CSCD 467/567 Lab1
Thread Interruption Using KeyListener in Java

No Late Submissions are accepted.

Rules: Your code must use Java Language. If your program shows a compilation error, you get a zero for this lab assignment. To avoid compatibility issues, I encourage you to upgrade your JRE to the latest version of SE 1.7 or 1.8.

Submission: Wrap up all your java files and a ReadMe text file into a single zip file. Name your zip file as *firstInitialYourLastName*CSCD467Lab1.zip. For example, if your legal name is Will Smith, you should name your zip file as wSmithCSCD467Lab1.zip.

You are required to submit the ReadMe text file along with all your java code. In the ReadMe file you should put your legal full name, description about how to compile and how to run your program. An example of ReadMe file should look like the following:

```
(This only serve as an Example. Your ReadMe file should contain the similar content.)
Name: Will Smith
Description: unzip the submitted wSmithCSCD467Lab1.zip, you get a folder named
smithLab1.
To Compile:  cd into folder smithLab1,
        javac *.java
To Run
        java myLab1   4    8
```

Before you leave the laboratory, please show the TA or the instructor how your program works, they will give you a score for this Lab assignment.

For archive purpose, please also submit your single zip file on EWU Canvas by following CSCD467-01 Course →Assignments→Lab1→ Submit Assignment to upload your single zip file.

Problem Description:

On the basis of the demonstration code I showed in last class, regarding JFrame, JTextArea and KeyListener, we will write an application that creates **two** additional Java Threads in the main thread.

1, Once these two threads start, they **continuously** display a message every second "Message from thread → *ThreadName*" in the JTextArea. A screen shot is shown in figure 1 after the program starts.

2, Then if we press a key on the keyboard, Thread-1 is interrupted and terminated. A message of "Thread-1 Gets Interrupted! Terminate!" is displayed. Figure 2 shows the screen shot for the program after we made the first key stroke. We can see that Thread-2 continues to send output to the JTextArea, after Thread-1 was terminated.

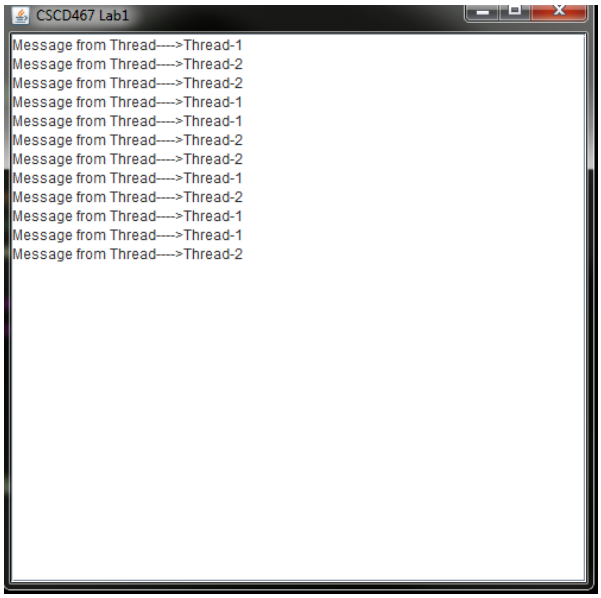

Figure 1, program just started.

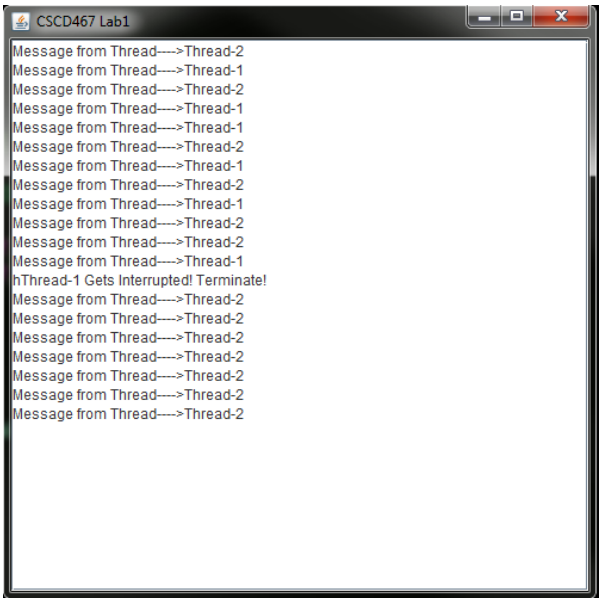

Figure 2, program behavior after we press first key.

3, If we press a seond key on the keyboard, the Thread-2 terminates and stops outputting messages, meanwhile a message “Thread-2 Gets Interrupted! Terminate!” is displayed. At this point, both Thread-1 and Thread-2 are terminated. If we continute to press the third key, a message “All threads are interrupted” is shown in the JTextArea. An screen shot is shown in figure 3.

4, After the third key stroke, the program will NOT response to any key stroke , except for displaying what you typed on the keyboard.

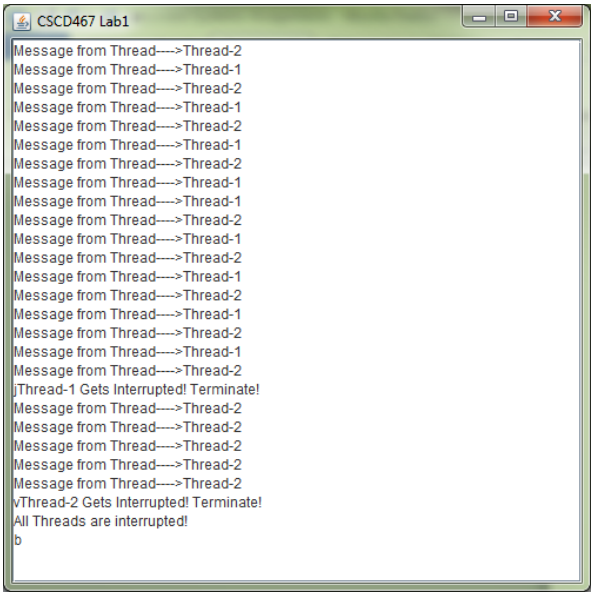

Figure 3, program behavior after we press second and third key.