Activity 6

10-15-2023

Kohl Johnson

# Part 1A Files

**Step 6 Screenshot**

**A screenshot of a computer

Description automatically generated**

**Step 7 Response**

"MyThread1 is running" was displayed first simply because it was created first within the main method, followed by MyThread2. Both are running at the same time, as stated in the activity guide, but the output is only called once and will only be displayed once.

# Part 1B Files

**Step 5 Screenshot**

**A screenshot of a computer

Description automatically generated**

**Step 6 Response**

On my end, all the iterations were shown at the same time. One the first line, "MyThread1.." was shown, and then "MyThread2..." was shown. This continued until Thread2 (with an iteration count of 100) was complete, then only Thread1 was printing to console.

**Step 9 Screenshot**

**A screenshot of a computer

Description automatically generated**

**Step 10 Response**

For every 2 iterations of Thread2, Thread1 did one iteration. I highlighted the output showing this in the screenshot, but Thread1 was delayed by 1 second while Thread2 only had a delay of 1/2 of a second, hence it did double the iterations Thread1 could do in one second.

# Part 1C Files

**Step 6 Screenshot**

**A screenshot of a computer

Description automatically generated**

**Step 7 Response**

I think the counter did not reach the value of 1000 because by the time the main method was finished and the program terminated, there were still CounterThreads sleeping. With my end counter result being 974, I think that means 26 Threads were still sleeping at the time the program was terminated.

**Step 11 Screenshot**

**A screenshot of a computer screen

Description automatically generated**

**Step 12 Response**

I think with using "synchronized" it waits until all Threads finish what they are doing. With the max sleep amount being 1000 ms (1 second), the max time it will take is one second, so there really is not much waiting on our end.

# Part 2 Files

**Step 6 Screenshot**

**A screenshot of a computer screen

Description automatically generated**

**Step 7 Screenshot**

**A screenshot of a computer screen

Description automatically generated**

**Step 8 Response**

Both the Client and the Server can send/receive messages from one another. First the server is created, then the client, because we need a server running before a client can connect. Basically, a network application works by reading/writing messages and doing other tasks based on those messages.

# Part 3 Files

**Step 7 Screenshot 1**

**A screenshot of a computer

Description automatically generated**

**Step 7 Screenshot 2**

**A screenshot of a computer

Description automatically generated**

**Step 8 Response**

This ties back to the main reason for using threads, speed. We can use multiple threads to perform different tasks at the same time. The ServerThread is simply printing a period every five seconds, while the Server is actually reading/writing data to and from the Client.