Title: CS490 Project 1

Course: CS 490 – Operating system

Instructor: Mr. Stephen Quattlebaum

Due Date: 03/03/2024

Iswarya Gadde

CS490 Project 1

Project Name: Iswarya_490_project_1

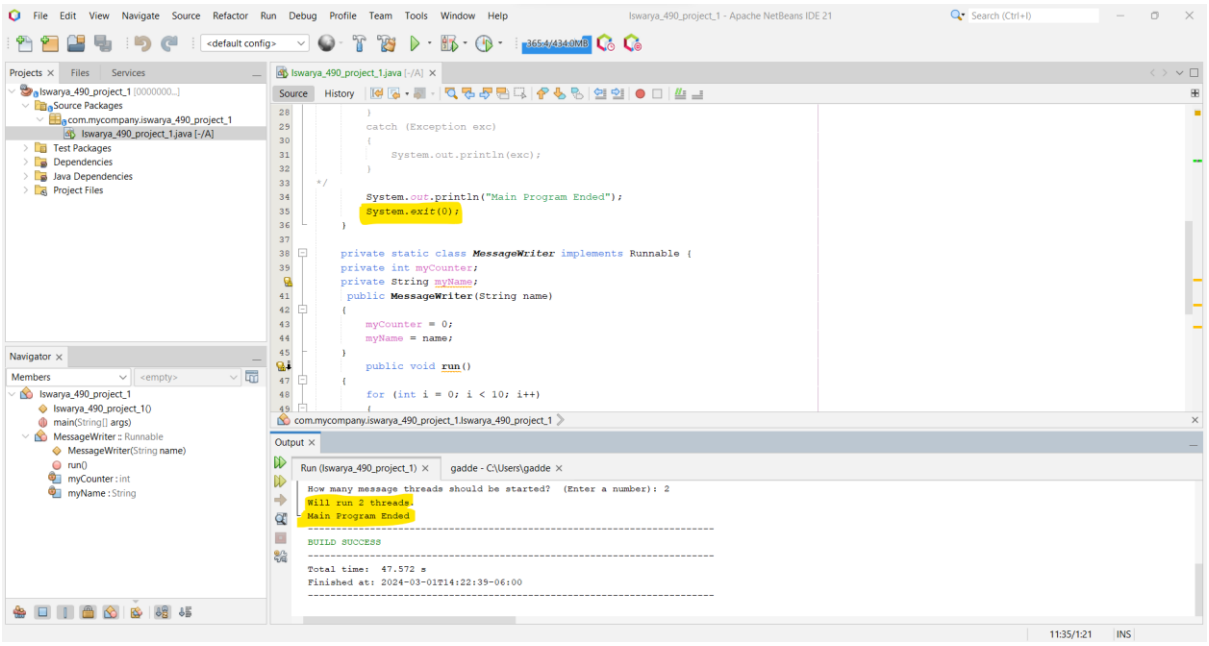
Purpose: Java threads observation.

Environment Setup:

1. According to the instructions install the NetBeans environment to create a Java project.
2. Prerequisites for NetBeans: Java SDK
3. Created project and added code provided by the professor.

Exercises:

1. What happens if you uncomment System.exit(0) at the end of the main program?

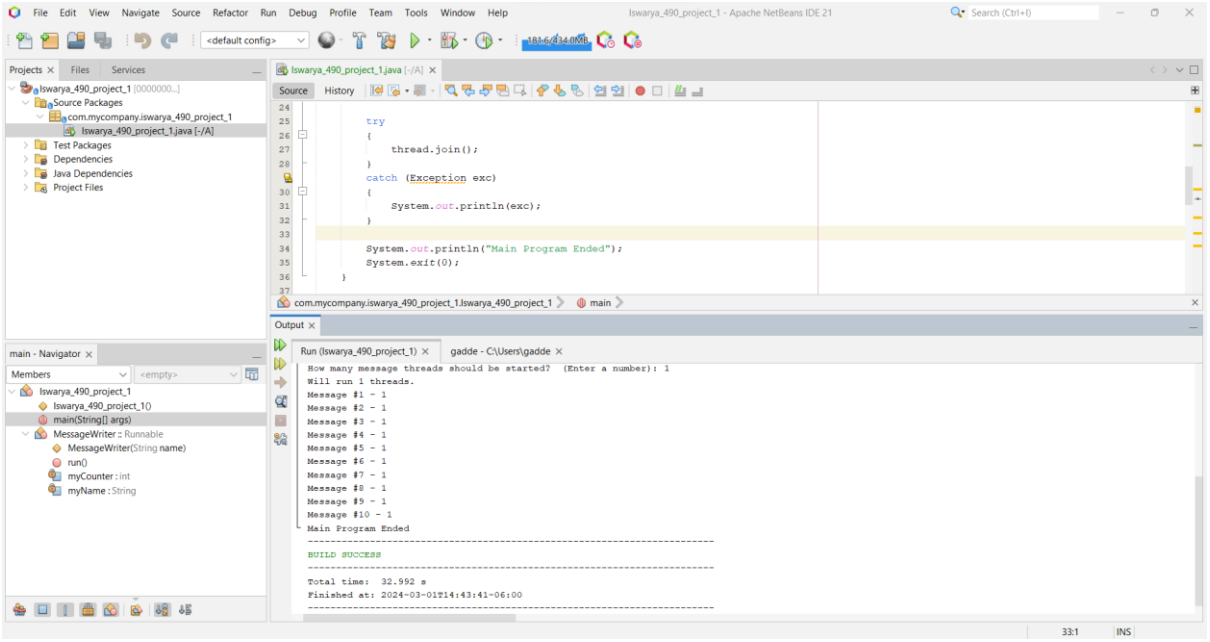


When we uncomment the System.exit(0), it will forcefully terminate the Java Virtual Machine, so in this case it will abruptly stop the entire program, including any running threads.

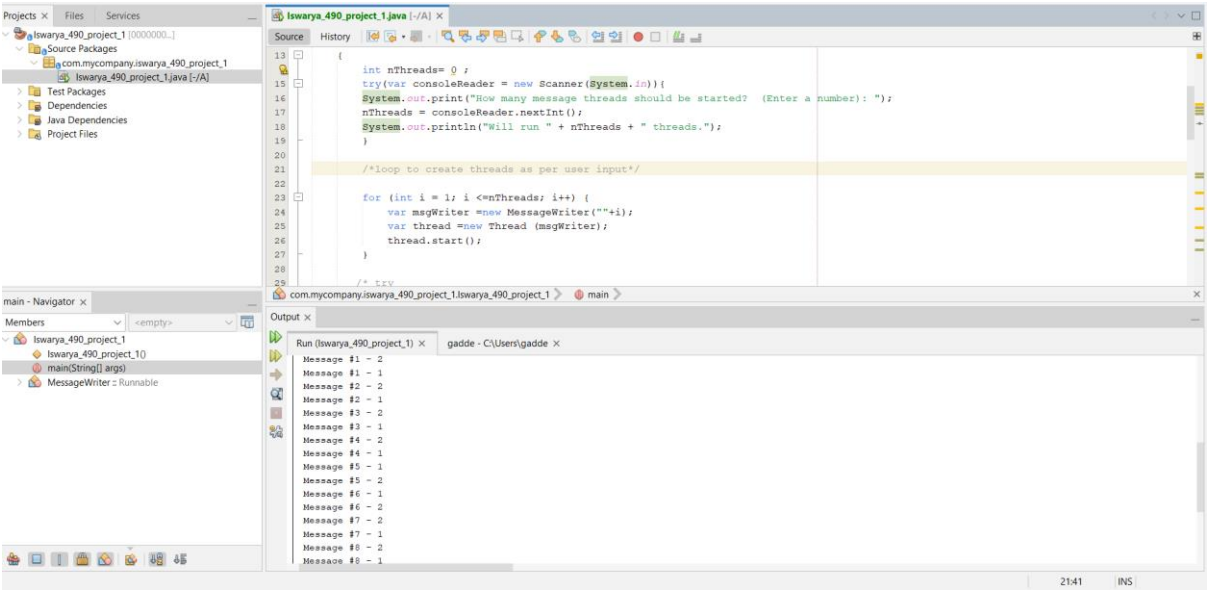
2. Why does uncommenting the try/catch around the call to thread.join resolve that problem?

   - It prompts the user to input the number of message threads to start.
   - For each thread to start, it creates an instance of the MessageWriter class, assigns it a unique name based on the thread number, and starts a new thread running the run method of the MessageWriter class.
   - The run method of the MessageWriter class prints 10 messages with a one-second delay between each message.

Here the thread.join pauses its execution and waits for the specified thread to finish its execution, so the main program will wait for the thread to finish its work before proceeding or allowing the program to exit.



3. Update the code to create as many message threads as are entered at the command line.

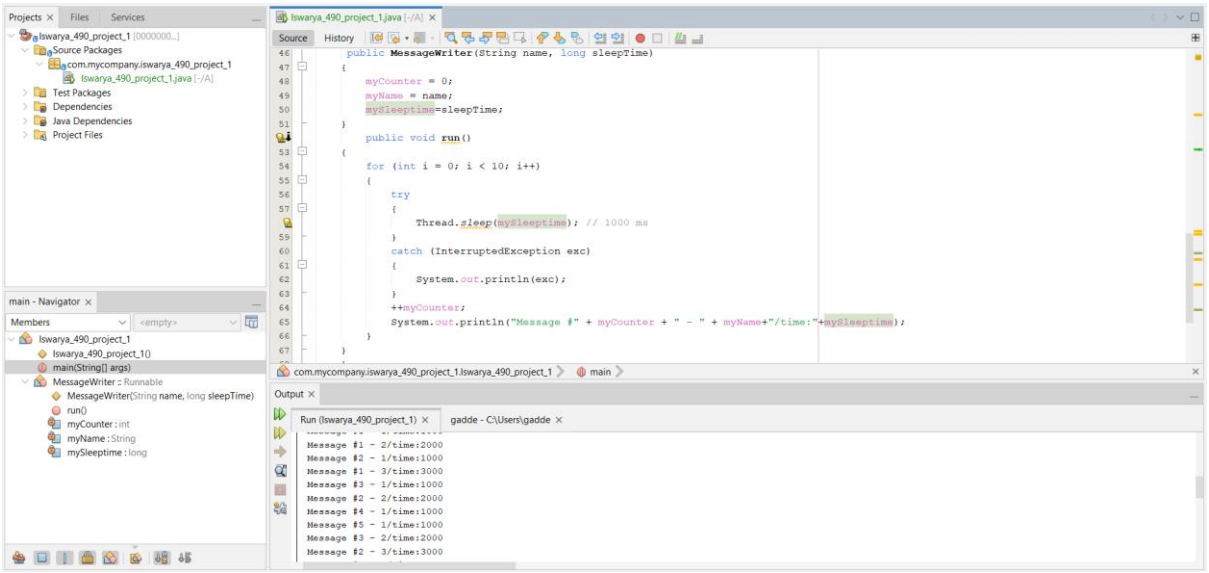


I implemented for loop to create threads according to the user input entered in the command line.

4. Update the code so that MessageWriter takes a parameter indicating how long to delay instead of always delaying 1 second. Pass 1 second to the first thread, 2 seconds to the second thread, and so on.

Modified for loop code and messageWriter class as per image

```
for (int i = 1; i <=nThreads; i++) {
        var msgWriter =new MessageWriter(""+i,1000*i);
        var thread =new Thread (msgWriter);
        thread.start();
    }
```



After modifications are added to the thread class and message writer class we can see time after every thread.

5. Update the code so that the MessageWriter objects share a single counter, instead of each having their own counter. This involves sharing the counter between Java threads Added Private static int in global level .