**Aim:** Implement program on Multithreading

**Objective**:

## Theory:

**Multithreading in**[**Java**](https://www.javatpoint.com/java-tutorial) is a process of executing multiple threads simultaneously.

A thread is a lightweight sub-process, the smallest unit of processing. Multiprocessing and multithreading, both are used to achieve multitasking.

However, we use multithreading than multiprocessing because threads use a shared memory area. They don't allocate separate memory area so saves memory, and context-switching between the threads takes less time than process.

Java Multithreading is mostly used in games, animation, etc.

Java provides **Thread class** to achieve thread programming. Thread class provides [constructors](https://www.javatpoint.com/java-constructor) and methods to create and perform operations on a thread. Thread class extends [Object class](https://www.javatpoint.com/object-class) and implements Runnable interface.

There are two ways to create a thread:

1. By extending Thread class
2. By implementing Runnable interface.

### **Thread class:**

Thread class provide constructors and methods to create and perform operations on a thread.Thread class extends Object class and implements Runnable interface.

### **1) Java Thread Example by extending Thread class**

**FileName:** Multi.java

**class** Multi **extends** Thread{

**public** **void** run(){

System.out.println("thread is running...");

}

**public** **static** **void** main(String args[]){

Multi t1=**new** Multi();

t1.start();

 }

}

**Output:**

thread is running...

### **2) Java Thread Example by implementing Runnable interface**

**FileName:** Multi3.java

**class** Multi3 **implements** Runnable{

**public** **void** run(){

System.out.println("thread is running...");

}

**public** **static** **void** main(String args[]){

Multi3 m1=**new** Multi3();

Thread t1 =**new** Thread(m1);   // Using the constructor Thread(Runnable r)

t1.start();

 }

}

**Output:**

thread is running...

**Code:**

class Multi2 implements Runnable{

public void run()

{

int a=5;

int b=7;

int c=a+b;

System.out.println("Addition :"+c);

}

public static void main(String args[]){

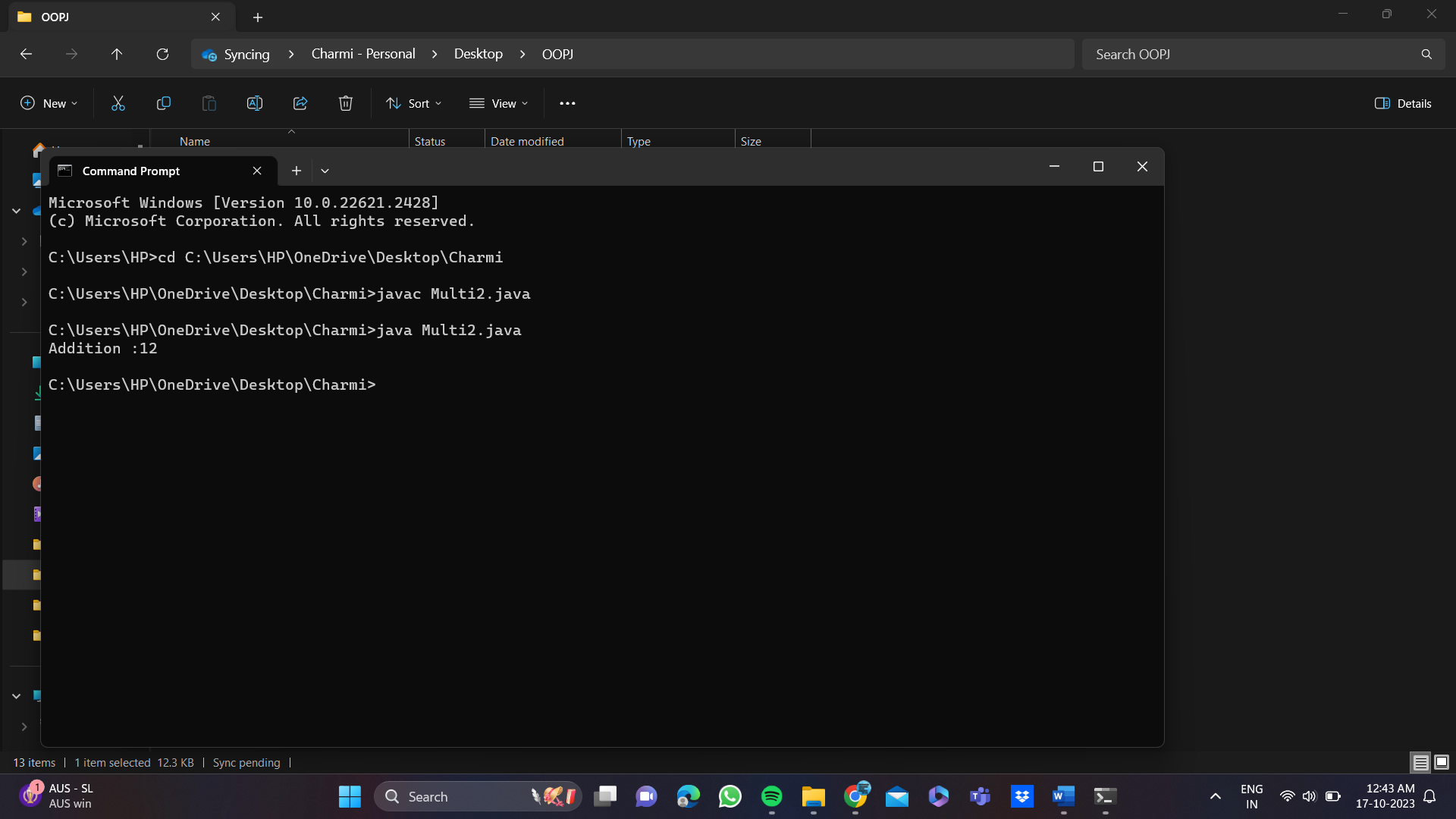
Multi2 m1=new Multi2();

Thread t1=new Thread(m1);

t1.start();

}

}



## Conclusion:

Comment on how multithreading is supported in JAVA.

Multithreading in Java is supported through the Thread class and the Runnable interface. You can create and manage threads by extending the Thread class or implementing the Runnable interface. Java provides thread synchronization, management, and various thread states to enable concurrent execution of tasks. It's a fundamental feature for efficient resource utilization, improved application responsiveness, and better performance in multi-tasking environments.