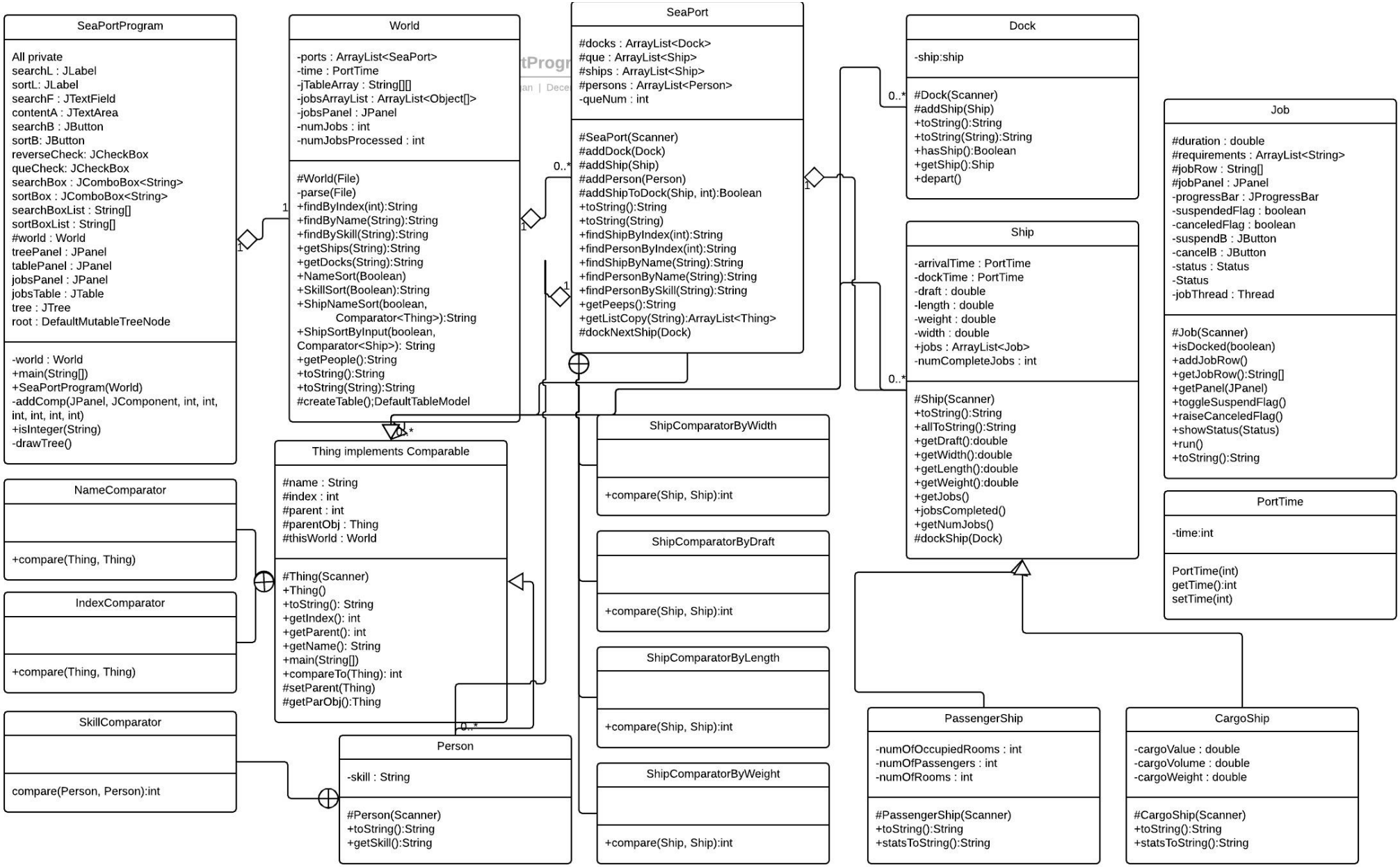
Zachary Finnegan

11/30/2019

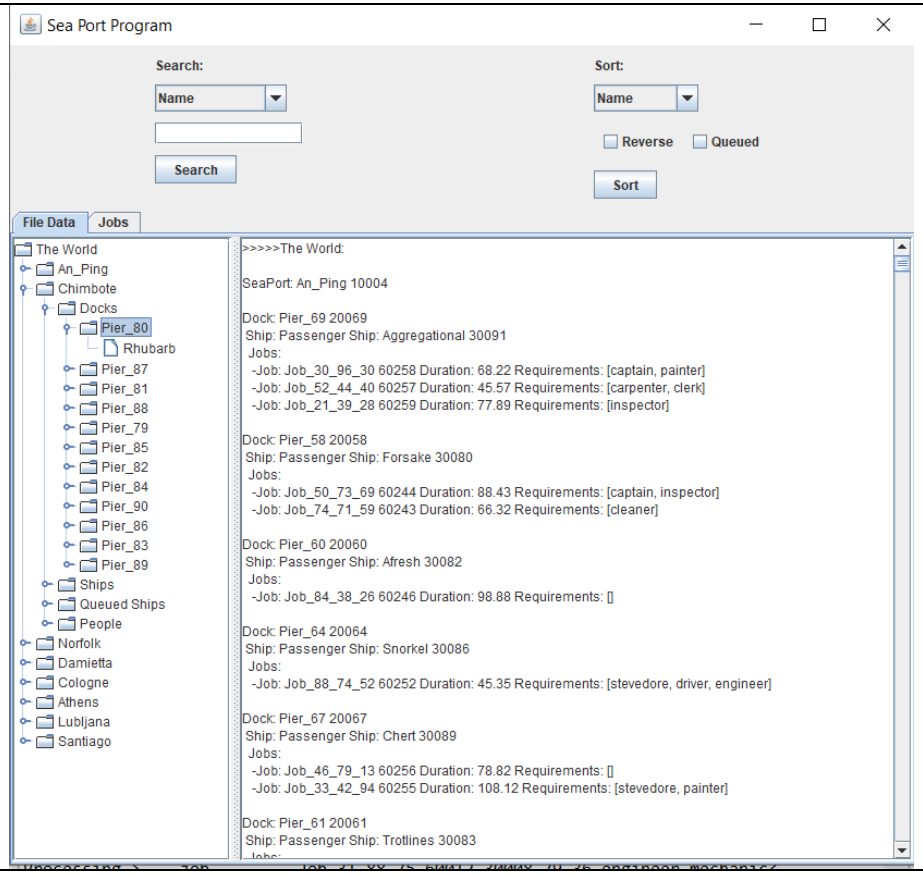
Project 3

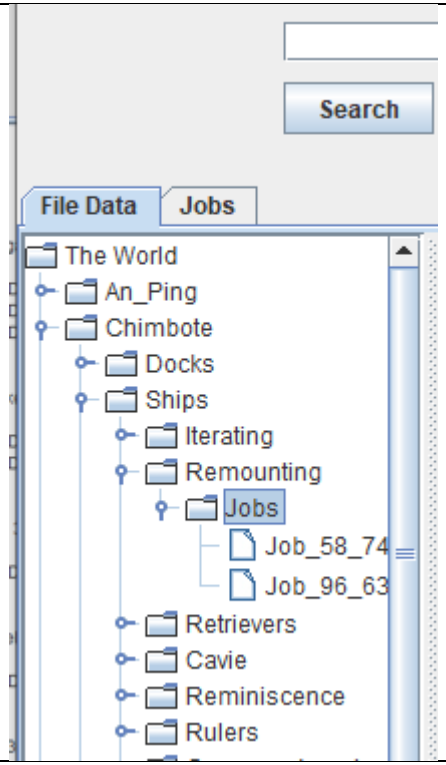
CMSC 335

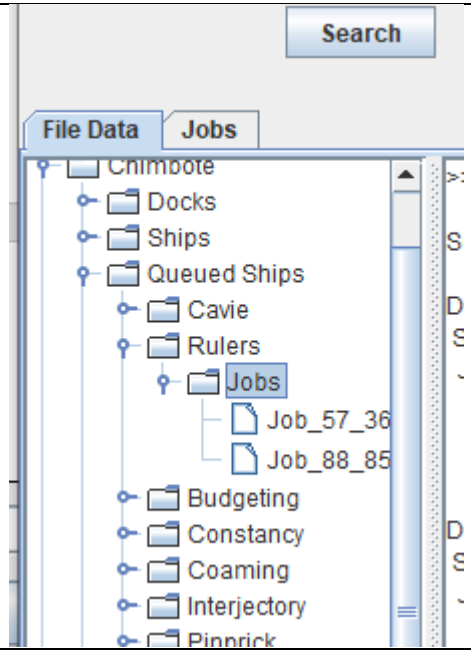
1. Design

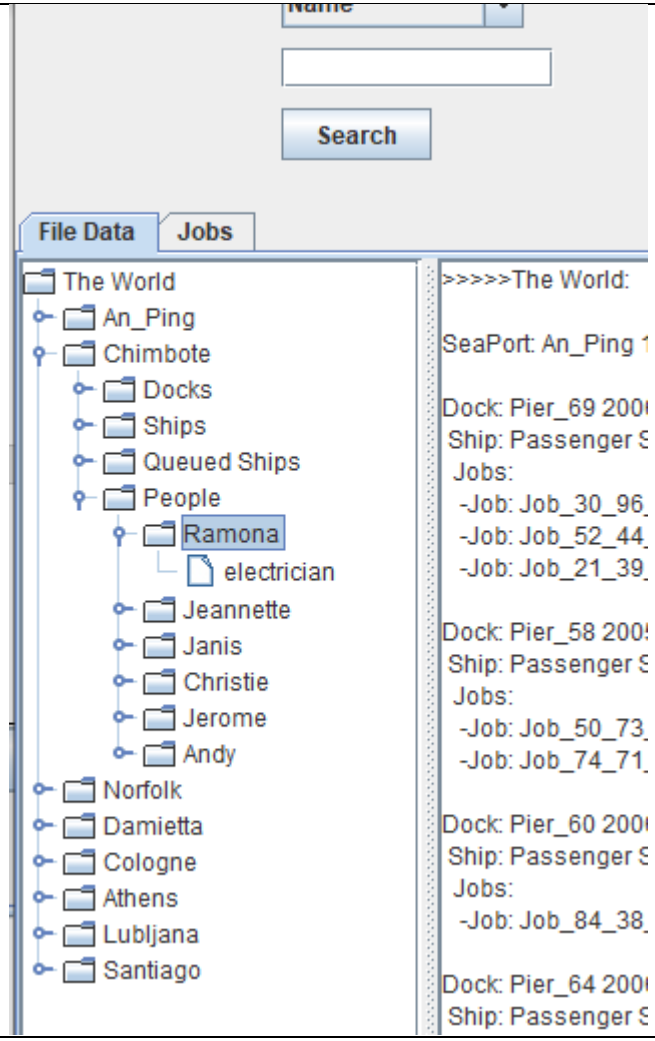

SeaPortProgram
All private
searchL : JLabel
sortL: JLabel
searchF : JTextField
contentA : JTextArea
searchB : JButton
sortB: JButton
reverseCheck: JCheckBox
queCheck: JCheckBox
searchBox : JComboBox<String>
sortBox : JComboBox<String>
searchBoxList : String[]
sortBoxList : String[]
#world : World
treePanel : JPanel
tablePanel : JPanel
jobsPanel : JPanel
jobsTable : JTable
tree : JTree
root : DefaultMutableTreeNode
-world : World
+main(String[])
+SeaPortProgram(World)
-addComp(JPanel, JComponent, int, int, int, int, int, int)
+isInteger(String)
-drawTree()
World
-ports : ArrayList<SeaPort>
-time : PortTime
-jTableArray : String[][]
-jobsArrayList : ArrayList<Object[]>
-jobsPanel : JPanel
-numJobs : int
-numJobsProcessed : int
#World(File)
-parse(File)
+findByIndex(int):String
+findByName(String):String
+findBySkill(String):String
+getShips(String):String
+getDocks(String):String
+NameSort(Boolean)
+SkillSort(Boolean):String
+ShipNameSort(boolean, Comparator<Thing>):String
+ShipSortByInput(boolean, Comparator<Ship>): String
+getPeople():String
+toString():String
+toString(String):String
#createTable();DefaultTableModel
SeaPort
#docks : ArrayList<Dock>
#que : ArrayList<Ship>
#ships : ArrayList<Ship>
#persons : ArrayList<Person>
-queNum : int
#SeaPort(Scanner)
#addDock(Dock)
#addShip(Ship)
#addPerson(Person)
#addShipToDock(Ship, int):Boolean
+toString():String
+toString(String)
+findShipByIndex(int):String
+findPersonByIndex(int):String
+findShipByName(String):String
+findPersonByName(String):String
+findPersonBySkill(String):String
+getPeeps():String
+getListCopy(String):ArrayList<Thing>
#dockNextShip(Dock)
Dock
-ship:ship
#Dock(Scanner)
#addShip(Ship)
+toString():String
+toString(String):String
+hasShip():Boolean
+getShip():Ship
+depart()
Job
#duration : double
#requirements : ArrayList<String>
#jobRow : String[]
#jobPanel : JPanel
-progressBar : JProgressBar
-suspendedFlag : boolean
-canceledFlag : boolean
-suspendB : JButton
-cancelB : JButton
-status : Status
-Status
-jobThread : Thread
#Job(Scanner)
+isDocked(boolean)
+addJobRow()
+getJobRow():String[]
+getPanel(JPanel)
+toggleSuspendFlag()
+raiseCanceledFlag()
+showStatus(Status)
+run()
+toString():String
Ship
-arrivalTime : PortTime
-dockTime : PortTime
-draft : double
-length : double
-weight : double
-width : double
+jobs : ArrayList<Job>
-numCompleteJobs : int
#Ship(Scanner)
+toString():String
+allToString():String
+getDraft():double
+getWidth():double
+getLength():double
+getWeight():double
+getJobs()
+jobsCompleted()
+getNumJobs()
#dockShip(Dock)
PortTime
-time:int
PortTime(int)
getTime():int
setTime(int)
Thing implements Comparable
#name : String
#index : int
#parent : int
#parentObj : Thing
#thisWorld : World
#Thing(Scanner)
+Thing()
+toString(): String
+getIndex(): int
+getParent(): int
+getName(): String
+main(String[])
+compareTo(Thing): int
#setParent(Thing)
#getParObj():Thing
NameComparator
+compare(Thing, Thing)
IndexComparator
+compare(Thing, Thing)
SkillComparator
compare(Person, Person):int
Person
-skill : String
#Person(Scanner)
+toString():String
+getSkill():String
ShipComparatorByWidth
+compare(Ship, Ship):int
ShipComparatorByDraft
+compare(Ship, Ship):int
ShipComparatorByLength
+compare(Ship, Ship):int
ShipComparatorByWeight
+compare(Ship, Ship):int
PassengerShip
-numOfOccupiedRooms : int
-numOfPassengers : int
-numOfRooms : int
#PassengerShip(Scanner)
+toString():String
+statsToString():String
CargoShip
-cargoValue : double
-cargoVolume : double
-cargoWeight : double
#CargoShip(Scanner)
+toString():String
+statsToString():String
1
0..*

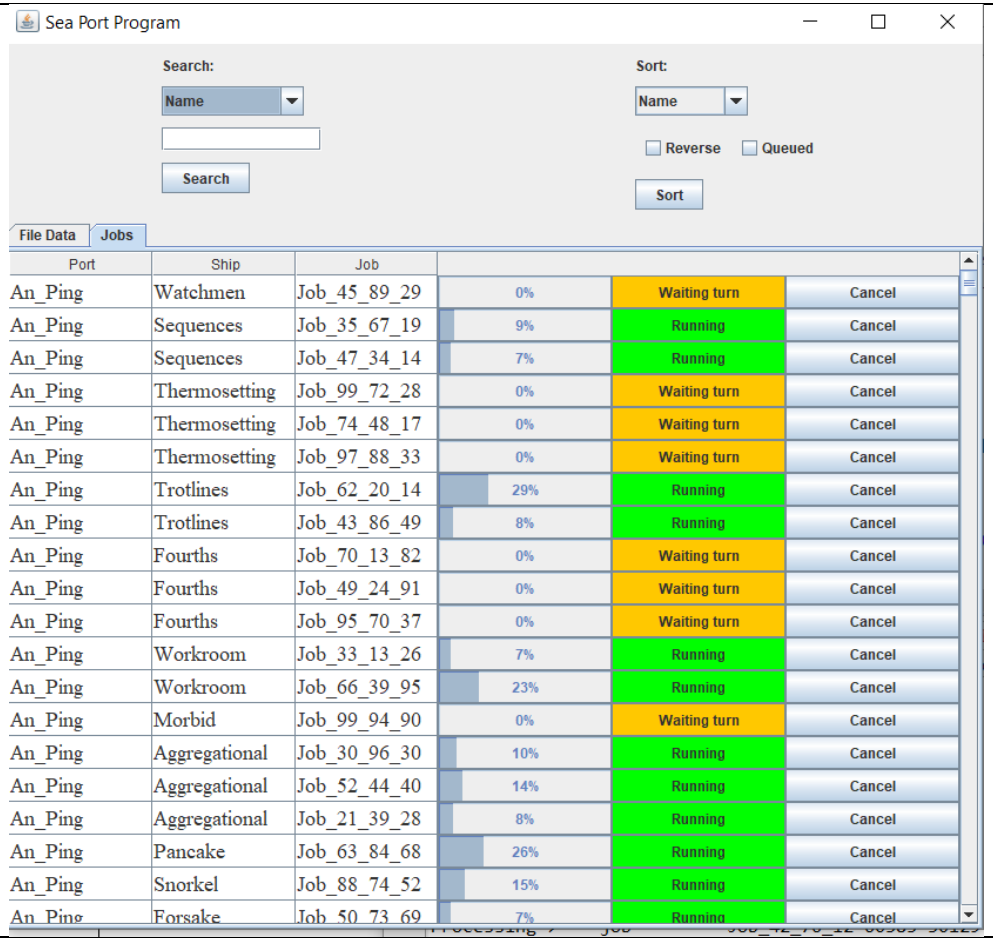
2. User's Guide
    a. Add the package to your IDE path. Import it to your IDE and then press run. Select the SeaPortData.txt when the JFileChooser pops up and then search and sort to your hearts content.
3. Test Plan
    a. This table will only represent new functionality. See previous documentation to see test cases for the original program and version 2 of the program.

Test	Input	Expected Output	Screenshot Output	Pass?
Tree Test	Open aSPac.txt	A nice tree -World --Each Port ---Dock ----Each Dock -----Ship in Dock ---Ship ----Each Ship -----Jobs ------Each Job ---Queued Ships ----Each Ship -----Jobs ------Each Job ---People ----Each People -----Skill		Yes

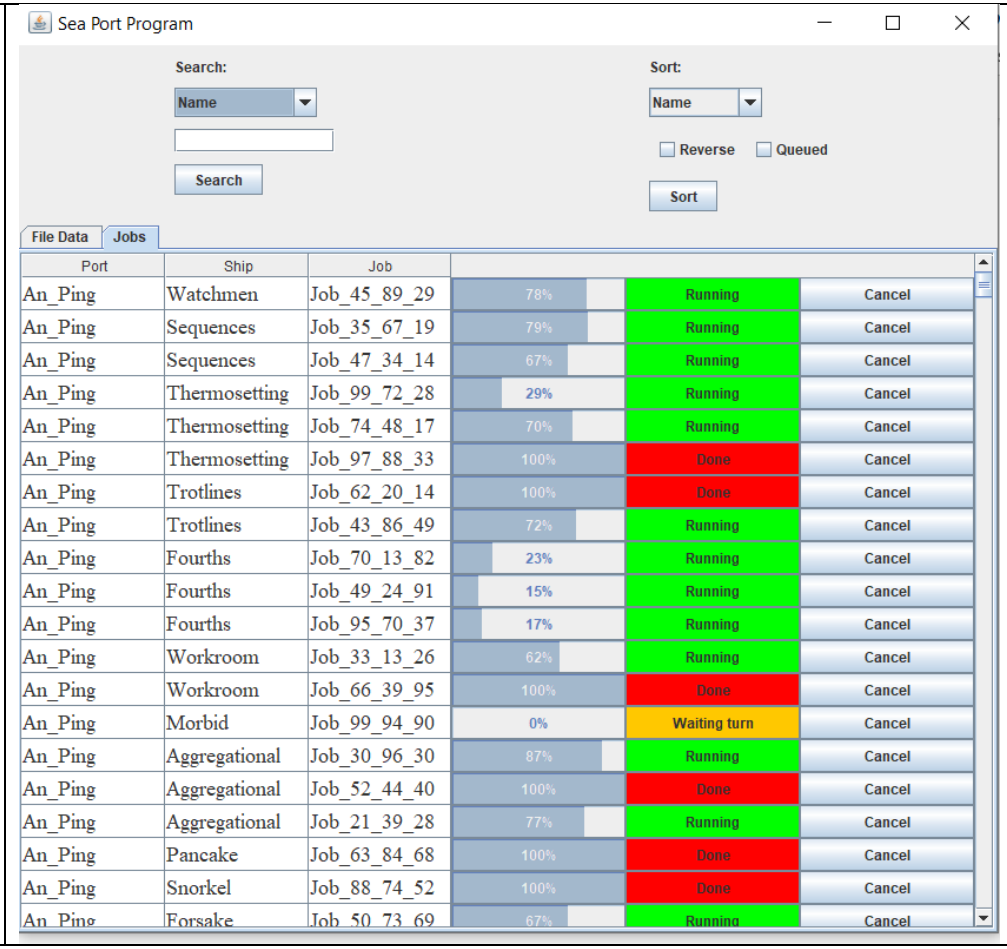
			 Search File Data Jobs The World An_Ping Chimbote Docks Ships Iterating Remounting Jobs Job_58_74 Job_96_63 Retrievers Cavie Reminiscence Rulers		

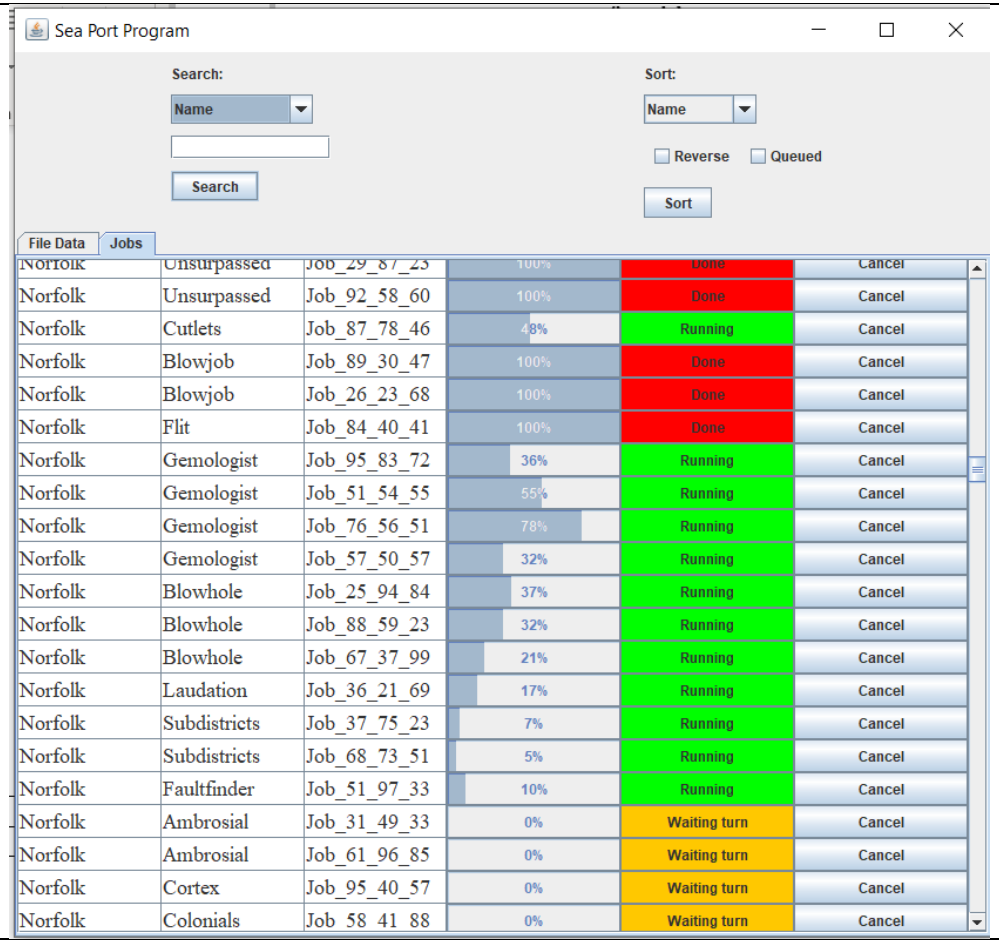
			 Search File Data Jobs Chimbote Docks Ships Queued Ships Cavie Rulers Jobs Job_57_36 Job_88_85 Budgeting Constancy Coaming Interjectory		

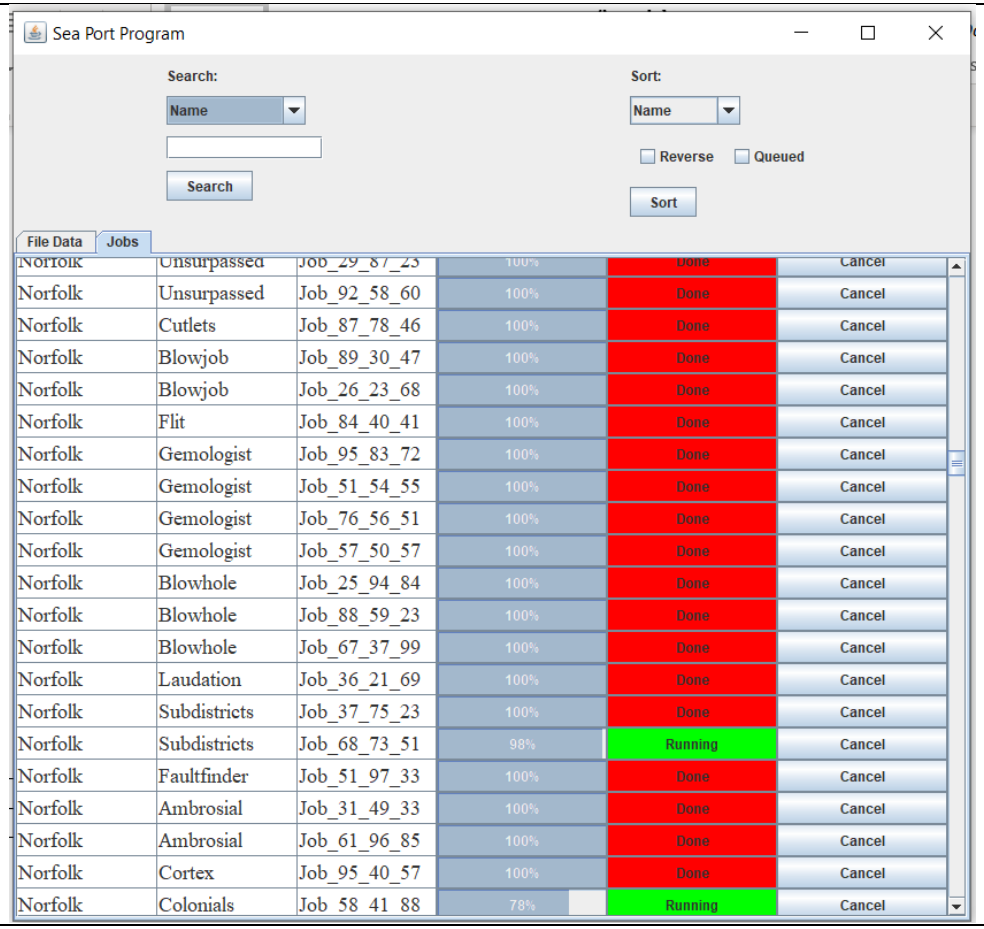
			 Search File Data Jobs The World An_Ping Chimbote Docks Ships Queued Ships People Ramona electrician Jeannette Janis Christie Jerome Andy Norfolk Damietta Cologne Athens Lubljana Santiago >>>>>The World: SeaPort: An_Ping 1 Dock: Pier_69 200 Ship: Passenger S Jobs: -Job: Job_30_96_ -Job: Job_52_44_ -Job: Job_21_39_ Dock: Pier_58 200 Ship: Passenger S Jobs: -Job: Job_50_73_ -Job: Job_74_71_ Dock: Pier_60 200 Ship: Passenger S Jobs: -Job: Job_84_38_ Dock: Pier_64 200 Ship: Passenger S		

Jobs Panel Test	Open the file and watch the jobs go	The jobs of the ships currently docked will run first. As each ship completes all jobs the ship will depart the dock and the next ship in the que will take its place.			

Port	Ship	Job			
An_Ping	Watchmen	Job_45_89_29	0%	Waiting turn	Cancel
An_Ping	Sequences	Job_35_67_19	9%	Running	Cancel
An_Ping	Sequences	Job_47_34_14	7%	Running	Cancel
An_Ping	Thermosetting	Job_99_72_28	0%	Waiting turn	Cancel
An_Ping	Thermosetting	Job_74_48_17	0%	Waiting turn	Cancel
An_Ping	Thermosetting	Job_97_88_33	0%	Waiting turn	Cancel
An_Ping	Trotlines	Job_62_20_14	29%	Running	Cancel
An_Ping	Trotlines	Job_43_86_49	8%	Running	Cancel
An_Ping	Fourths	Job_70_13_82	0%	Waiting turn	Cancel
An_Ping	Fourths	Job_49_24_91	0%	Waiting turn	Cancel
An_Ping	Fourths	Job_95_70_37	0%	Waiting turn	Cancel
An_Ping	Workroom	Job_33_13_26	7%	Running	Cancel
An_Ping	Workroom	Job_66_39_95	23%	Running	Cancel
An_Ping	Morbid	Job_99_94_90	0%	Waiting turn	Cancel
An_Ping	Aggregational	Job_30_96_30	10%	Running	Cancel
An_Ping	Aggregational	Job_52_44_40	14%	Running	Cancel
An_Ping	Aggregational	Job_21_39_28	8%	Running	Cancel
An_Ping	Pancake	Job_63_84_68	26%	Running	Cancel
An_Ping	Snorkel	Job_88_74_52	15%	Running	Cancel
An_Ping	Forsake	Job_50_73_69	7%	Running	Cancel


Sea Port Program
Search:
Name
Search
Sort:
Name
Reverse
Queued
Sort
File Data
Jobs
Port
Ship
Job
An_Ping Watchmen Job_45_89_29 78% Running Cancel
An_Ping Sequences Job_35_67_19 79% Running Cancel
An_Ping Sequences Job_47_34_14 67% Running Cancel
An_Ping Thermosetting Job_99_72_28 29% Running Cancel
An_Ping Thermosetting Job_74_48_17 70% Running Cancel
An_Ping Thermosetting Job_97_88_33 100% Done Cancel
An_Ping Trotlines Job_62_20_14 100% Done Cancel
An_Ping Trotlines Job_43_86_49 72% Running Cancel
An_Ping Fourths Job_70_13_82 23% Running Cancel
An_Ping Fourths Job_49_24_91 15% Running Cancel
An_Ping Fourths Job_95_70_37 17% Running Cancel
An_Ping Workroom Job_33_13_26 62% Running Cancel
An_Ping Workroom Job_66_39_95 100% Done Cancel
An_Ping Morbid Job_99_94_90 0% Waiting turn Cancel
An_Ping Aggregational Job_30_96_30 87% Running Cancel
An_Ping Aggregational Job_52_44_40 100% Done Cancel
An_Ping Aggregational Job_21_39_28 77% Running Cancel
An_Ping Pancake Job_63_84_68 100% Done Cancel
An_Ping Snorkel Job_88_74_52 100% Done Cancel
An_Ping Forsake Job_50_73_69 67% Running Cancel


Sea Port Program
Search:
Name
Search
Sort:
Name
Reverse
Queued
Sort
File Data
Jobs
Norfolk Unsurpassed Job_92_58_60 100% Done Cancel
Norfolk Cutlets Job_87_78_46 48% Running Cancel
Norfolk Blowjob Job_89_30_47 100% Done Cancel
Norfolk Blowjob Job_26_23_68 100% Done Cancel
Norfolk Flit Job_84_40_41 100% Done Cancel
Norfolk Gemologist Job_95_83_72 36% Running Cancel
Norfolk Gemologist Job_51_54_55 55% Running Cancel
Norfolk Gemologist Job_76_56_51 78% Running Cancel
Norfolk Gemologist Job_57_50_57 32% Running Cancel
Norfolk Blowhole Job_25_94_84 37% Running Cancel
Norfolk Blowhole Job_88_59_23 32% Running Cancel
Norfolk Blowhole Job_67_37_99 21% Running Cancel
Norfolk Laudation Job_36_21_69 17% Running Cancel
Norfolk Subdistricts Job_37_75_23 7% Running Cancel
Norfolk Subdistricts Job_68_73_51 5% Running Cancel
Norfolk Faultfinder Job_51_97_33 10% Running Cancel
Norfolk Ambrosial Job_31_49_33 0% Waiting turn Cancel
Norfolk Ambrosial Job_61_96_85 0% Waiting turn Cancel
Norfolk Cortex Job_95_40_57 0% Waiting turn Cancel
Norfolk Colonials Job_58_41_88 0% Waiting turn Cancel



4. Lessons Learned

So, this project was a lot more difficult and time intensive than I anticipated. That is probably because it needed a lot of components and methods that I had never used before. Figuring out how to set up the JTable took a long time and googling. I just could get it to work right. I am amazed at how many pieces were needed to make it work. I am also pretty sure there is another way to build the table. I used panels like the example did, however, I found that you can turn a cell in the JTable into

a component if you use the right methods. I was already down the panel route when I found this out and switching was going to take too much time.

Time was my enemy on this project. I started earlier than usual but I still was crunched for time because each piece took a while to figure out and implement correctly. For instance, the Status buttons weren't displaying correctly at first. I had to find the reason why it was broken and find a solution.

Getting the threads to work (which I've never used before either) wasn't so bad. I don't think I needed to synchronize both of the methods that I did. I am not really sure which one I should if either.

Accessing information from dependent/child classes to parent classes (from job to ship, or ship to dock) was difficult with how the program was set up. Honestly, I began to think it would have been easier to redo the whole program with the new functionality in mind. My initial set up just wasn't conducive to the level of information sharing and access that I need to write this program well. Using the tree as the database instead of the arraylists may have helped. One thing I would do is make the que arraylist an actual que. That would have helped with getting the ships from the que to open docks easier. I might try and implement that for project 4.

One thing I didn't do was use PortTime. Was I supposed too? It didn't seem that useful, so I just used the current time in milliseconds as my time for the Job class.

All in all, this was a very challenging and fun experience. I am going to need to start the next project earlier than I did this one if it will be a similar level of difficulty.