

STIWK3014 REAL TIME PROGRAMMING
Tutorial / Exercise 8: TestAtomicInteger1p.java & Synchronization.java

Tasks 2:

- Write a program to compare execution times between normal thread and synchronized thread. The results should be displayed in 'second'.
- Sample output:

Normal thread = 0.00000012 seconds

Synchronized thread = 0.00000025 seconds

- For normal thread (unsynchronized), kindly create a class: `NormalThread extends Thread {}`
- To enable the synchronized thread, kindly create a class: `SynchronizedThread extends Thread {}`
- The number of test thread is 10 threads.
- Copy and paste the sample code and the output for the submission. (Rename as: Comparison of Normal Thread and Synchronized Thread and the Outputs)

Code

```
package Synchronization_Exercise8;

public class Comparison {

    public static void main(String[] args) throws InterruptedException {

        int threadCount = 10;

        //Normal
        Thread[] normalThreads = new NormalThread[threadCount];

        long normalStartTime = System.nanoTime();
        long normalEndTime = System.nanoTime();
        double normalTime = (normalEndTime - normalStartTime) / 1_000_000_000.0;

        for (int i = 0; i < threadCount; i++) {
            normalThreads[i] = new NormalThread();
            normalThreads[i].start();
        }

        for (Thread t: normalThreads) {
            t.join();
        }

        //Synchronized
        Thread[] syncThreads = new SynchronizedThread[threadCount];

        long syncStartTime = System.nanoTime();
        long syncEndTime = System.nanoTime();
        double syncTime = (syncEndTime - syncStartTime) / 1_000_000_000.0;

        for (int i = 0; i < threadCount; i++) {
            syncThreads[i] = new SynchronizedThread();
            syncThreads[i].start();
        }

        for (Thread t: syncThreads) {
            t.join();
        }

        System.out.println("Normal thread = " + normalTime + " seconds");
        System.out.println("Synchronized thread = " + syncTime + " seconds");
    }
}
```

```
}  
}
```

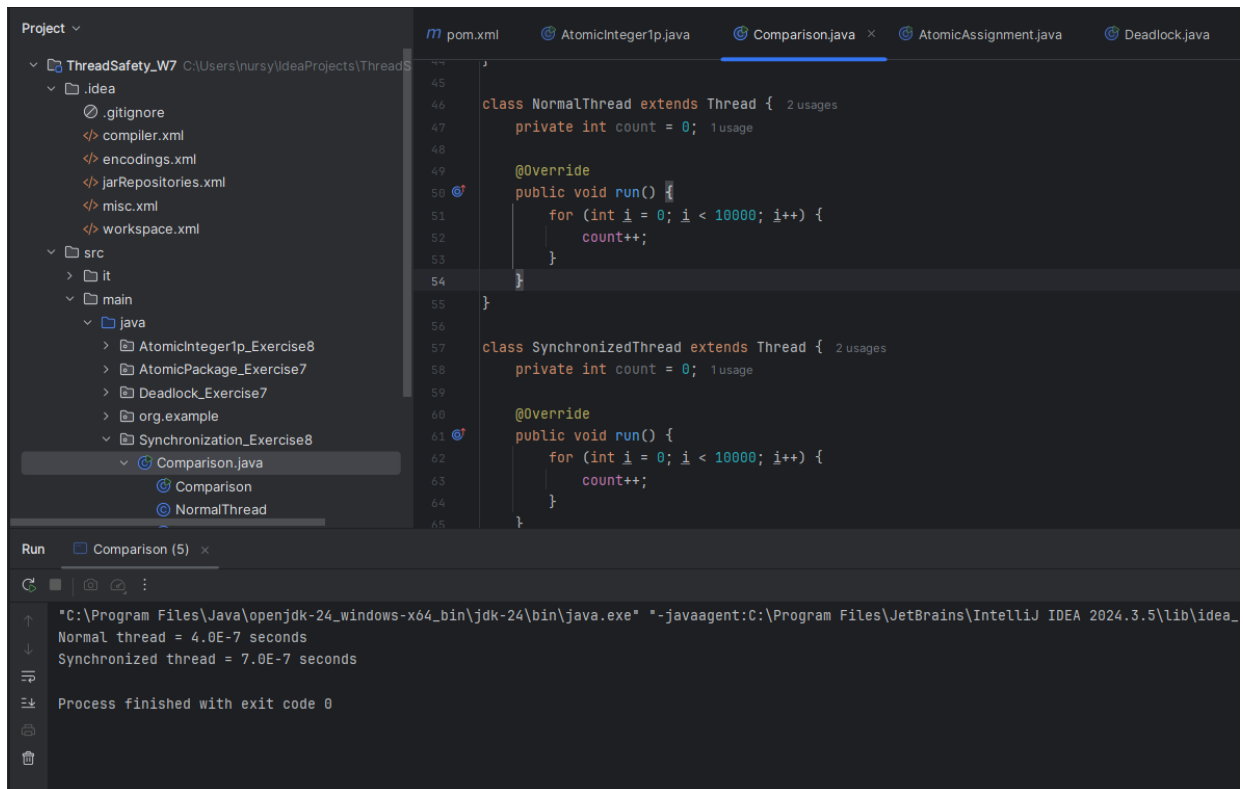
```
class NormalThread extends Thread {  
    private int count = 0;
```

```
    @Override  
    public void run() {  
        for (int i = 0; i < 10000; i++) {  
            count++;  
        }  
    }  
}
```

```
class SynchronizedThread extends Thread {  
    private int count = 0;
```

```
    @Override  
    public void run() {  
        for (int i = 0; i < 10000; i++) {  
            count++;  
        }  
    }  
}
```

Output




The screenshot shows the IntelliJ IDEA interface. The left sidebar displays the project structure for 'ThreadSafety_W7', with 'Comparison.java' selected under 'src/main/java'. The main editor shows the code for 'NormalThread' and 'SynchronizedThread'. The 'Run' console at the bottom shows the execution output.

```
44 }
45
46 class NormalThread extends Thread { 2 usages
47     private int count = 0; 1 usage
48
49     @Override
50     public void run() {
51         for (int i = 0; i < 10000; i++) {
52             count++;
53         }
54     }
55 }
56
57 class SynchronizedThread extends Thread { 2 usages
58     private int count = 0; 1 usage
59
60     @Override
61     public void run() {
62         for (int i = 0; i < 10000; i++) {
63             count++;
64         }
65     }
66 }
```

Run Comparison (5) x

```
"C:\Program Files\Java\openjdk-24_windows-x64_bin\jdk-24\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2024.3.5\lib\idea_
Normal thread = 4.0E-7 seconds
Synchronized thread = 7.0E-7 seconds
Process finished with exit code 0
```



This is a close-up of the Run console from the previous screenshot, showing the execution output for 'Comparison (5)'.

```
Run Comparison (5) x

"C:\Program Files\Java\openjdk-24_windows-x64_bin\jdk-24\bin\java.exe" "-javaagent:C:\Pro
Normal thread = 4.0E-7 seconds
Synchronized thread = 7.0E-7 seconds
Process finished with exit code 0
```

Plagiarism

No mark will be given for plagiarism activities especially those copied from ChatGPT.

Submission:

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

Date: 11 May 2025 (Sunday, before 12.30 noon)