STIWK3014 REAL TIME PROGRAMMING
Tutorial / Exercise 6: Multithreading

Tasks 1:

You have to develop a program that creates and runs 3 threads. Each thread calculates and prints the multiplication table of a number between one (1) and three (3). The example of the output is shown below:

```
Thread-0: 1 * 1 = 1
Thread-2: 3 * 1 = 3
Thread-2: 3 * 2 = 6
Thread-2: 3 * 3 = 9
Thread-1: 2 * 1 = 2
Thread-1: 2 * 2 = 4
Thread-1: 2 * 3 = 6
Thread-0: 1 * 2 = 2
Thread-0: 1 * 3 = 3
```

Tasks 2:
Complete the following Java codes to ensure that the thread can be started and display unlimited strings. Finally, the thread will be terminated if someone press an **ENTER** key. (Hints: Must use volatile keyword)

```java
class MyThread extends Thread {

    public void run() {







    }

    public void shutdown() {



    }
}

public class MyVolatile {

    public static void main(String[] args) {







    }
}
```

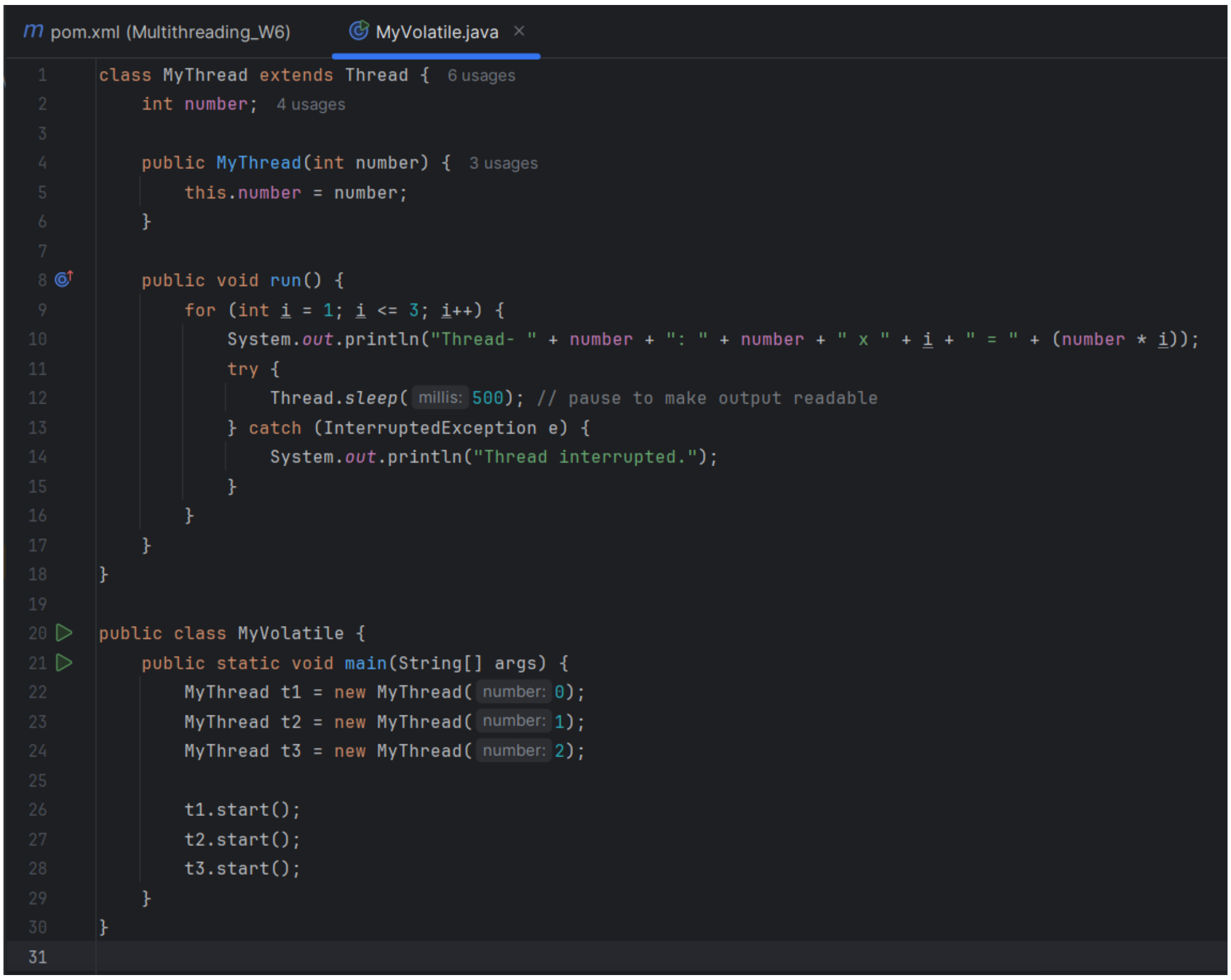
Plagiarism
No mark will be given for plagiarism activities.

Submission:

Platform: 1. Online Learning - Sample Coding & Output in PdF form
2. GitHub – Upload the file

Date: 27 April 2025 (Sunday, before 12.30 noon)

NUR SYAZALINA BINTI BADRUL HISHAM
297527



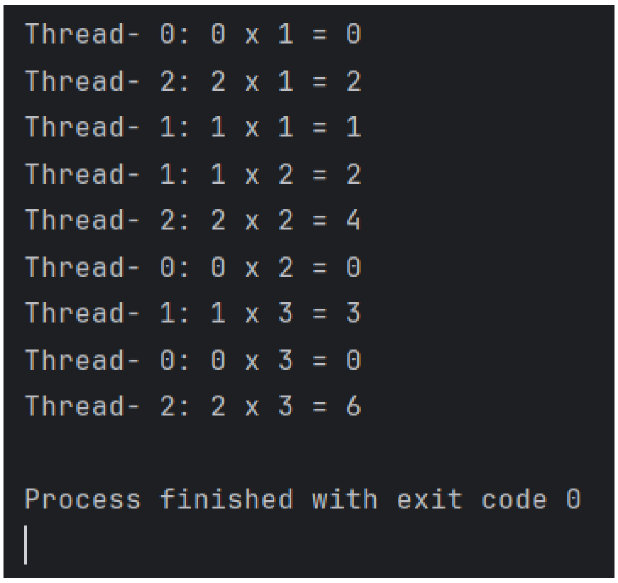
```java
class MyThread extends Thread {  6 usages
    int number;  4 usages

    public MyThread(int number) {  3 usages
        this.number = number;
    }

    public void run() {
        for (int i = 1; i <= 3; i++) {
            System.out.println("Thread- " + number + ": " + number + " x " + i + " = " + (number * i));
            try {
                Thread.sleep( millis: 500); // pause to make output readable
            } catch (InterruptedException e) {
                System.out.println("Thread interrupted.");
            }
        }
    }
}

public class MyVolatile {
    public static void main(String[] args) {
        MyThread t1 = new MyThread( number: 0);
        MyThread t2 = new MyThread( number: 1);
        MyThread t3 = new MyThread( number: 2);

        t1.start();
        t2.start();
        t3.start();
    }
}
```



```
Thread- 0: 0 x 1 = 0
Thread- 2: 2 x 1 = 2
Thread- 1: 1 x 1 = 1
Thread- 1: 1 x 2 = 2
Thread- 2: 2 x 2 = 4
Thread- 0: 0 x 2 = 0
Thread- 1: 1 x 3 = 3
Thread- 0: 0 x 3 = 0
Thread- 2: 2 x 3 = 6

Process finished with exit code 0
```