Jake

Jeremy

5/24/2018

My approach to this project was to first open and parse through the input file line by line. To do this “while(std::getline(inFile,str))” is used. This allows the program to parse each line independently. For each line there are “if” conditional statements searching for ‘A’,’Q’,’L’, or ‘D’. This determines what command to run. The first function to run should (dependent on the input file) be the System Configuration function.

The System Configuration function determines the time the program should start, the total main memory the total number of serial devices, and the time quantum. All these values are assigned to global variables and prints the values to console.

The next function to run is Job Arrival. This function will determine the arrival time, job number, required memory, it’s maximum demand, the total run time of the Job. Upon parsing and assigning this information to variables, this function will determine if the Job is possible based on the total system memory and the total number of concurrent devices the system supports. If the job is not possible the console will print “Not possible”.

Otherwise if there is enough memory currently available the Job will be added to the ready queue.

The next function to run is Request Device and it’s job is to determine what to run. This function parses through the line and determines time, job, device requests. After determining these variables, the function will call a Run function.

The Run function takes a Job, time request, and number of devices as parameters.
It allocates the memory and sets the Job to “Running”.

Release Device is a function that, upon command releases the specified Job. It gives the memory allocated to the job back to main memory. Afterwards the Job that has been released is noted and added to a released Queue list.

The display Function prints the variables associated with the command as well as printing all the values associated with the Hold1, Hold2, and ready queue to the console .

When it came to classifying job type, we decided to create a class called Jobs with the following fields: Job_Number[String],

Job_Memory[string], Job_Priority[string], Job_Devices[int], Job_Runtime[int],

Job_Arrivaltime[string], Job_devicesRunning[int],

Job_Running[bool], Job_onHold2[bool], Job_Requested[bool], and Job_Released[bool]. We gave the class no methods at it was only

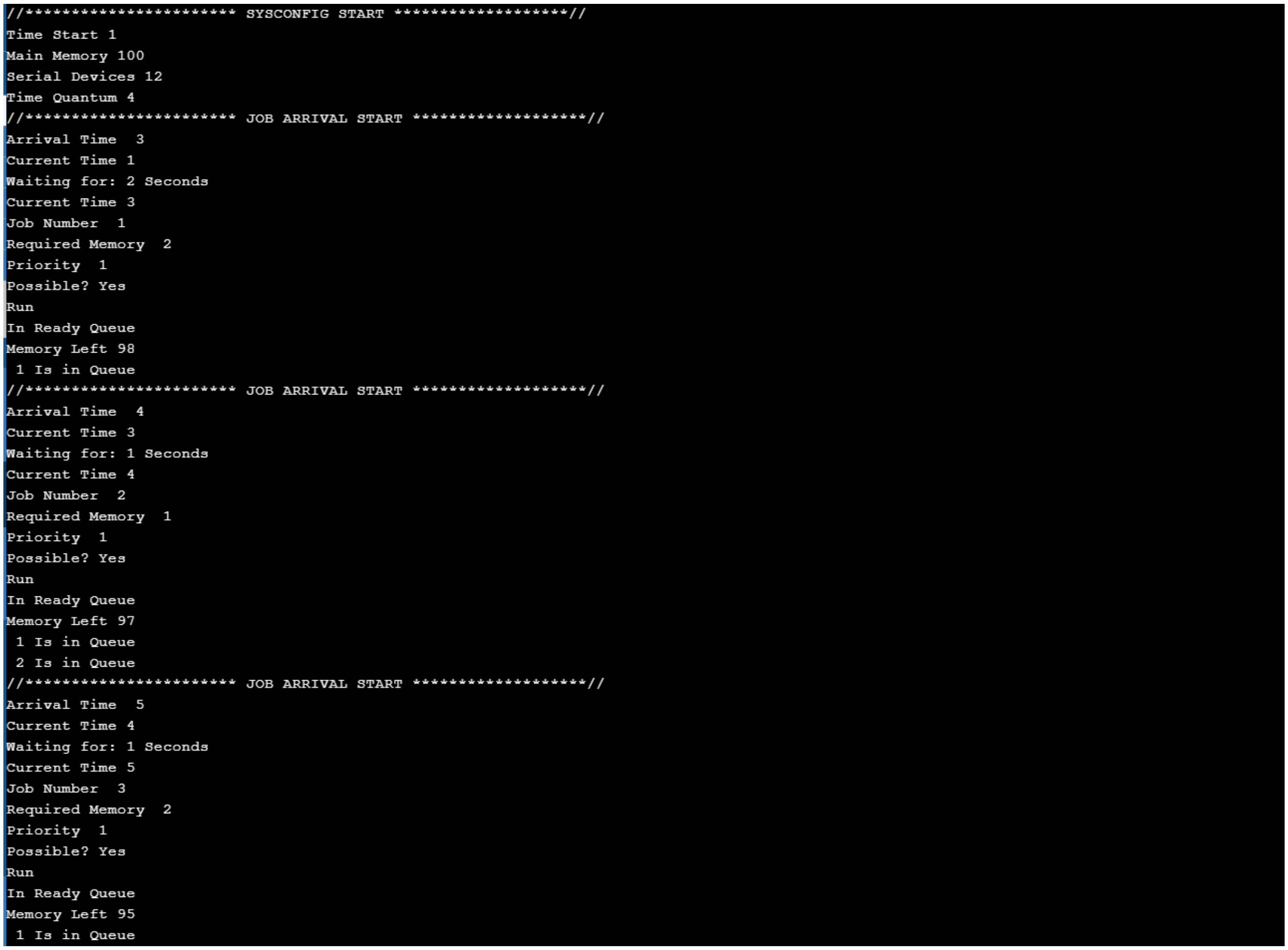
intended as a means to store the information given to us. With this data structure in place, we were able to use the nodes

in the Lists class. The Lists class has three fields which are the following: node[nodePtr], curr[nodePtr], head[nodePtr],

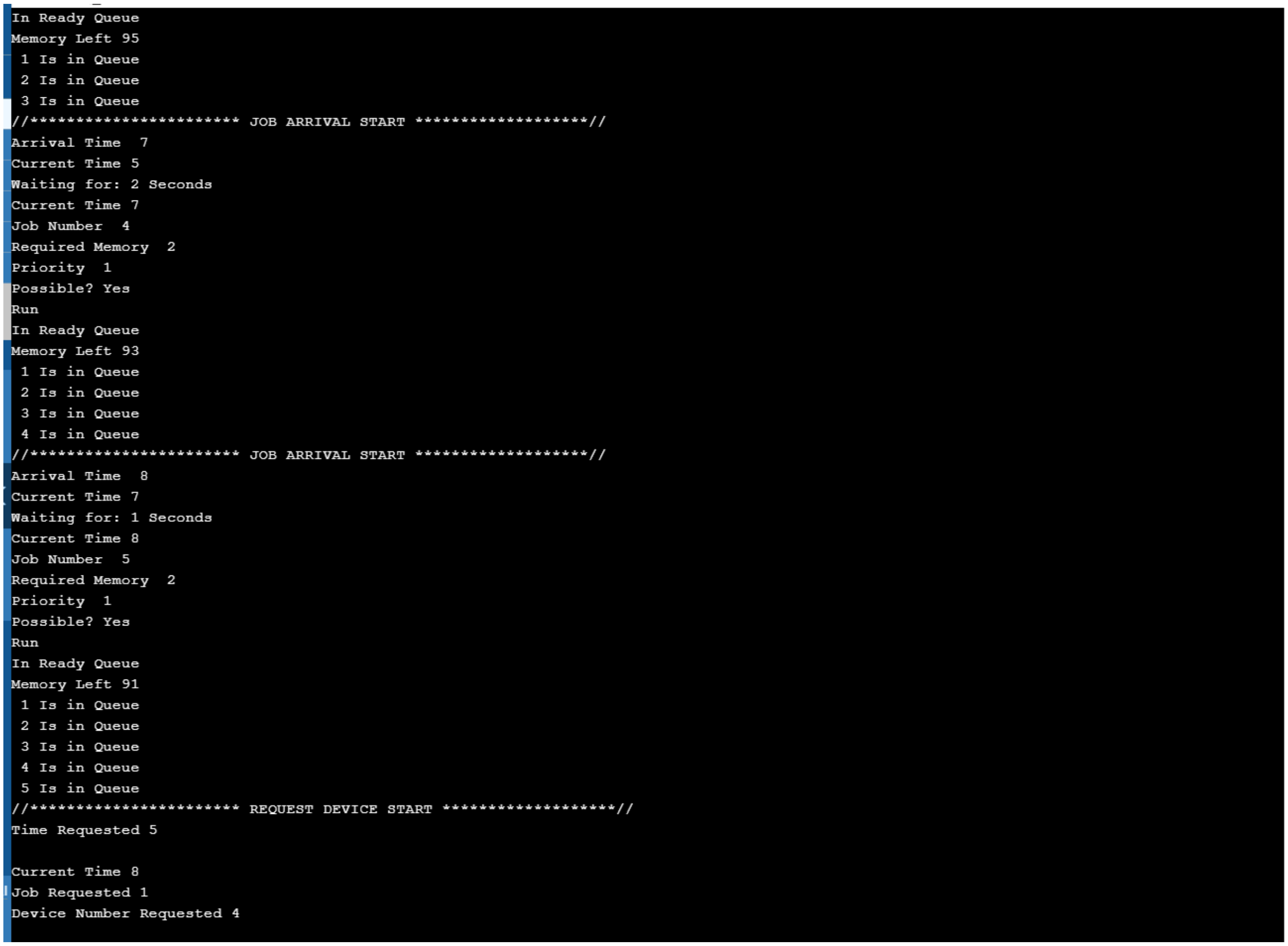
and temp[nodePtr]. It has a constructor method, a sort method, an addNode method, a deleteNode method, a findNode method,

a PrintList method, a pop method, and a swap method.

SAMPLE OUTPUT

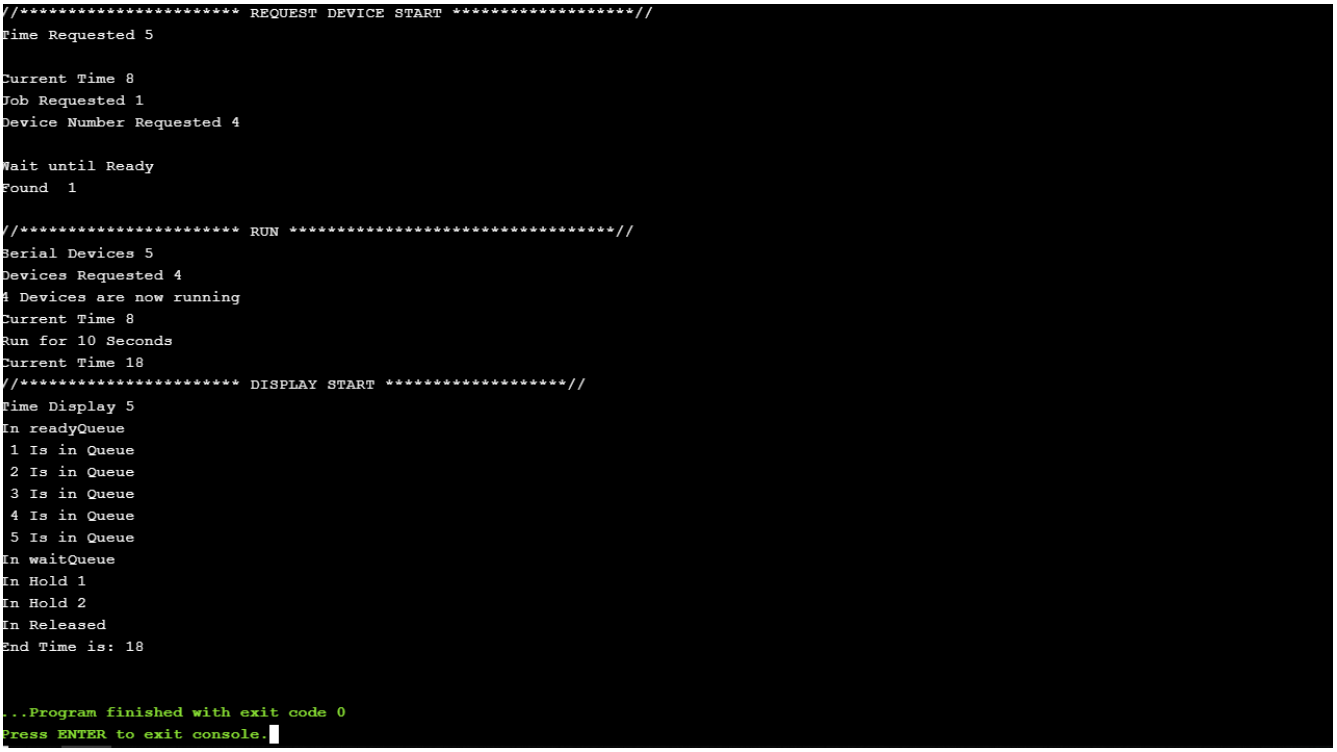


```
//*********************** SYSCONFIG START *******************//
Time Start 1
Main Memory 100
Serial Devices 12
Time Quantum 4
//*********************** JOB ARRIVAL START *******************//
Arrival Time  3
Current Time 1
Waiting for: 2 Seconds
Current Time 3
Job Number  1
Required Memory  2
Priority  1
Possible? Yes
Run
In Ready Queue
Memory Left 98
 1 Is in Queue
//*********************** JOB ARRIVAL START *******************//
Arrival Time  4
Current Time 3
Waiting for: 1 Seconds
Current Time 4
Job Number  2
Required Memory  1
Priority  1
Possible? Yes
Run
In Ready Queue
Memory Left 97
 1 Is in Queue
 2 Is in Queue
//*********************** JOB ARRIVAL START *******************//
Arrival Time  5
Current Time 4
Waiting for: 1 Seconds
Current Time 5
Job Number  3
Required Memory  2
Priority  1
Possible? Yes
Run
In Ready Queue
Memory Left 95
 1 Is in Queue
```



```
In Ready Queue
Memory Left 95
 1 Is in Queue
 2 Is in Queue
 3 Is in Queue
//*********************** JOB ARRIVAL START *******************//
Arrival Time  7
Current Time 5
Waiting for: 2 Seconds
Current Time 7
Job Number  4
Required Memory  2
Priority  1
Possible? Yes
Run
In Ready Queue
Memory Left 93
 1 Is in Queue
 2 Is in Queue
 3 Is in Queue
 4 Is in Queue
//*********************** JOB ARRIVAL START *******************//
Arrival Time  8
Current Time 7
Waiting for: 1 Seconds
Current Time 8
Job Number  5
Required Memory  2
Priority  1
Possible? Yes
Run
In Ready Queue
Memory Left 91
 1 Is in Queue
 2 Is in Queue
 3 Is in Queue
 4 Is in Queue
 5 Is in Queue
//*********************** REQUEST DEVICE START *******************//
Time Requested 5

Current Time 8
Job Requested 1
Device Number Requested 4
```



```
//*********************** REQUEST DEVICE START *******************//
Time Requested 5

Current Time 8
Job Requested 1
Device Number Requested 4

Wait until Ready
Found  1

//*********************** RUN **********************************//
Serial Devices 5
Devices Requested 4
4 Devices are now running
Current Time 8
Run for 10 Seconds
Current Time 18
//*********************** DISPLAY START *******************//
Time Display 5
In readyQueue
 1 Is in Queue
 2 Is in Queue
 3 Is in Queue
 4 Is in Queue
 5 Is in Queue
In waitQueue
In Hold 1
In Hold 2
In Released
End Time is: 18


...Program finished with exit code 0
Press ENTER to exit console.
```