**Date:** *11-Jun-2025*

**Name:** *Vivek Samant*

**Empid:** *109085619*

**alias:** *samantvs*

**Tasks:**

Task 1: What do you understand by exceptions?

In a normal flow of program if a condition is not accounted for in the logic, exceptions occur. In other terms, it is similar to finding what is not supposed to be there. This has a potential to abruptly terminate the flow of control.

Task 2: What are the categories of Exceptions do we have in Java? What are they?

There are basically two categories of exceptions:

1. Checked: this is during compile team. If all the syntax and declaration and the entire program is syntactically correct, compiler still checks for places where something could lead to exception (like when we work with division, division by 0 could lead to this). Hence, compiler by design forces the programmer to write that exception into the program.
2. Unchecked: this occurs during run time. Like if a user is trying to access 6th element of an array only having 5 elements leading to out of bound exception.

Task 003:

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

What is the output of the code..?

// 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!");

        }

catch (ArithmeticException e) {

            // Handling the exception

            System.out.println(

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

        }

        finally {

            System.out.println(

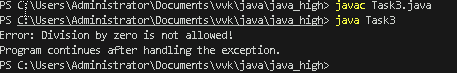
                "Program continues after handling the exception.");

        }

    }

}

5 min 10.37 to 10.42



Task 4: List of checked and unchecked exceptions.

Checked Exception: be design enforced by compiler. Must be explicitly handled or declared.

EX: IOException, FileNotFoundException, EOFException, InterruptedIOException, SQLException, ClassNotFoundException, InterruptedException, CloneNotSupportedException

RuntimeException – these are runtime exceptions, optinal handling done

ArithmeticException // Division by zero

NullPointerException // Null reference

IndexOutOfBoundsException

├─ ArrayIndexOutOfBoundsException

└──StringIndexOutOfBoundsException

IllegalArgumentException

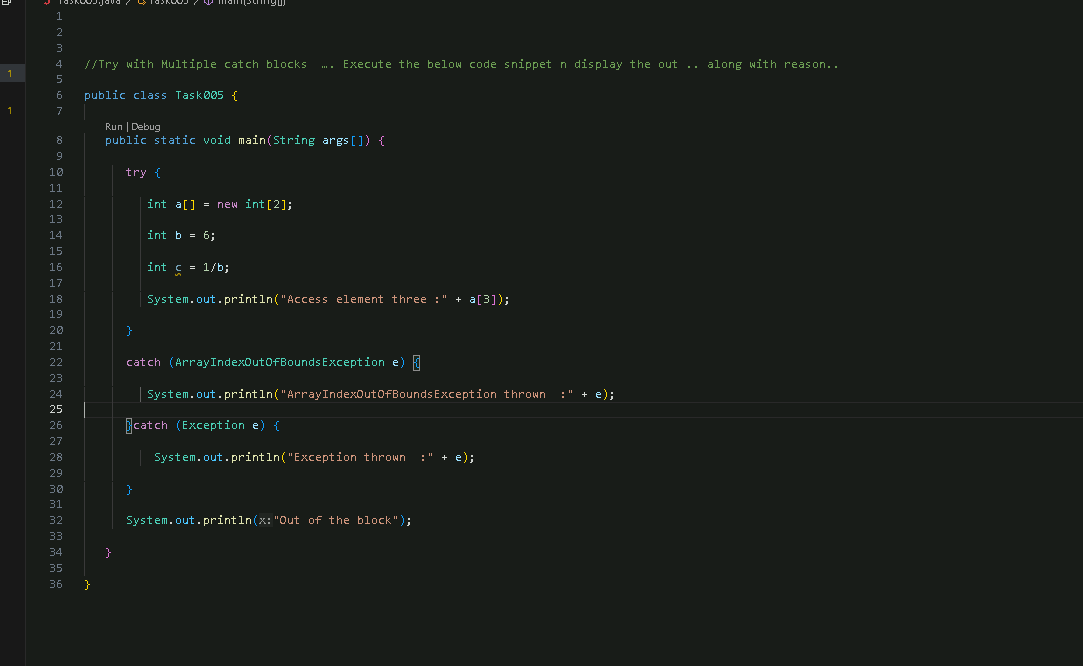
ClassCastException

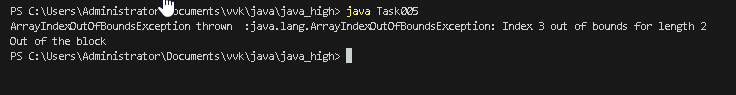
*Error* // very serious

├── StackOverflowError

├── OutOfMemoryError

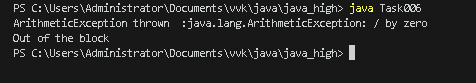
└── NoClassDefFoundError

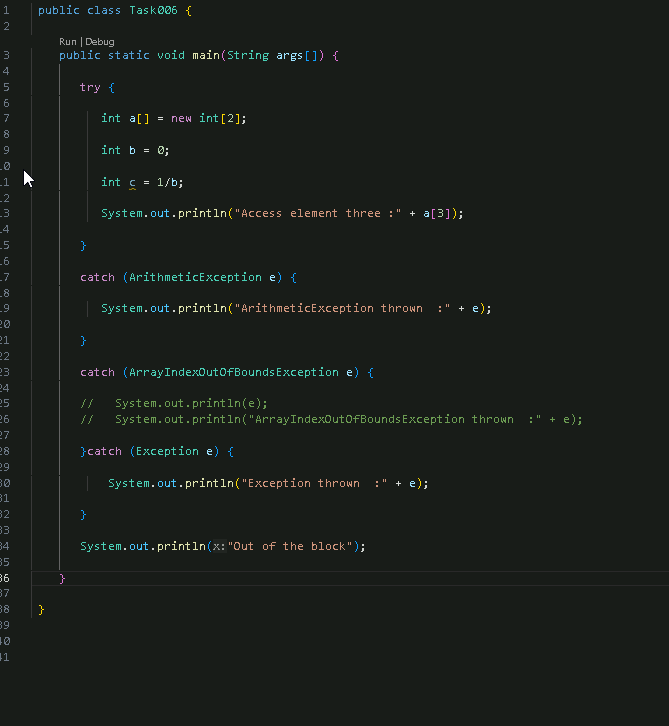
Task 5: Try with Multiple catch blocks  …. Execute the below code snippet n display the out .. along with reason..



.

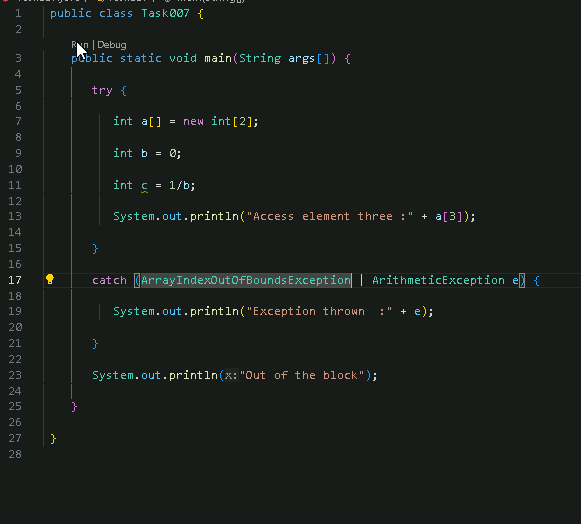
Task 6: What is the output of the below code… give your  reason for the output



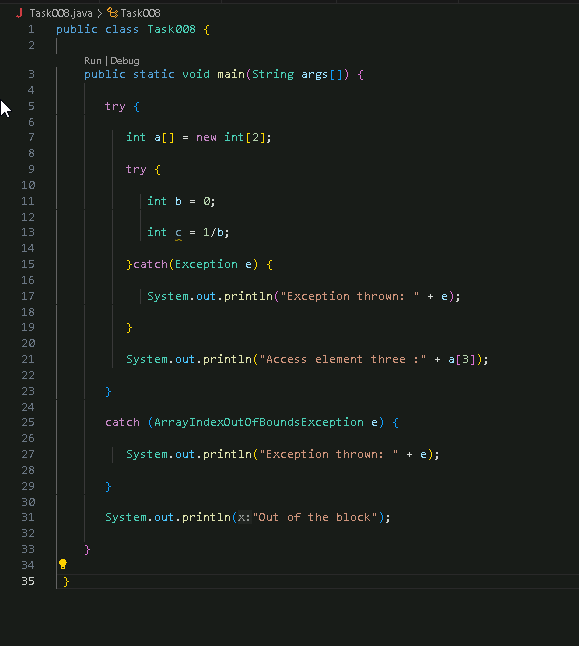


First arithmetic exception is caught and catch ends. Final sout is outside this, hence prints.

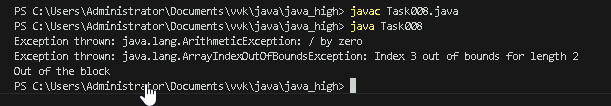
Task 7: In the below code we are having use multiple catch in a single statement: find the output and try to understand the code.



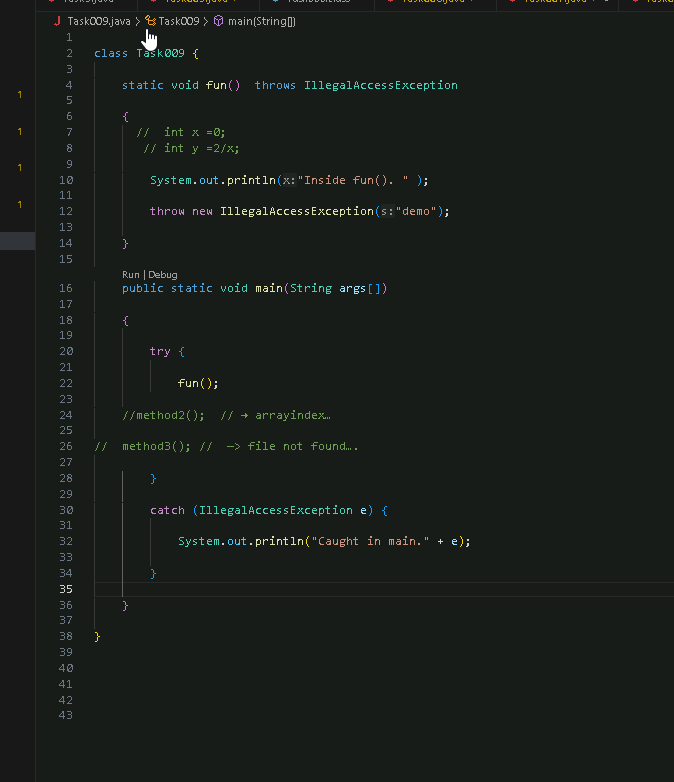
Task 008:

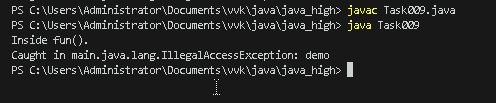




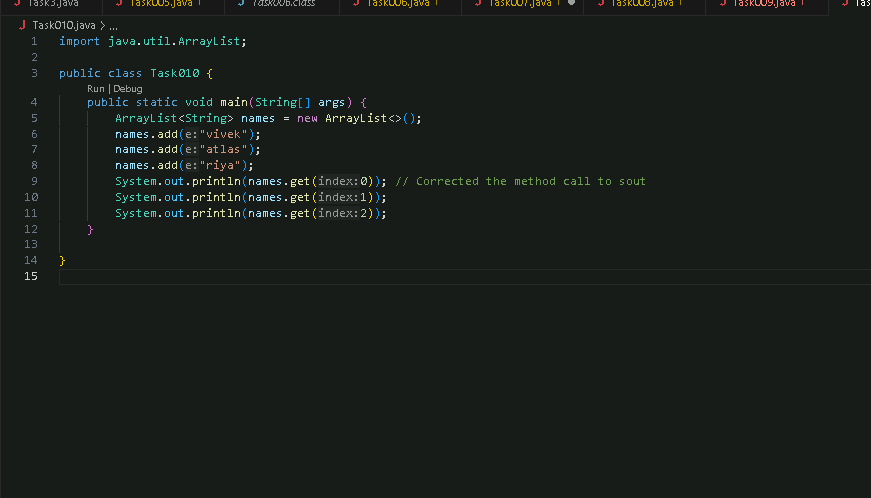


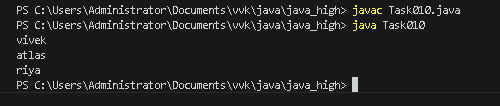
**Task 009**



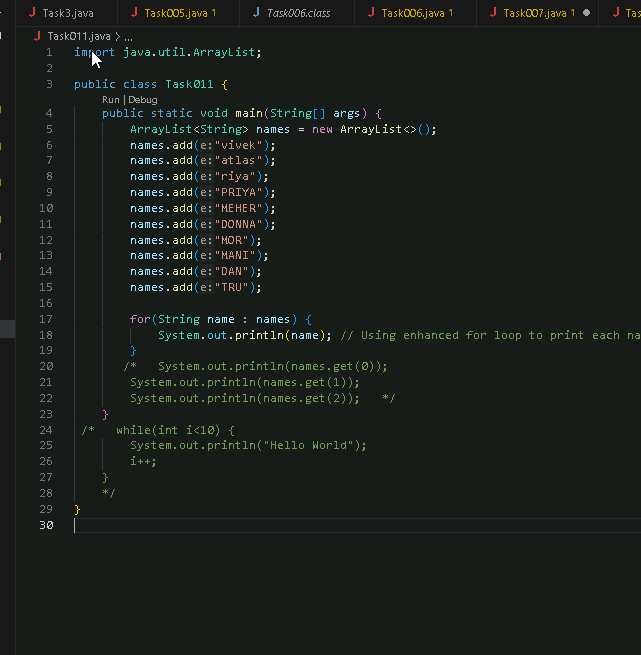


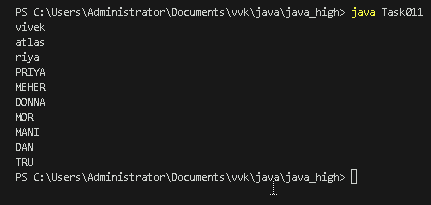
Task 010:



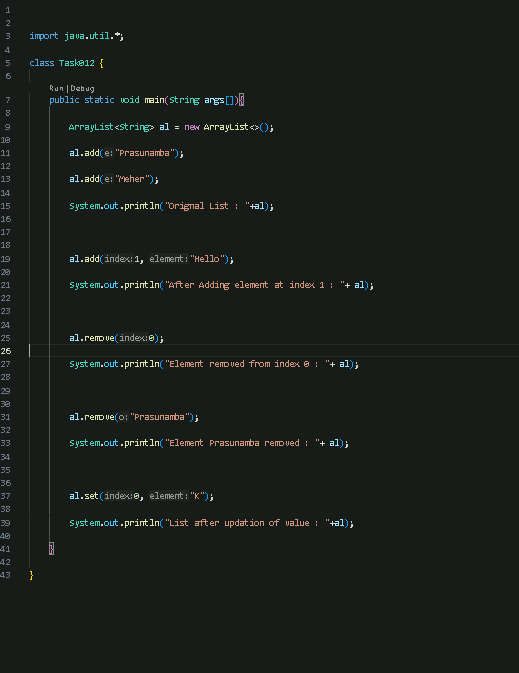


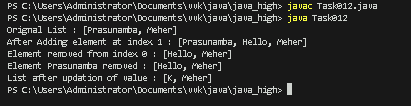
Task011



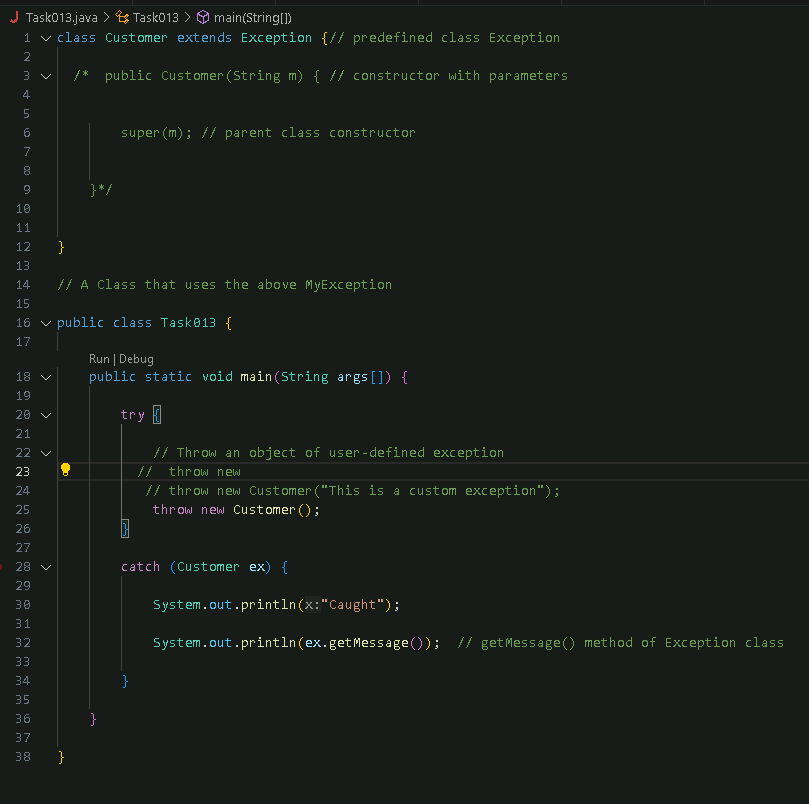


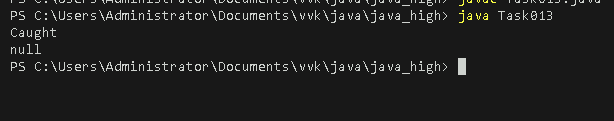
Task 012:





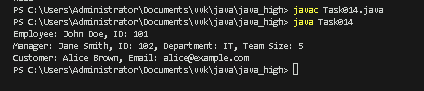
**Task 013:**



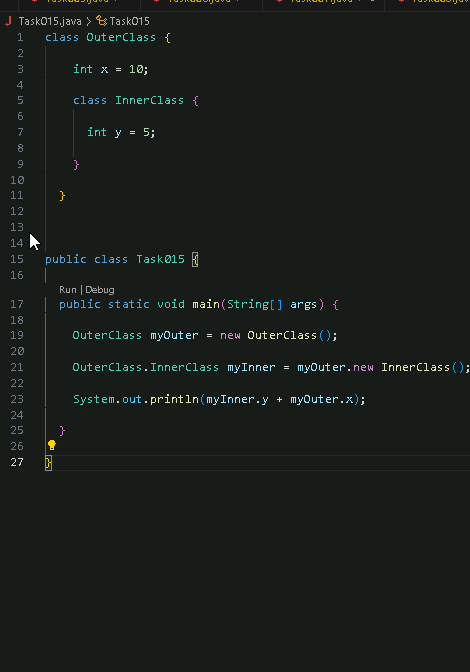


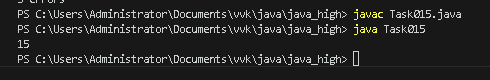
**Task014:**



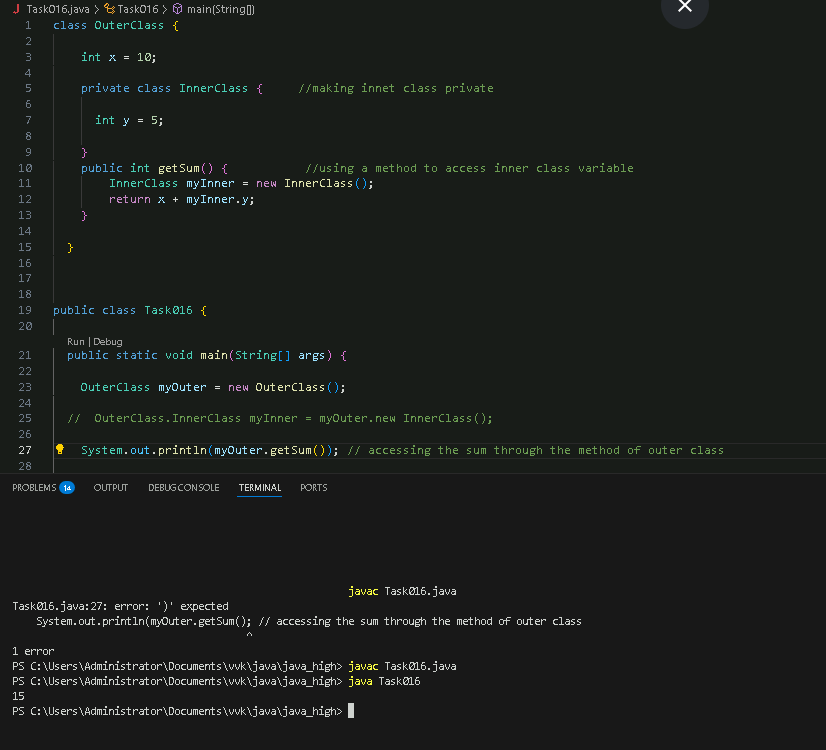


**Task 015:**

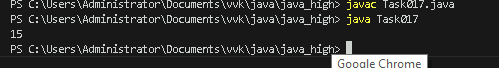




**Task016:**

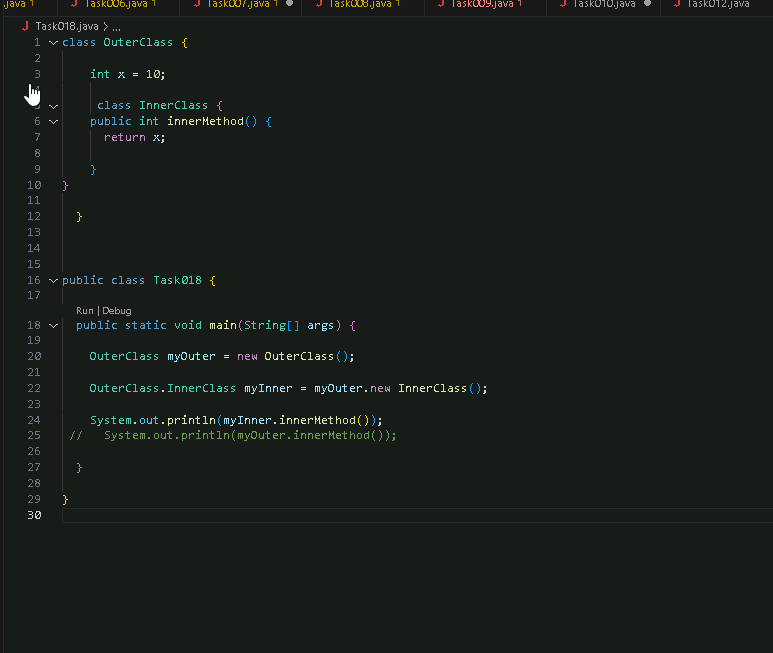
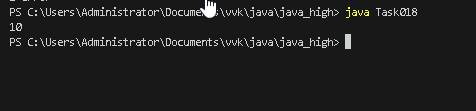


**Task 017:** Use the above code Task 015 and make the inner class static … see the output and explain..



**Using *static* in Inner class doesn’t require an inner class object to be instantiated using object from outer class.**

**Task018:** Use the above code Task 015 and create a method in inner class and return the outer class variable



Task 019  — query by vivek

class OuterClass {

 int x = 10;

static class InnerClass {

static int y = 5;

  }

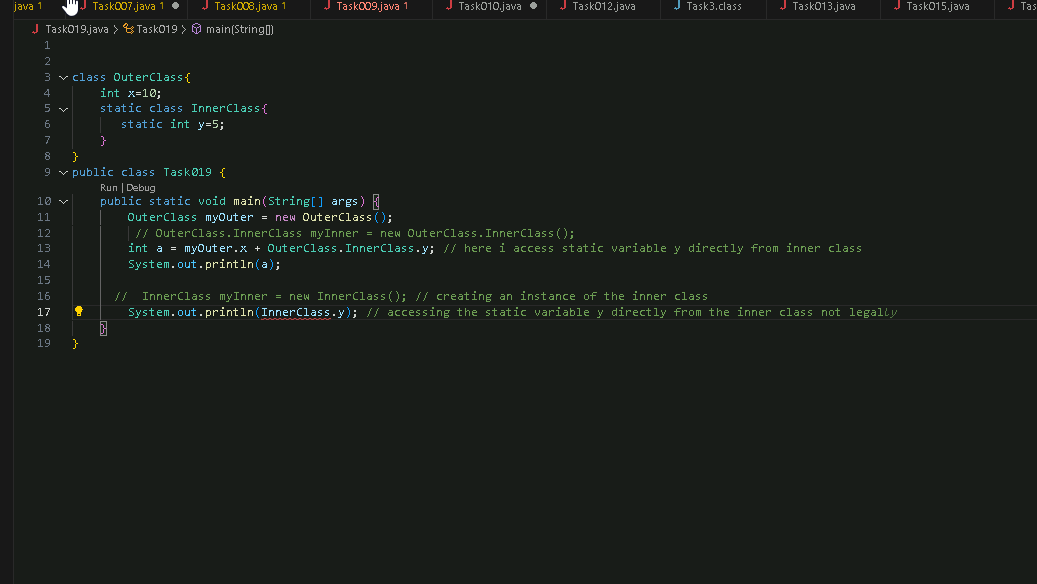
}

public class Main {

public static void main(String[] args) {

OuterClass.InnerClass myInner = new OuterClass.InnerClass();

System.out.println(myInner.y);

  }

}

