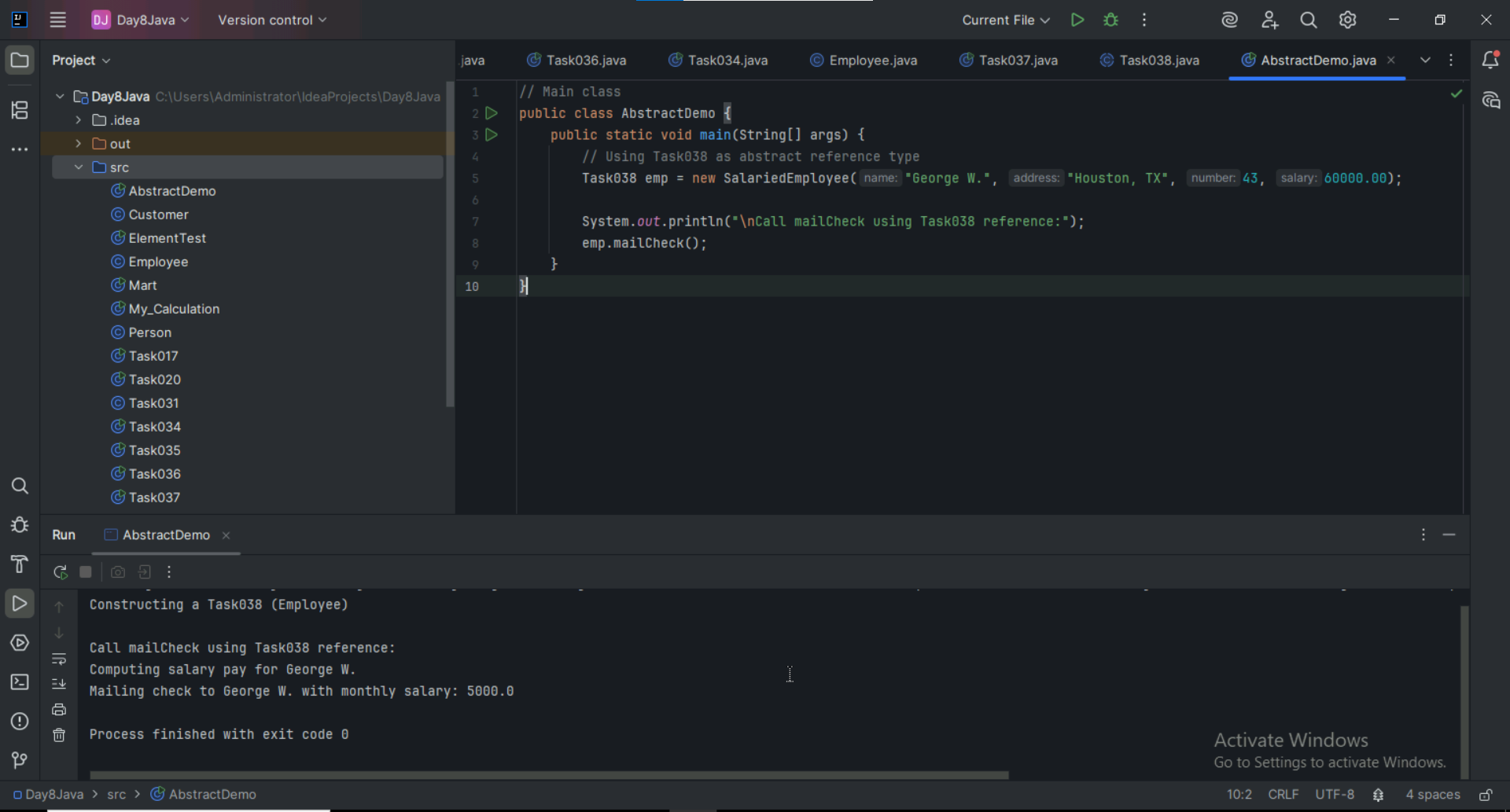
\*\*

It is throwing error cauz abstract class cannot be instantiated, they are meant to be inherited.



Task 039 :

Rewrite the above code to give output without errors.



Task 040

// Working of Abstraction in Java

abstract class Gadgets {

abstract void turnOn();

abstract void turnOff();

}

// Concrete class implementing the abstract methods

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.");

}

}

// Main class to demonstrate abstraction

public class Main {

public static void main(String[] args) {

Gadgets remote = new TVRemote();

Gadgets remote = new ACRemote();

remote.turnOn();

remote.turnOff();

Gadgets remote = new FanRemote();

Gadgets remote = new CoolerRemote();

remote.turnOn();

remote.turnOff();

}

}

