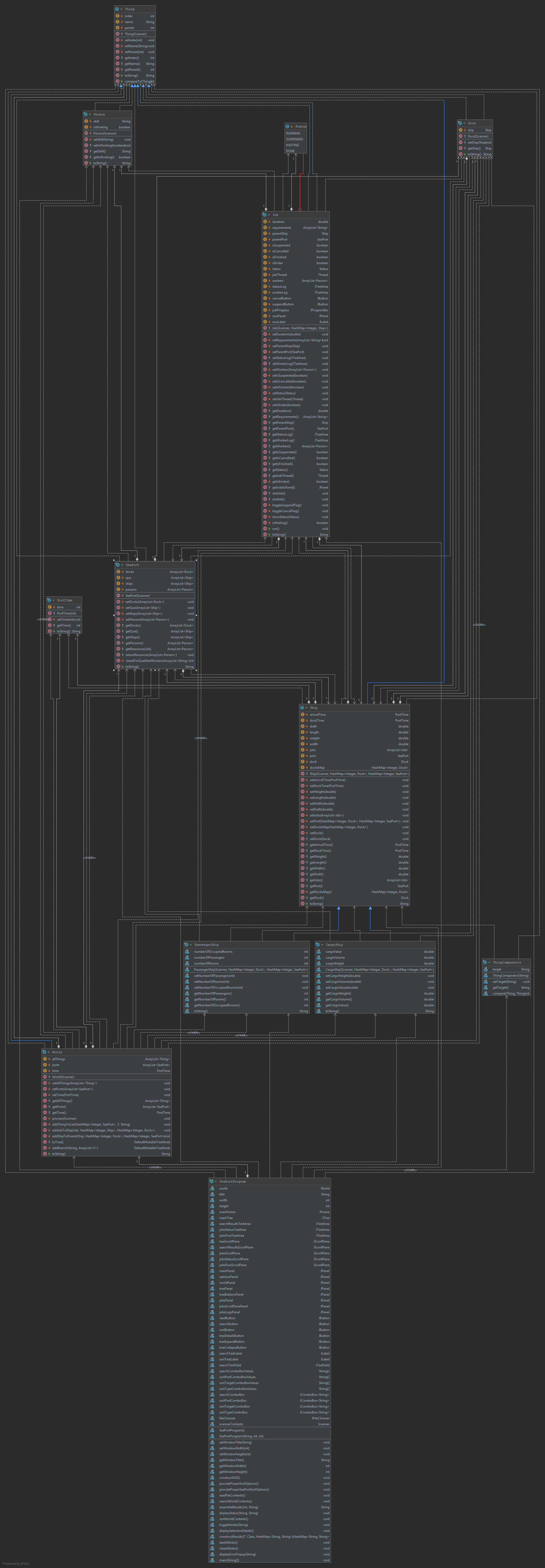
CMSC 335 Project 3

Stanislav Fatkhutdinov

University of Maryland University College

Part 1: Due to lack screen resolution the class diagram might be appearing small. I attached the PNG file into the project folder to view in higher resolution.

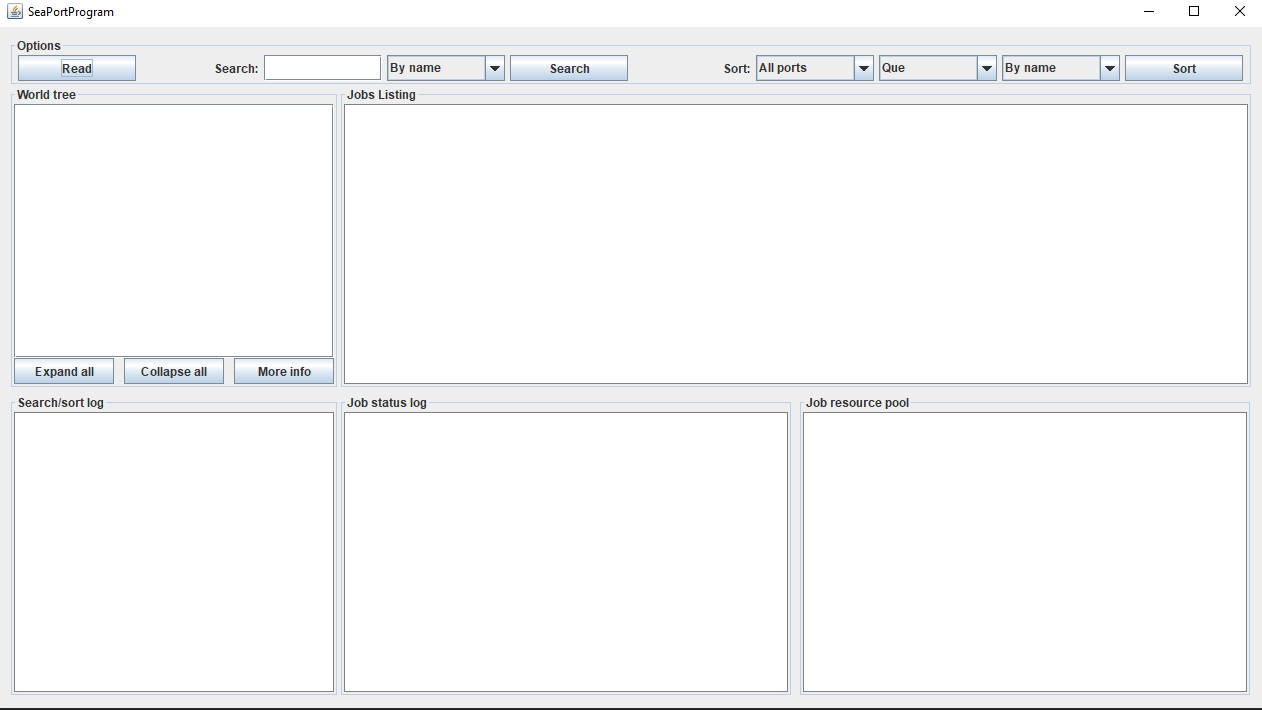


Part 2:

User’s Guide:

In order to run this program, the user needs to either have a java IDE (Netscape, IntelliJ, etc) or to run the SeaPortProgram.java from the command line. To do that the user needs to open up his or her command line prompt and type in “javac SeaPortProgram.java” and press ENTER. The program will compile and now can be run through typing “java SeaPortProgram” and then pressing ENTER. I reworked my GUI a little bit to make it clearer. There are additional panes reflecting the job status log, search log, and job resource pool. The rest of the GUI doesn’t differ from the project 2.

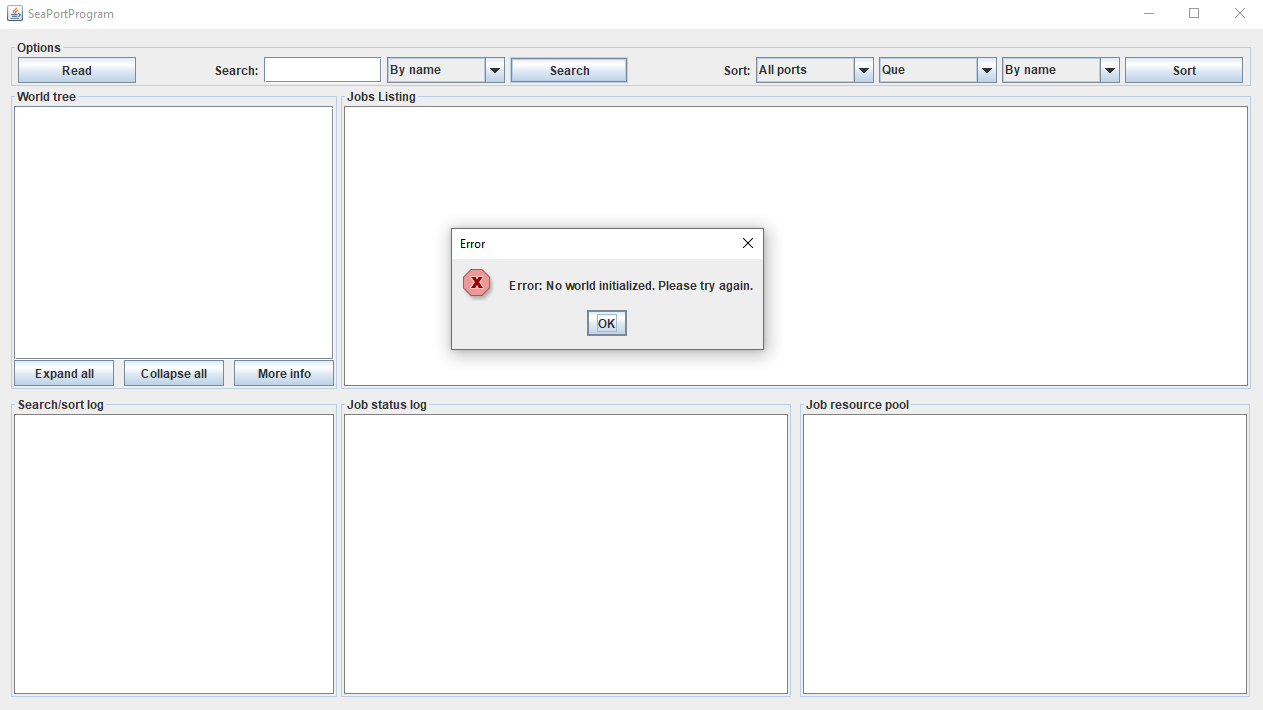
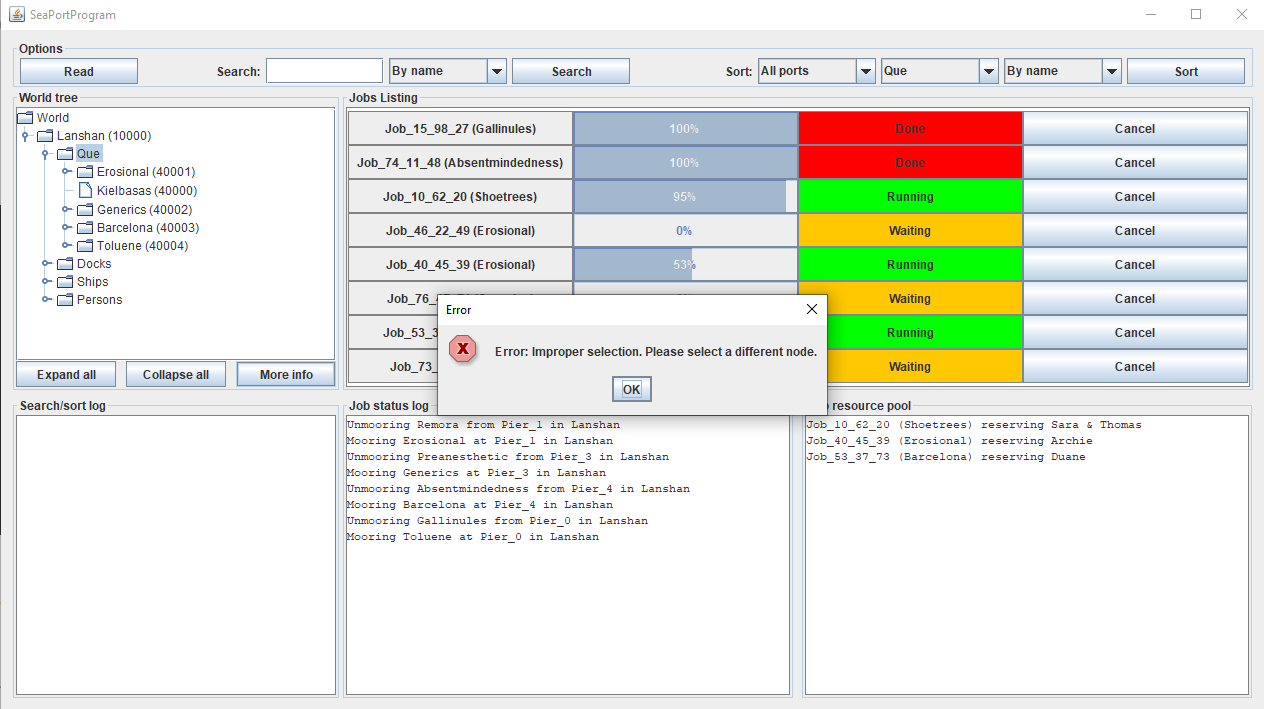
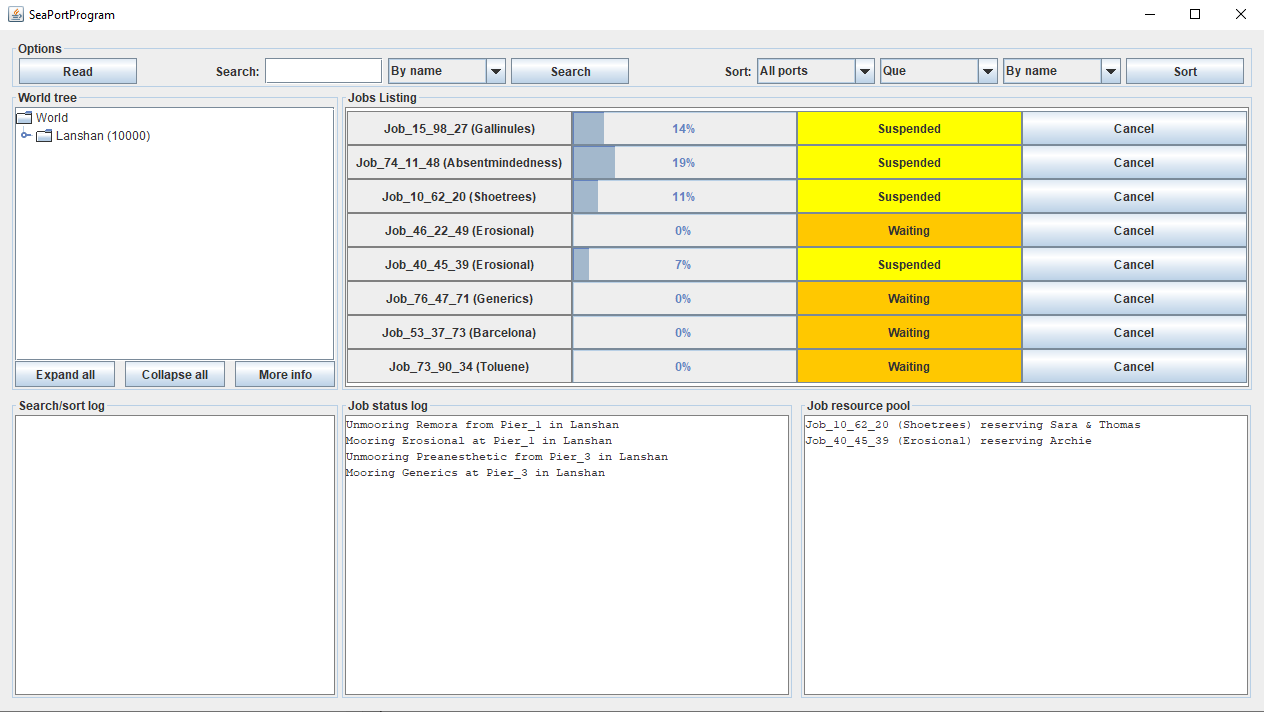
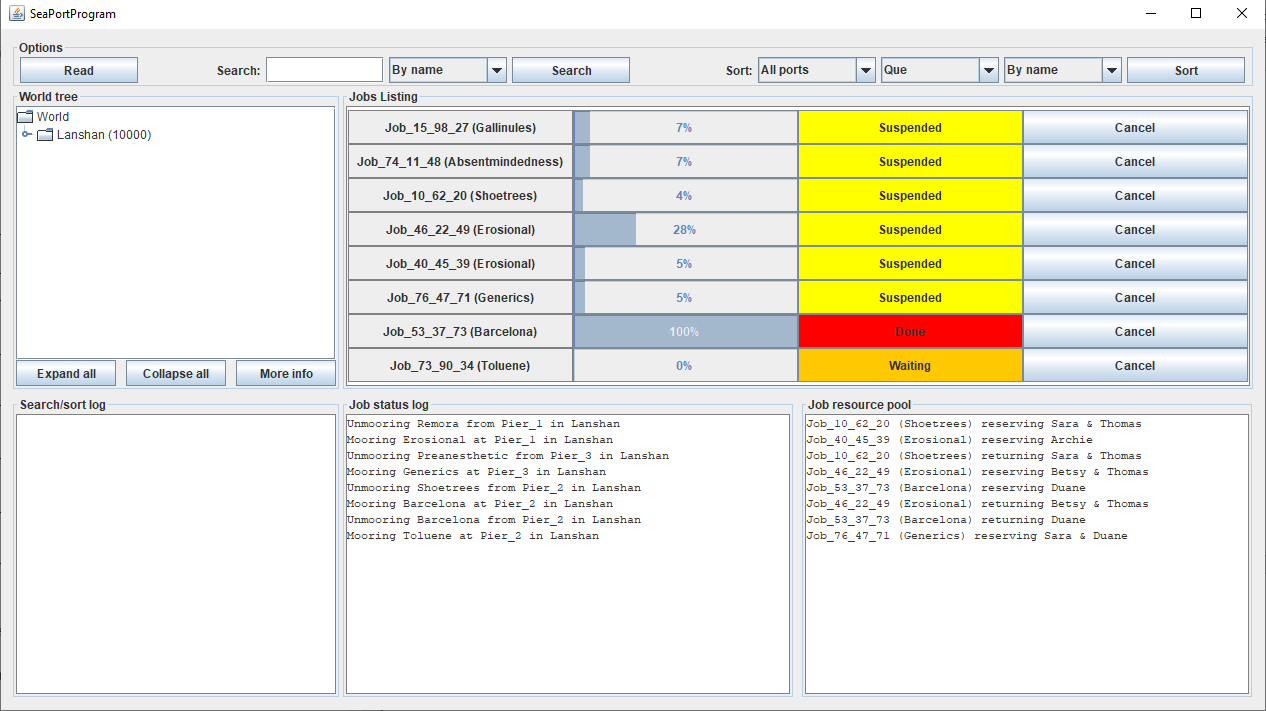
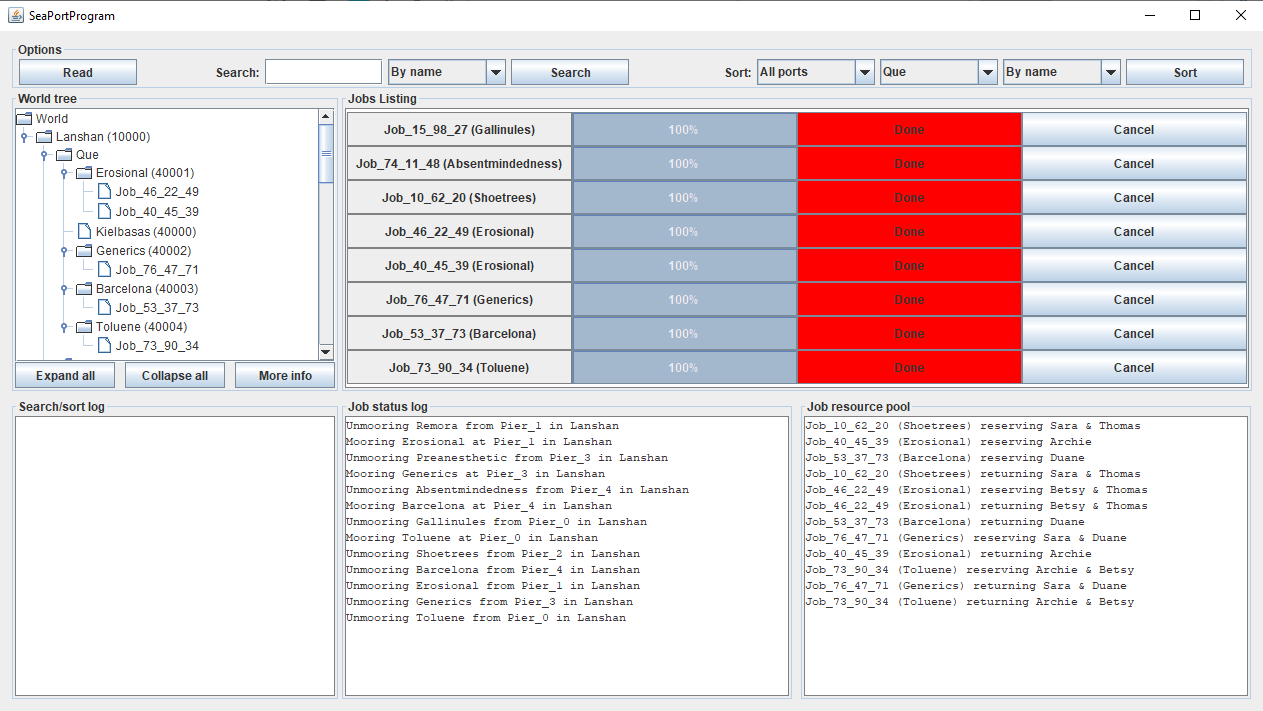
The GUI will pop up, this means that the program is working.



Part 3:

Test Cases:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Input | Expected Output | Actual Output | Passed? |
| 1 | Sort button | Error. No world initialized. Please try again. |  | y |
| 2 | aSpaa.txt  Que node and press More Info button | Error: Improper Selection. Please select a different node. |  | y |
| 3 | aSpaa.txt  Suspend button on all running jobs | Progressing bar paused, the cell colors changed |  | y |
| 4 | aSpaa.txt  Cancel button for Barcelona | Ship’s progression is halted the text and color will change |  | y |
| 5 | aSpaa.txt  Wait for job completion | The jobs will complete concurrently in a different order depending on the resource pool |  | y |

1. 
2. 
3. 
4. 
5. 

Lesson Leared:

This week I was heavily researching Threads and runnables. I was influenced by the resource website idea “an alternate approach is to create the job threads but not start them until the data file is completely read, using a loop at the end of the read file method”. I think all the threads in the Job class are properly loaded. JTables were a little tricky to implement because of its complexity.