Task 1:

What do you understand by exceptions?

Exceptions in Java are unexpected events or errors that happen while a program is running. They interrupt the normal flow of the program. For example, if we try to divide a number by zero or access an array element that doesn’t exist, the program throws an exception.

Java provides a way to handle these exceptions using try, catch, and finally blocks so that the program doesn’t crash suddenly. Handling exceptions properly makes the program more reliable and user-friendly.

So, in simple terms, exceptions help us manage errors in a safe and structured way during program execution.

Task 2:

What are the categories of Exceptions do we have in Java? What are they?

There are 2 main categories of exceptions in Java:

1. Checked Exceptions
2. Unchecked Exceptions

Task 3:

Can you try the below code snippet and let me know which kind of exception is this

// Java program to demonstrates handling

// the exception using try-catch block

import java.io.\*;

class Geeks {

public static void main(String[] args)

{

int n = 10;

int m = 0;

try {

// Code that may throw an exception

int ans = n / m;

System.out.println("Answer: " + ans);

}

catch (ArithmeticException e) {

// Handling the exception

System.out.println(

"Error: Division by zero is not allowed!");

}

finally {

System.out.println(

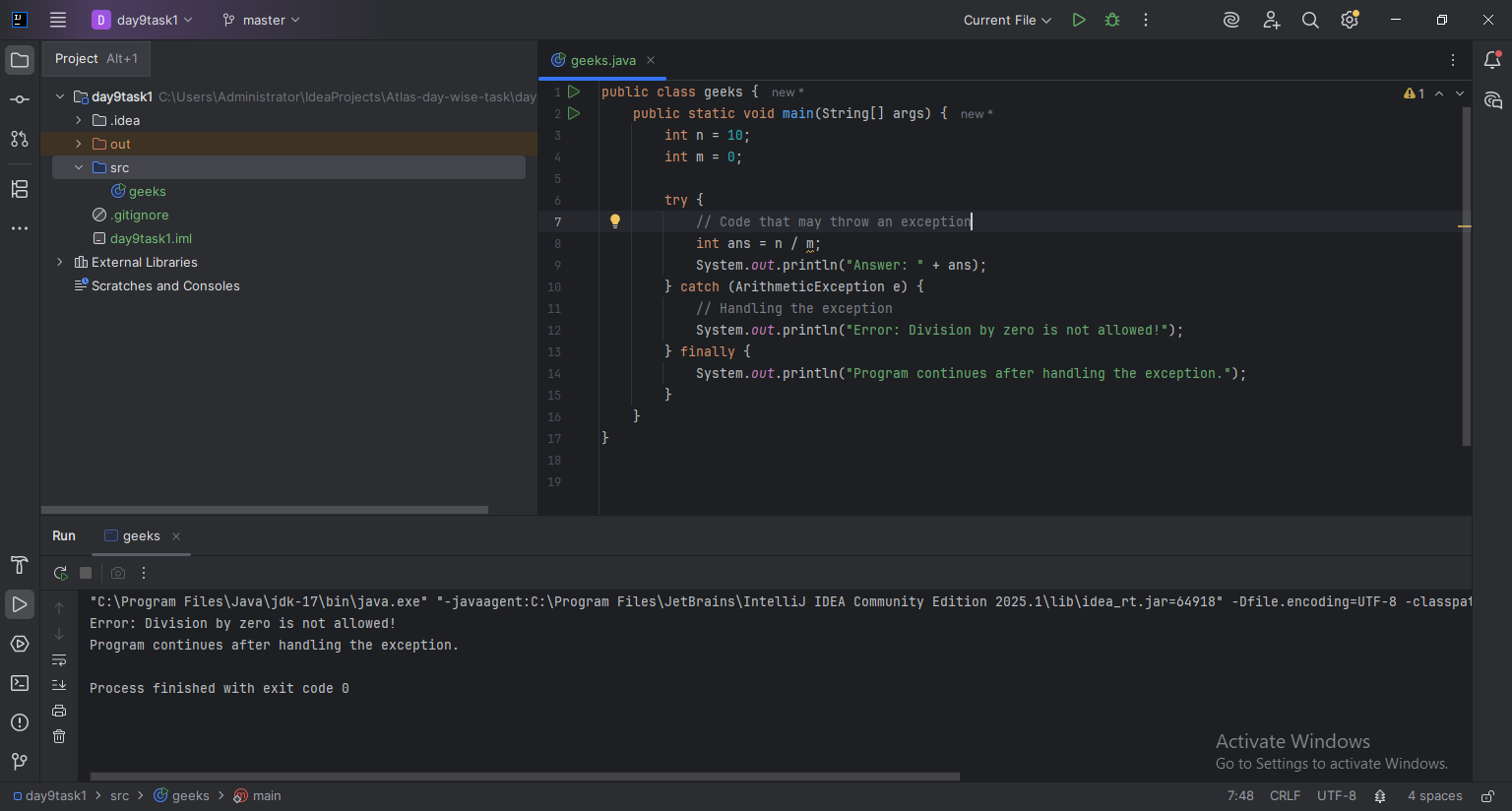
"Program continues after handling the exception.");

}

}

}

Answer : Unchecked Exception



Task 4:

List of checked and unchecked exceptions.

Sure! Here’s a simple student-style answer for Task 4: List of checked and unchecked exceptions:

### **Checked Exceptions**

These are checked at compile-time and we must handle them using try-catch or throws.

Examples:

1. IOException
2. FileNotFoundException
3. SQLException
4. ClassNotFoundException
5. InterruptedException

### **Unchecked Exceptions**

These are checked at runtime and the compiler doesn’t force us to handle them.

Examples:

1. ArithmeticException
2. NullPointerException
3. ArrayIndexOutOfBoundsException
4. NumberFormatException
5. IllegalArgumentException

Task 5:

Try with Multiple catch blocks …. Execute the below code snippet n display the out .. along with reason..

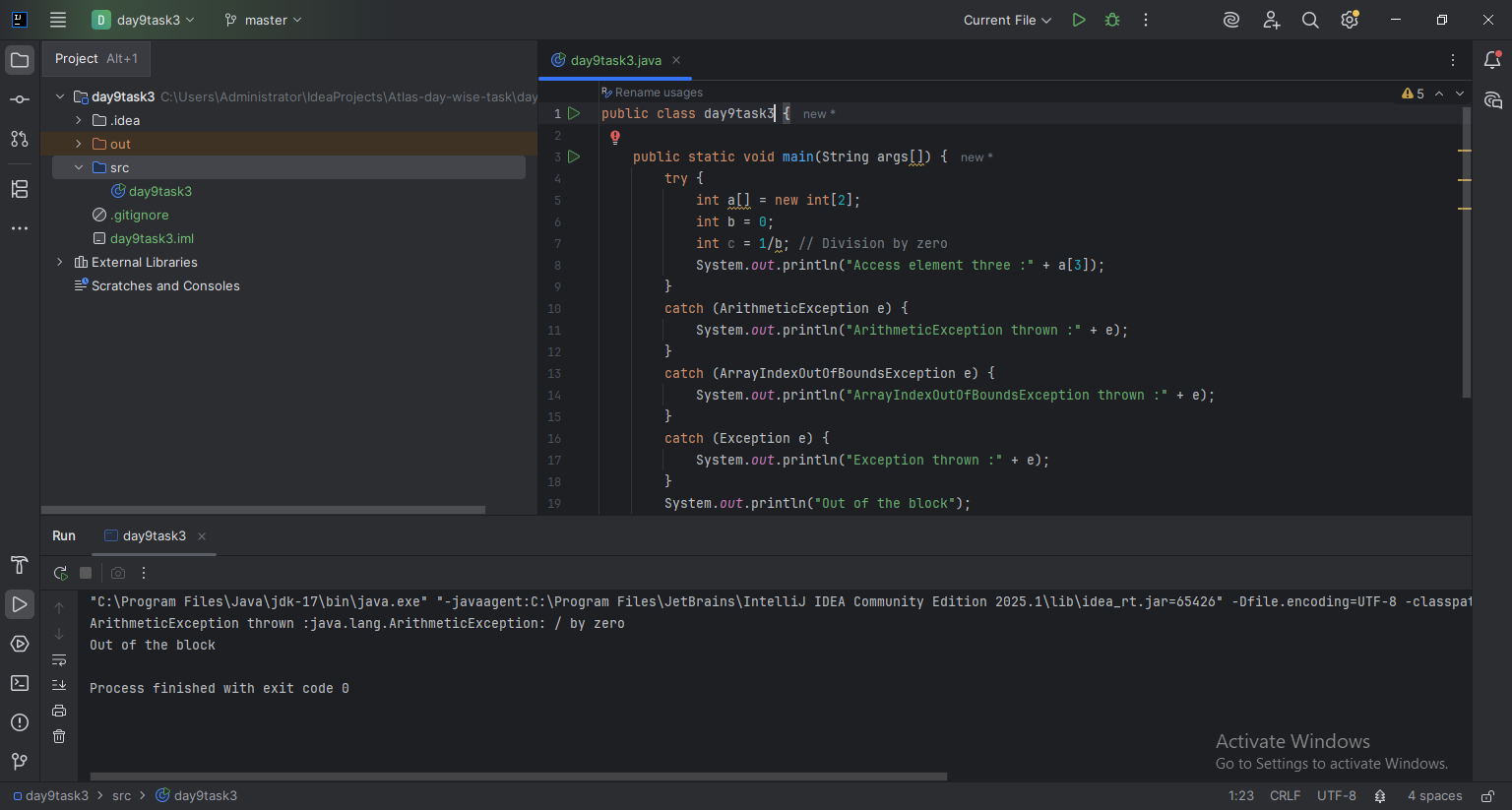
### 

### **Reason:**

* Multiple catch blocks help handle different types of exceptions separately.
* The specific exception (ArithmeticException) was not caught in the first catch, so it goes to the more general Exception catch block.
* After handling, the program continues and prints "Out of the block".

Task 6:

### 



### **Reason:**

1. int c = 1 / b; → b = 0, so this line causes an ArithmeticException.
2. The try block immediately jumps to the first matching catch block.
3. The catch (ArithmeticException e) matches the thrown exception and is executed.
4. Code after the exception line inside try (like System.out.println("Access element three...") is not executed.
5. After the catch block, the program continues and prints:  
     
    "Out of the block"

Task 7:

In the below code we are having use multiple catch in a single statement: find the output and try to understand the code..

public class ExcepTest {

public static void main(String args[]) {

try {

int a[] = new int[2];

int b = 0;

int c = 1/b;

System.out.println("Access element three :" + a[3]);

}

catch (ArrayIndexOutOfBoundsException | ArithmeticException e) {

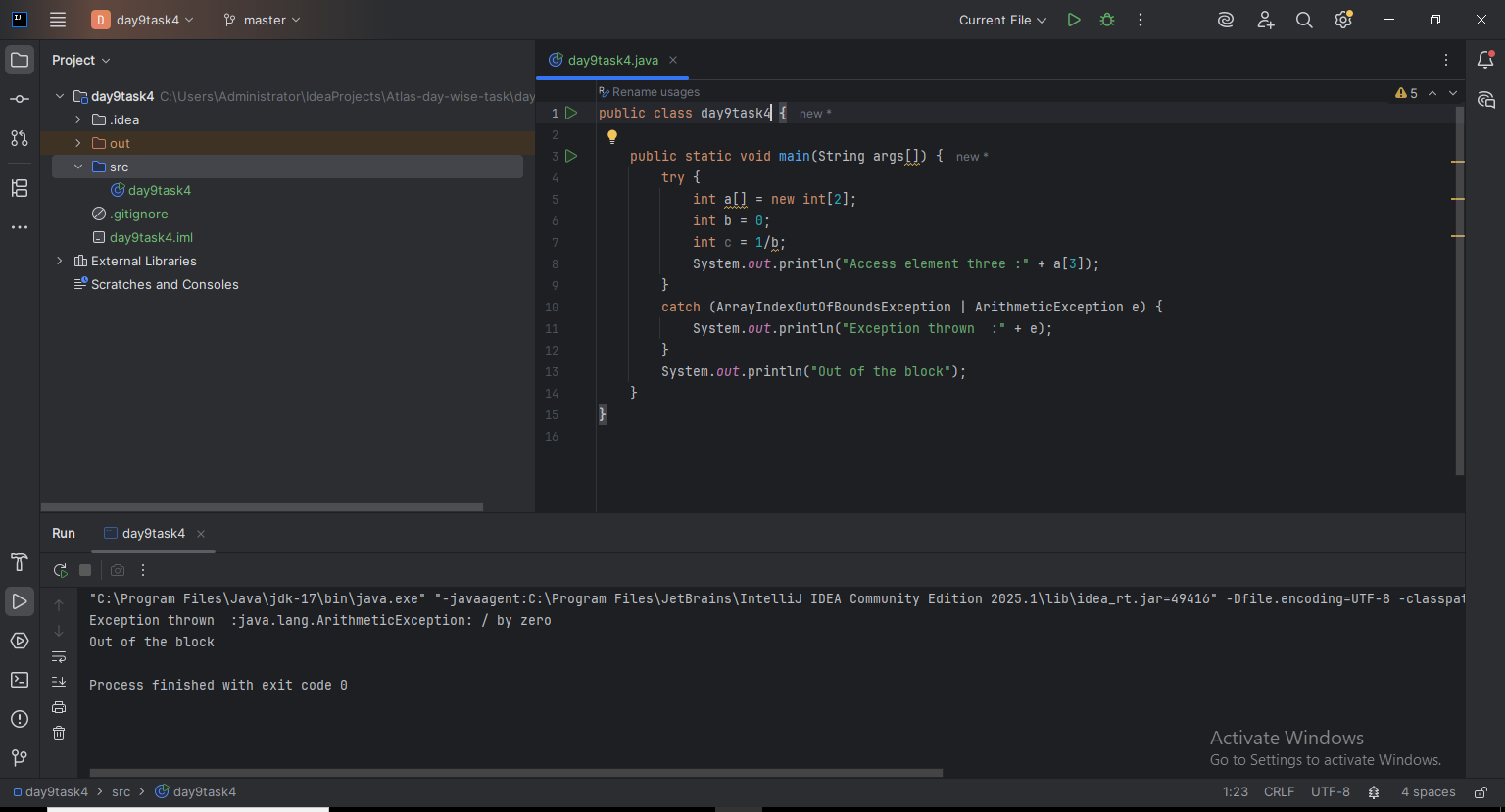
System.out.println("Exception thrown :" + e);

}

System.out.println("Out of the block");

}

}



Task 8:

public class ExcepTest {

public static void main(String args[]) {

try {

int a[] = new int[2];

try {

int b = 0;

int c = 1/b;

}catch(Exception e) {

System.out.println("Exception thrown: " + e);

}

System.out.println("Access element three :" + a[3]);

}

catch (ArrayIndexOutOfBoundsException e) {

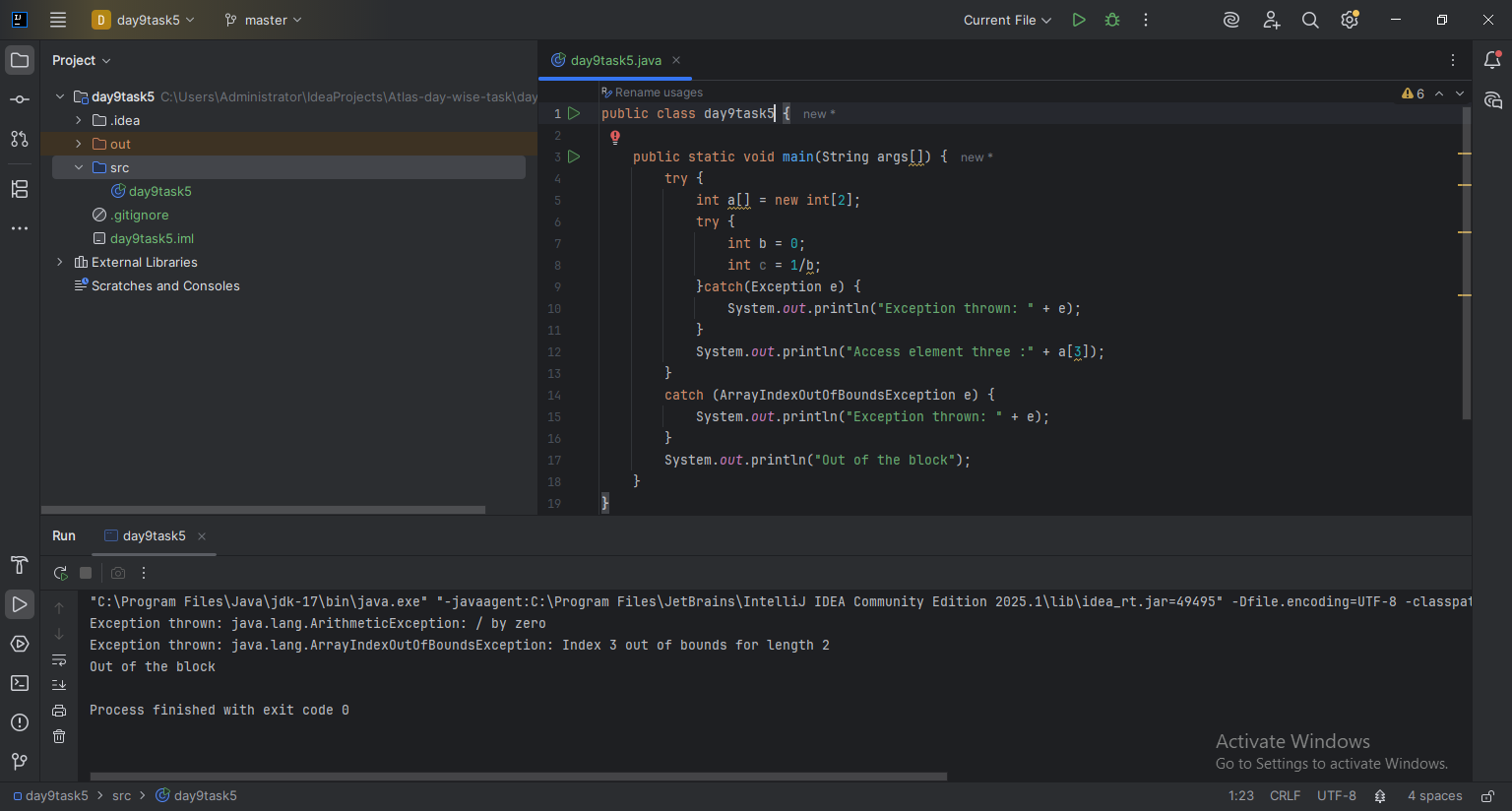
System.out.println("Exception thrown: " + e);

}

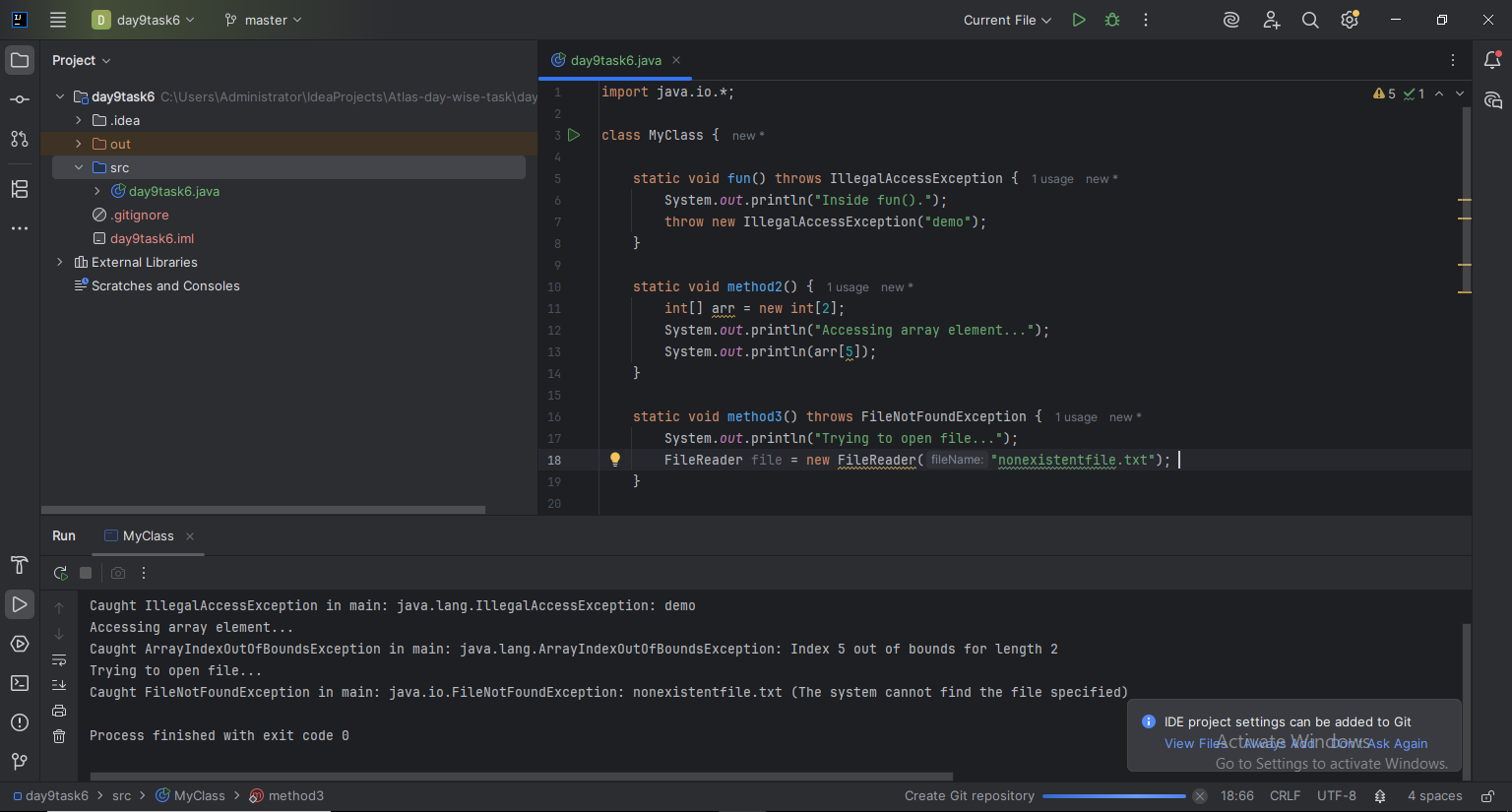
System.out.println("Out of the block");

}

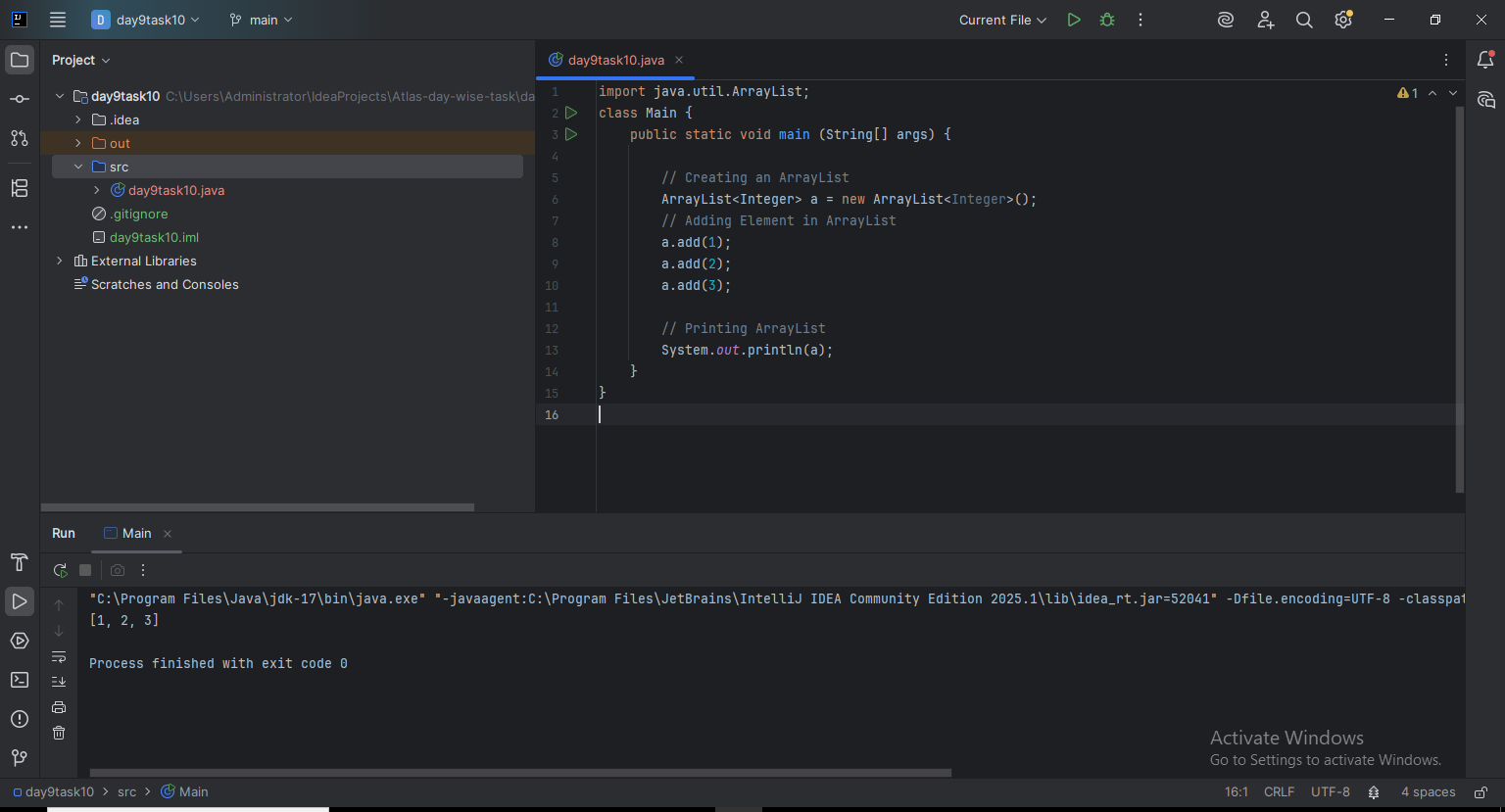
}



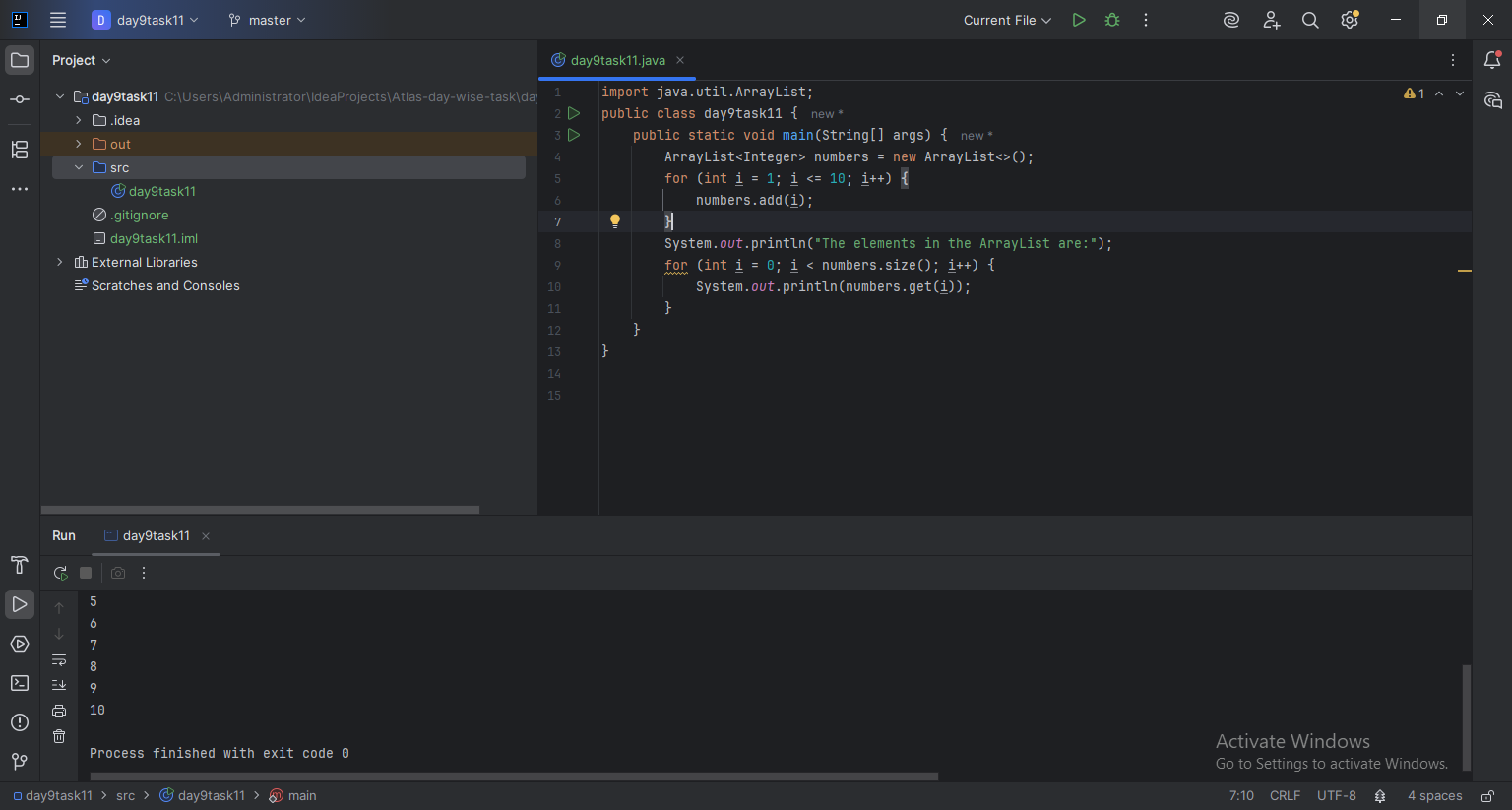
Task 9:



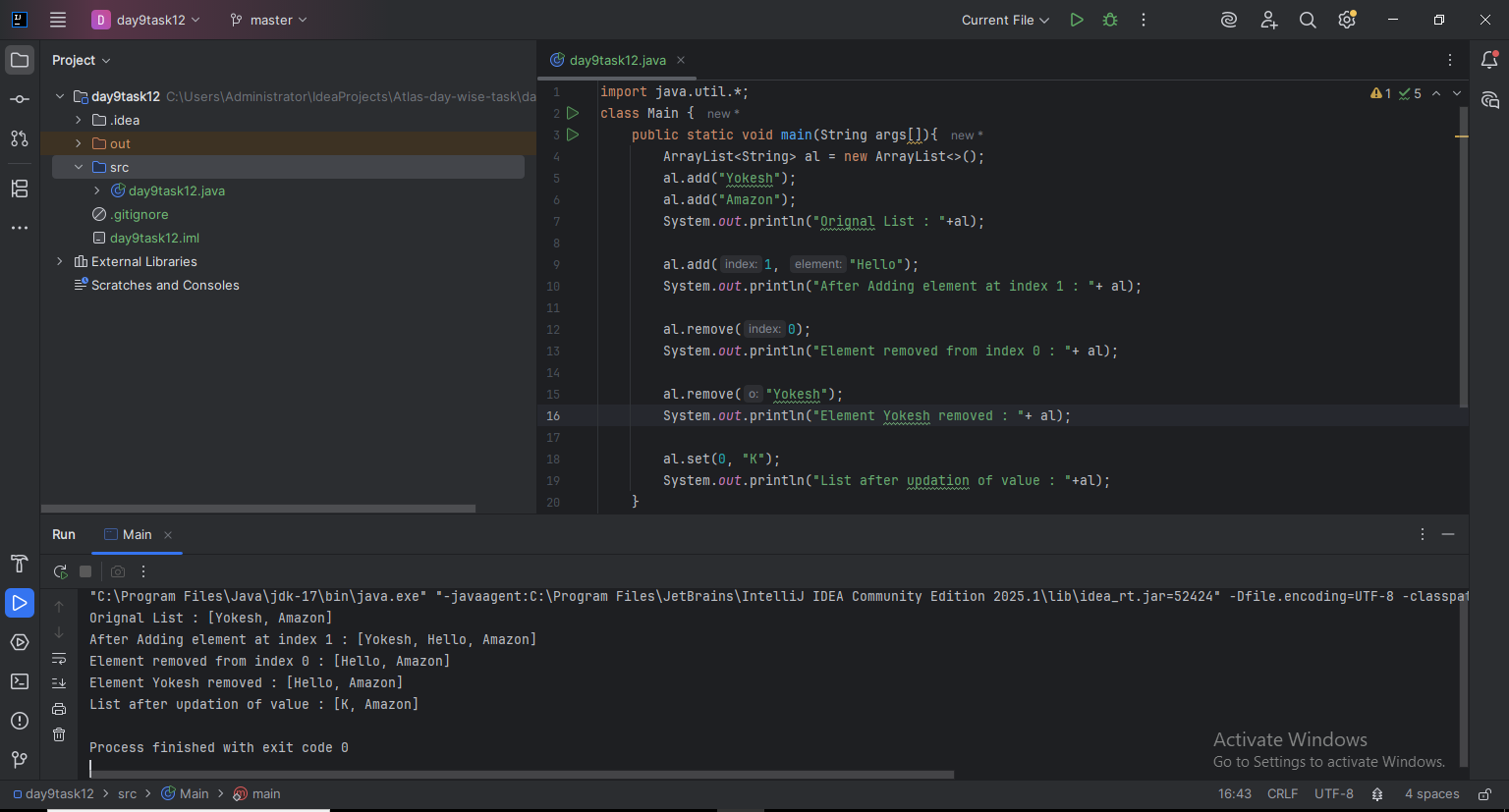
Task 10:



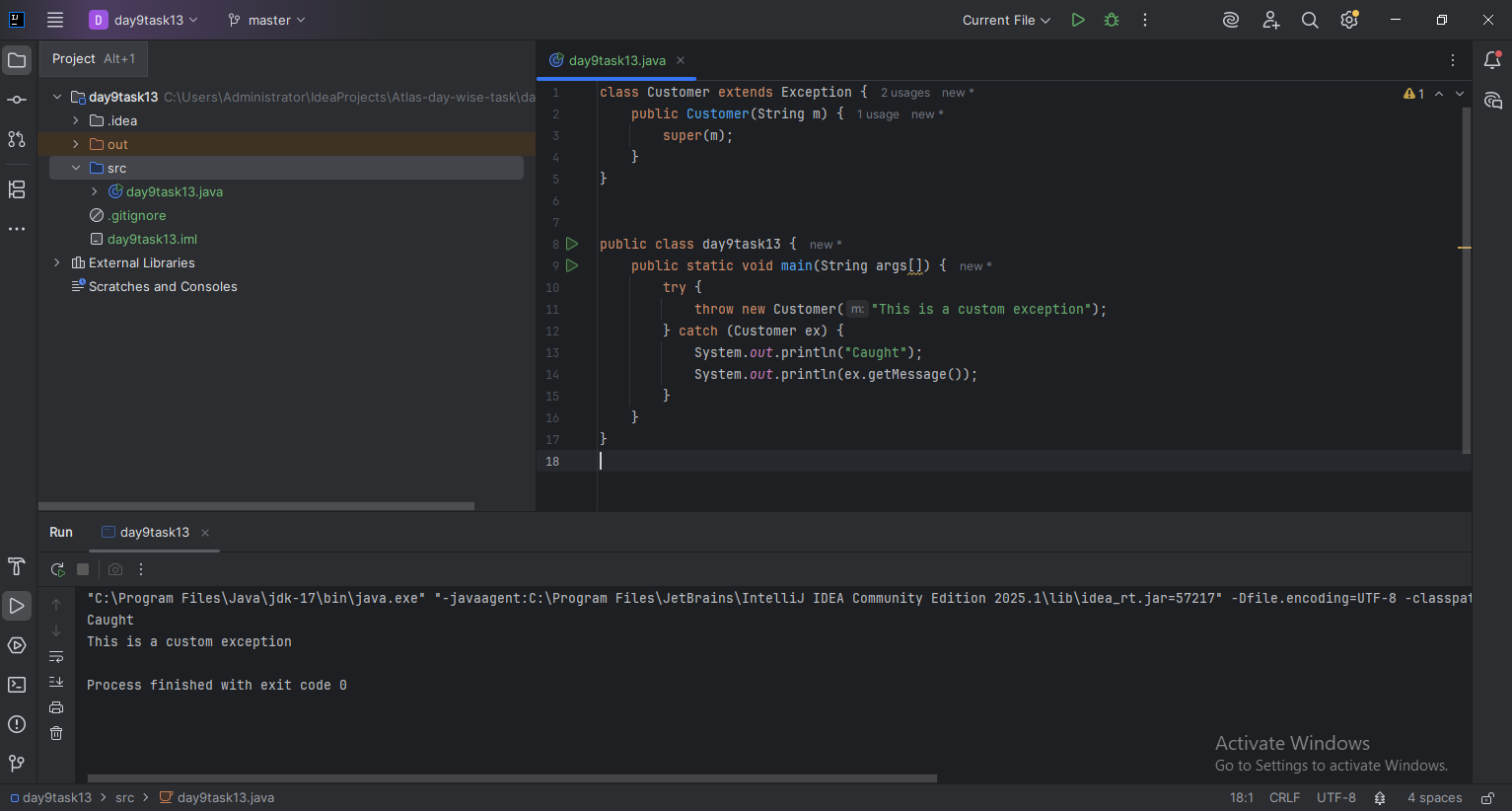
Task 11:



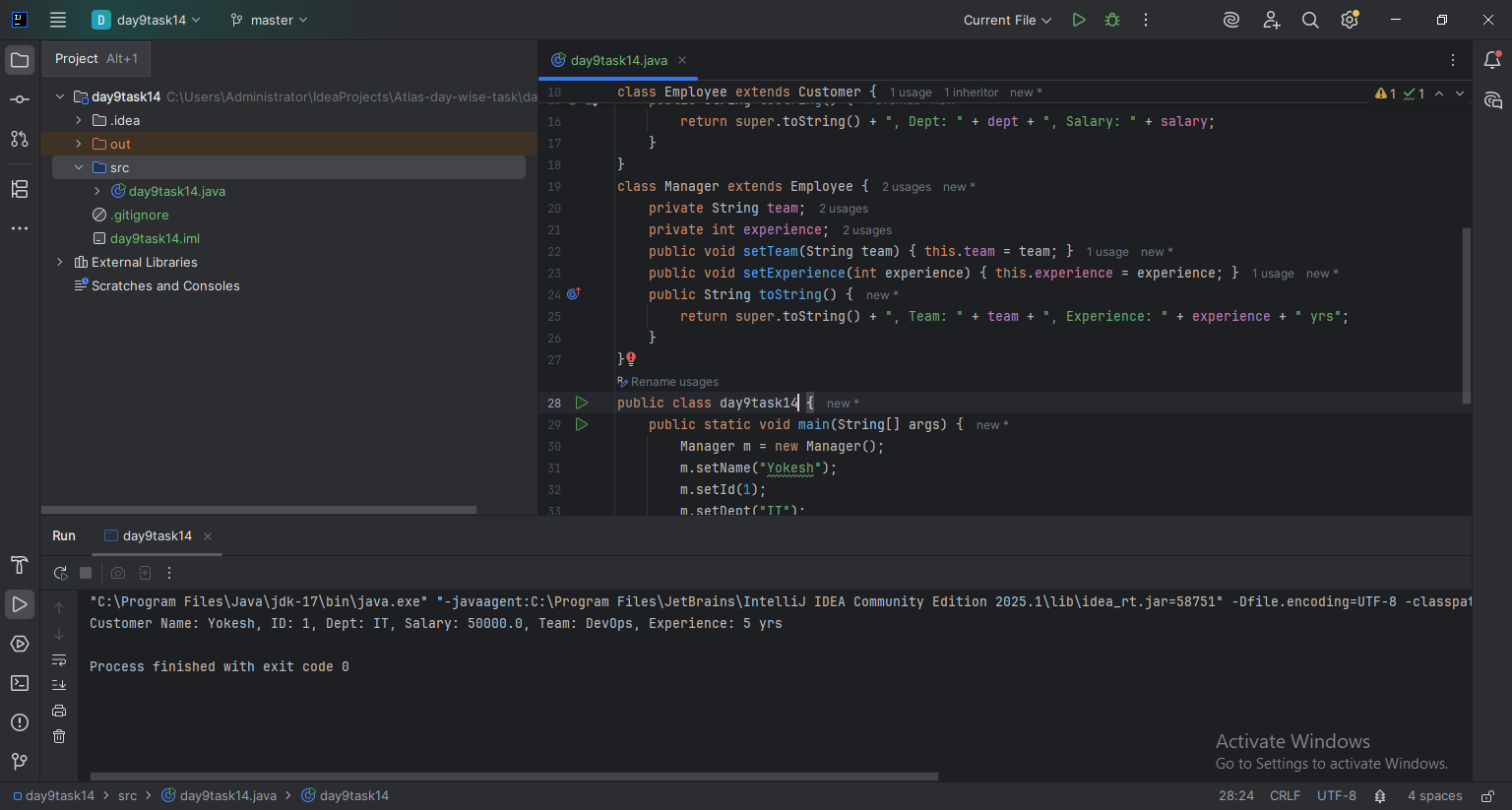
Task 12:



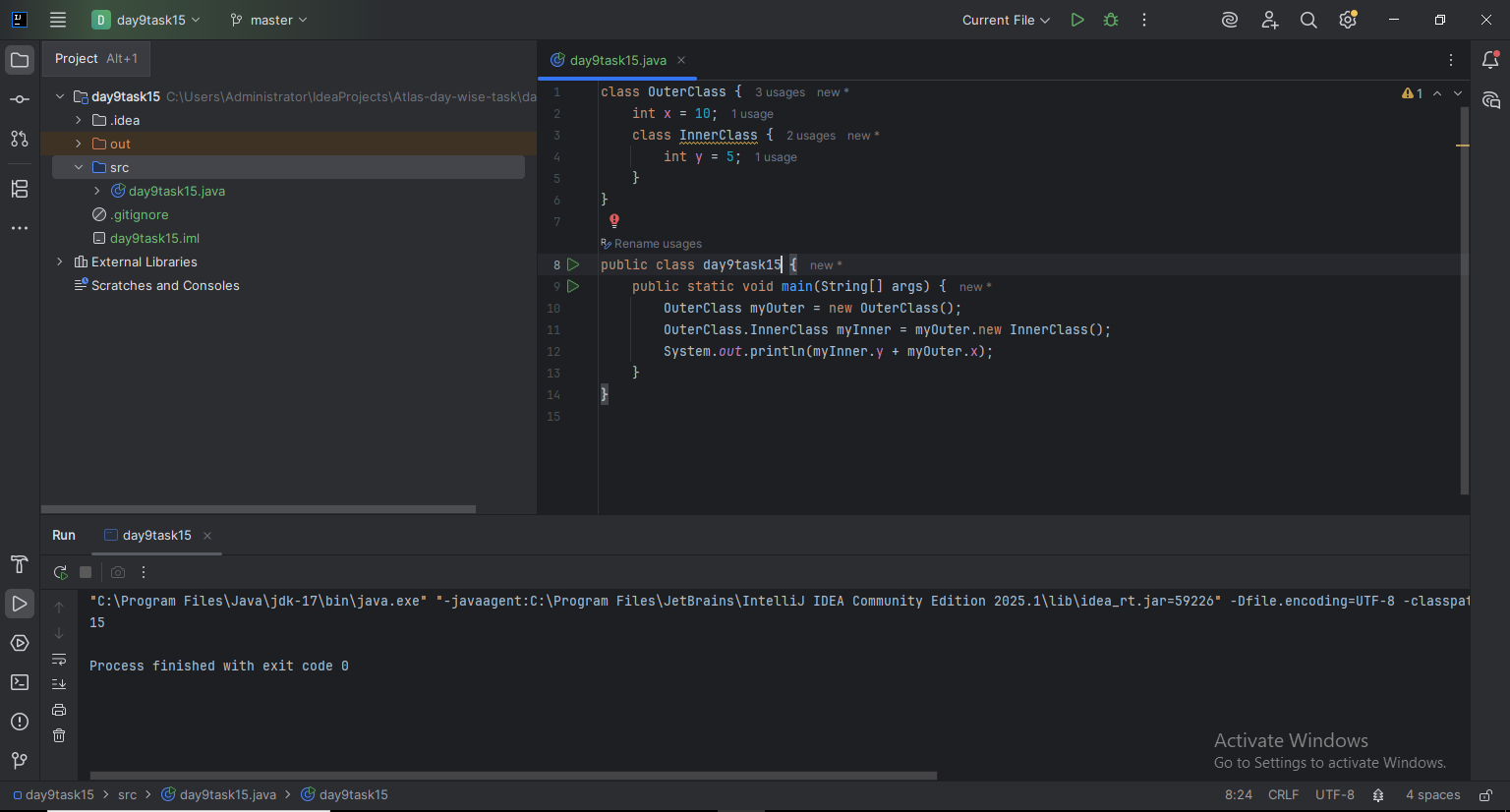
Task 13:



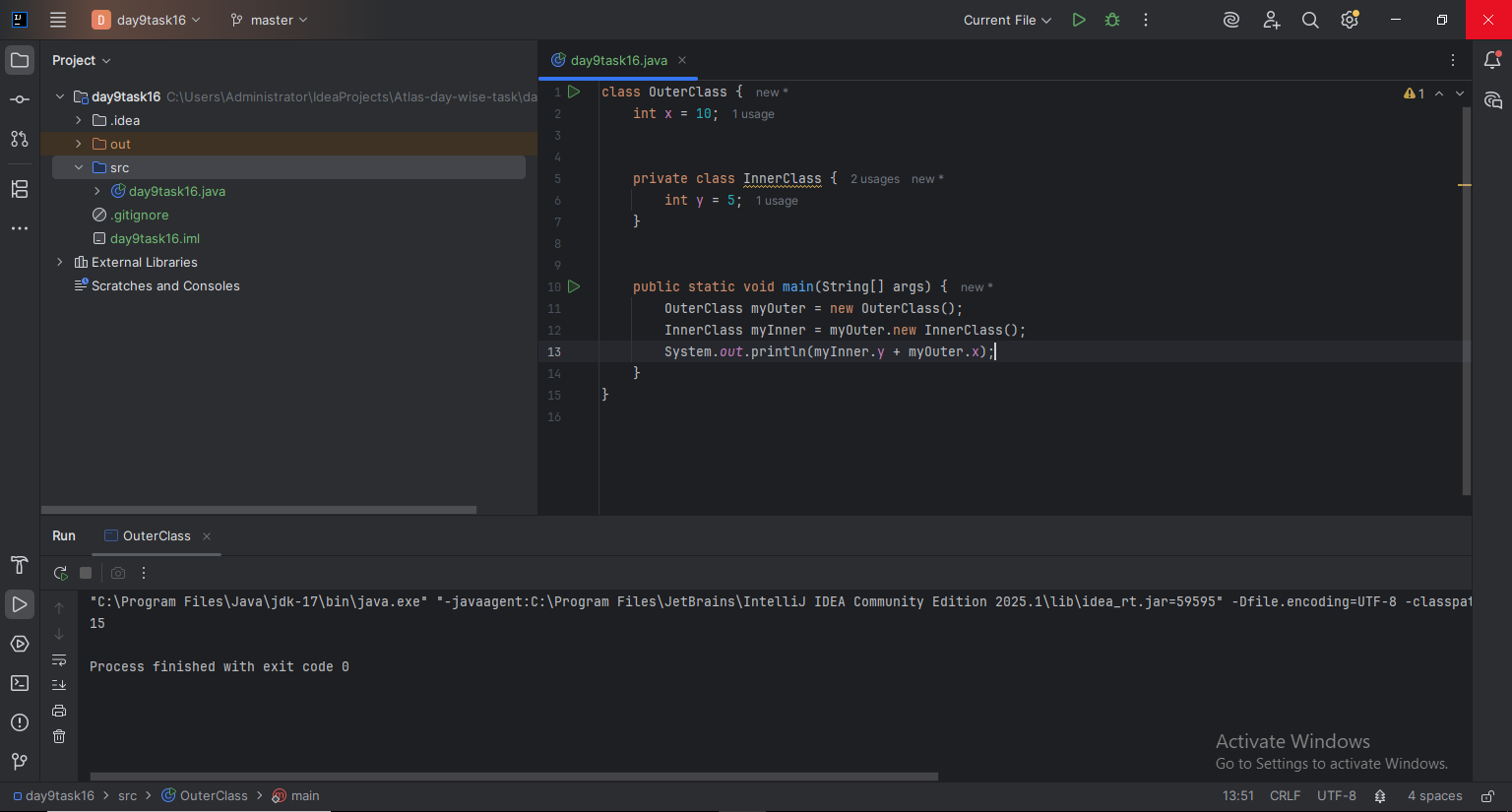
task 14:



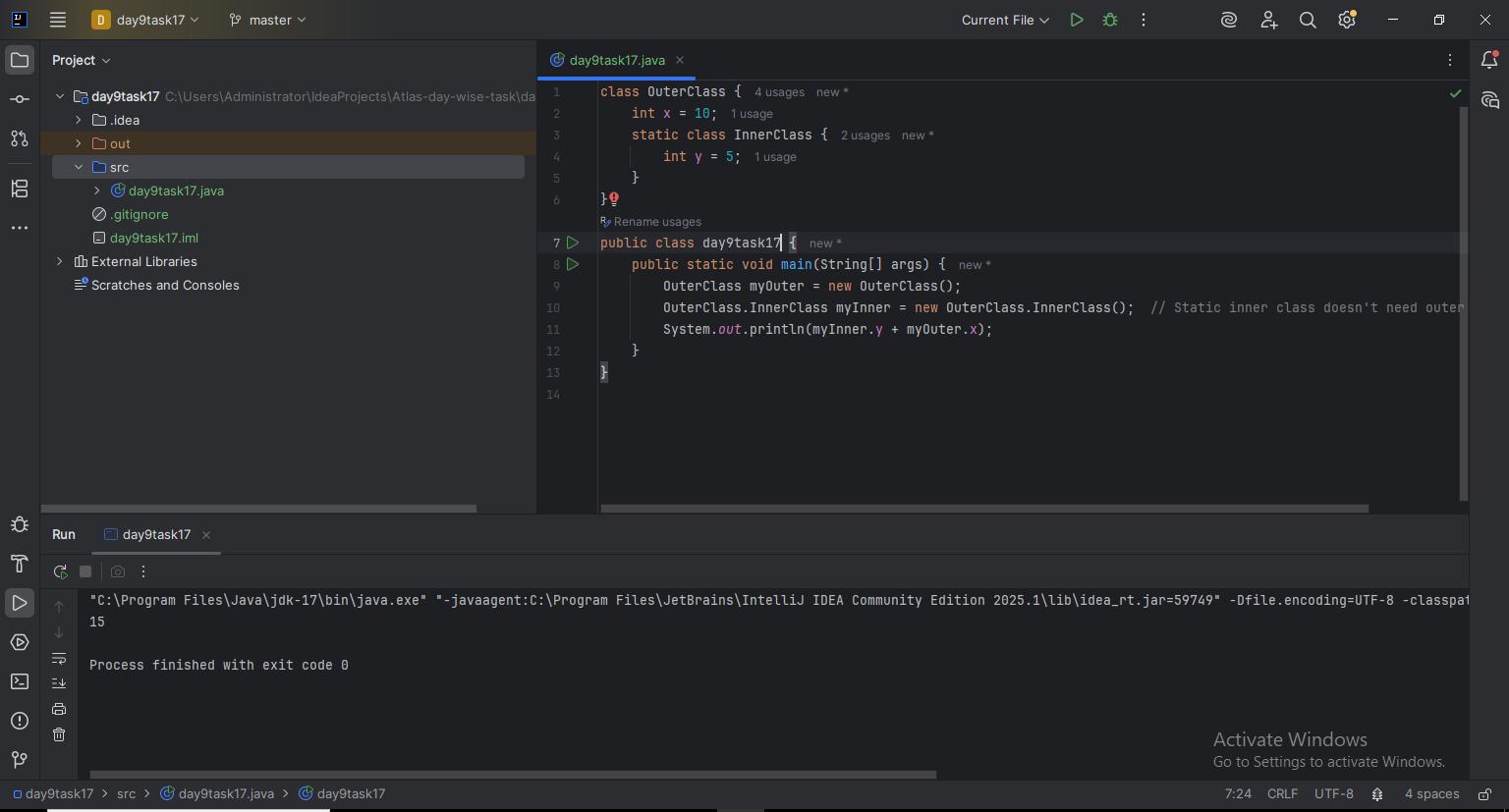
Task 15:



Task 16:



Task 17:



Task 18:

