Lab Sheet 1

Name : Lokesh Chandra Basu
E.No. : 10114026
Course : Operating Systems Lab

Question 3

Implement Queue Data Structure. Try to implement various queues like Job Queue, Ready Queue, and Waiting Queue using the data structure developed. Initially, we will use PID(Process ID) (an integer) as the data to be stored in these Queues. You should keep in mind that, in future, they can be modified to store Process Control Block(PCB).

Algorithm

Algorithm used :

```
//to create the Ready Queue
take input from the user for the number of queues in each of the Ready Queue
Initialize count to 0
    While count is less then input number
        generate a random number to be stored in queue as pID
        count = count +1

//to create the Job Queue
 take input from the user for the number of queues in each of the Job Queue
Initialize count to 0
    While count is less then input number
        generate a random number to be stored in queue as pID
        count = count +1

//to create the Waiting Queue
 take input from the user for the number of queues in each of the Waiting
 Queue
Initialize count to 0
    While count is less then input number
        generate a random number to be stored in queue as pID
        count = count +1

//to display the Ready Queue
initialize count to zero again
    while count is less then length.readyQueue
        display info.readyQueue

//to display the Job Queue
initialize count to zero again
    while count is less then length.jobQueue
        display info.jobQueue

//to display the Waiting Queue
initialize count to zero again
    while count is less then length.waitQueue
        display info.waitQueue
```

Gantt Chart

Procees	Execution Tiem(in ms)
P1	5
P2	15
P3	10
P4	7
P5	3

Job Queue with processes and execution time (assuming all arrives at same time) :

P1	P2	P3	P4	P5

0 5 20 30 37 40

After Long Term Scheduler or Job Scheduler :

Ready Queue :

P1	P2	P3	P4	P5

0 5 20 30 37 40

time elapsed = 0ms

After Short Term Scheduler or Scheduler Dispatch :

Ready Queue :

P2	P3	P4	P5

5 20 30 37 40

Running Queue :

P1(Running)

0 5

time elapsed = 5ms

Ready Queue :

P3	P4	P5

20 30 37 40

Running Queue :

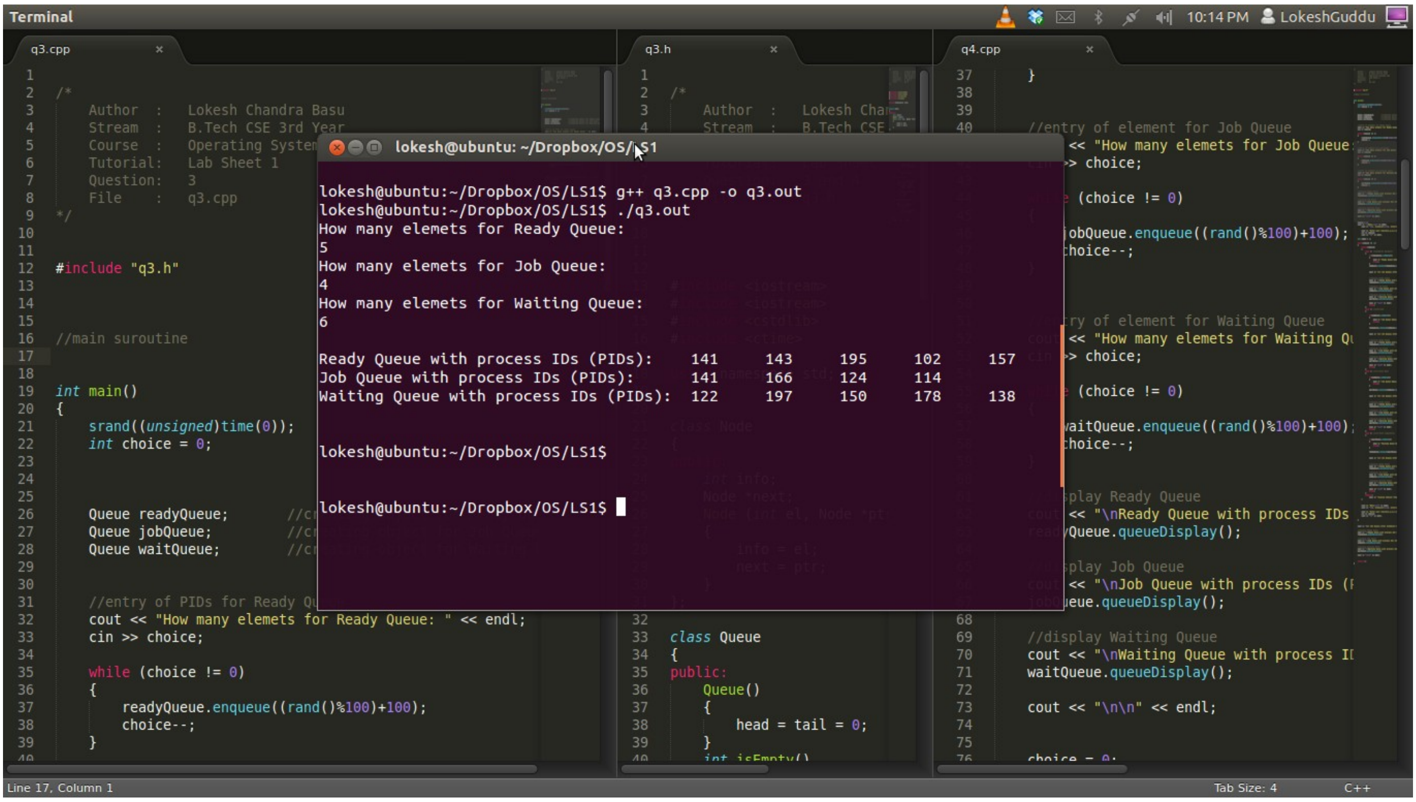
P1(Terminated)	P2(Running)

0 5 20

time elapsed = 20ms (just after P2 terminates)

And So on...

Output



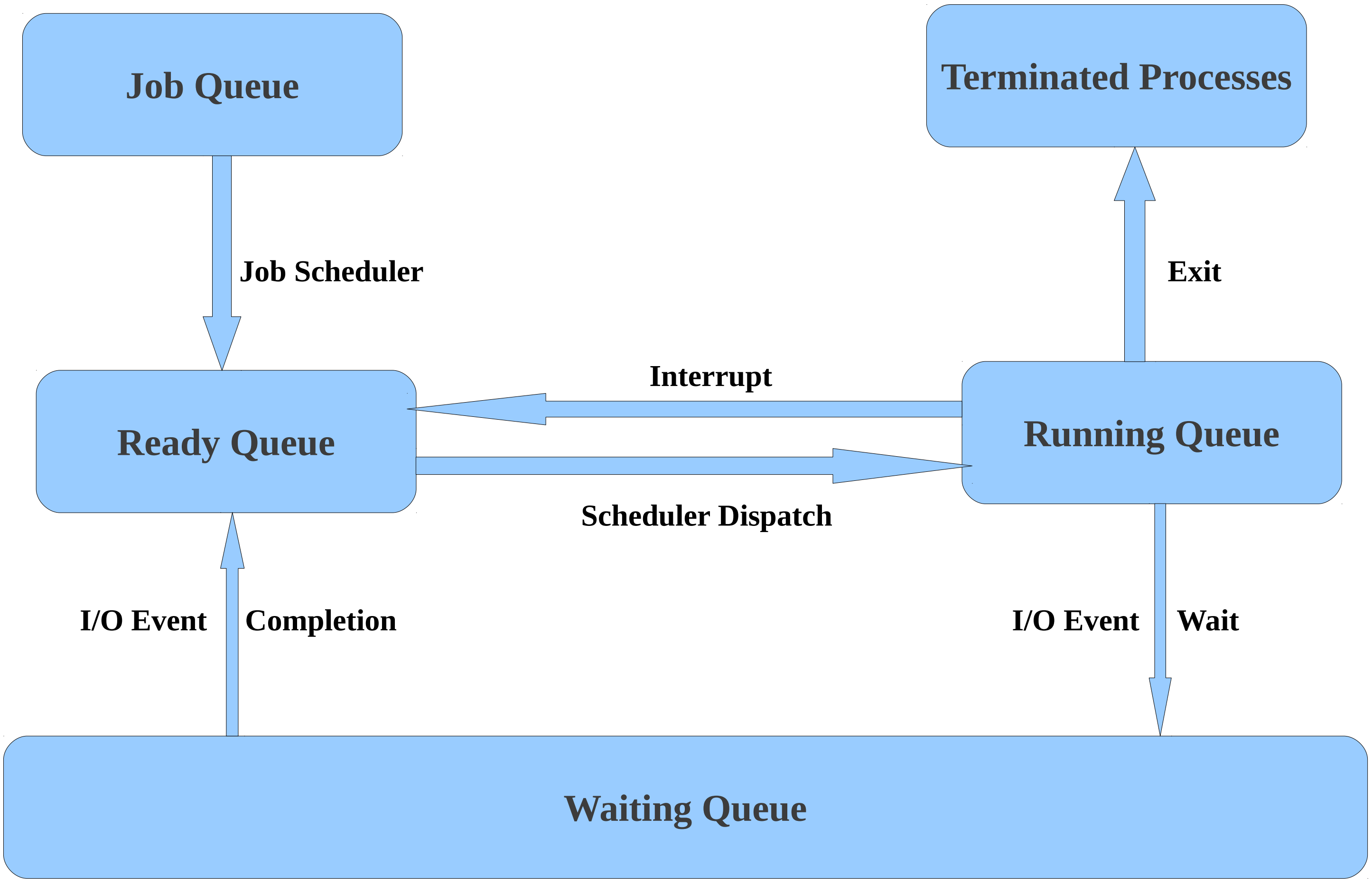
```
lokesh@ubuntu:~/Dropbox/OS/LS1$ g++ q3.cpp -o q3.out
lokesh@ubuntu:~/Dropbox/OS/LS1$ ./q3.out
How many elemets for Ready Queue:
5
How many elemets for Job Queue:
4
How many elemets for Waiting Queue:
6

Ready Queue with process IDs (PIDs):    141     143     195     102     157
Job Queue with process IDs (PIDs):      141     166     124     114
Waiting Queue with process IDs (PIDs):  122     197     150     178     138


lokesh@ubuntu:~/Dropbox/OS/LS1$


lokesh@ubuntu:~/Dropbox/OS/LS1$ 
```

gcc version 4.6.3 (Ubuntu/Linaro 4.6.3-1 ubuntu5)

Question 4

Simulate (Diagrammatically, if Possible) various Process States. Three key strokes (or buttons) can be used as input to represent a scheduler, an interrupt, and the I/O event. The program should be able use the queues designed in question 3.

Solution to question 4



Algorithm

Algorithm used :

```
//to create the Ready Queue
take input from the user for the number of queues in each of the Ready Queue
Initialize count to 0
    While count is less then input number
        generate a random number to be stored in queue as pID
        count = count +1

//to create the Job Queue
 take input from the user for the number of queues in each of the Job Queue
Initialize count to 0
    While count is less then input number
        generate a random number to be stored in queue as pID
        count = count +1

//to create the Waiting Queue
 take input from the user for the number of queues in each of the Waiting
 Queue
Initialize count to 0
    While count is less then input number
        generate a random number to be stored in queue as pID
        count = count +1
```

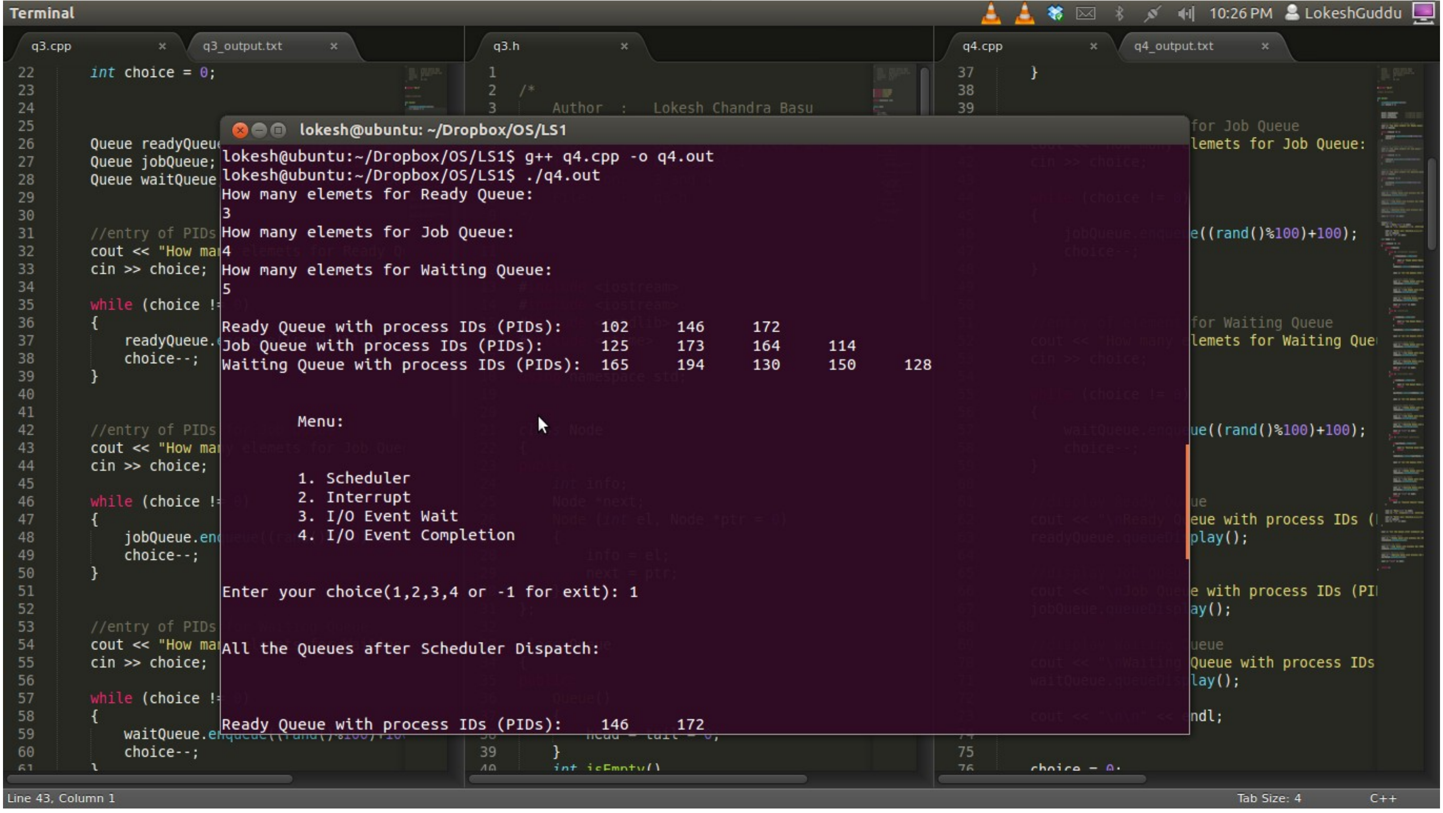
to display the Menu with all the options:

1. ***Scheduler***
2. ***Interrupt***
3. ***I/O Event Wait***
4. ***I/O Event Completion***

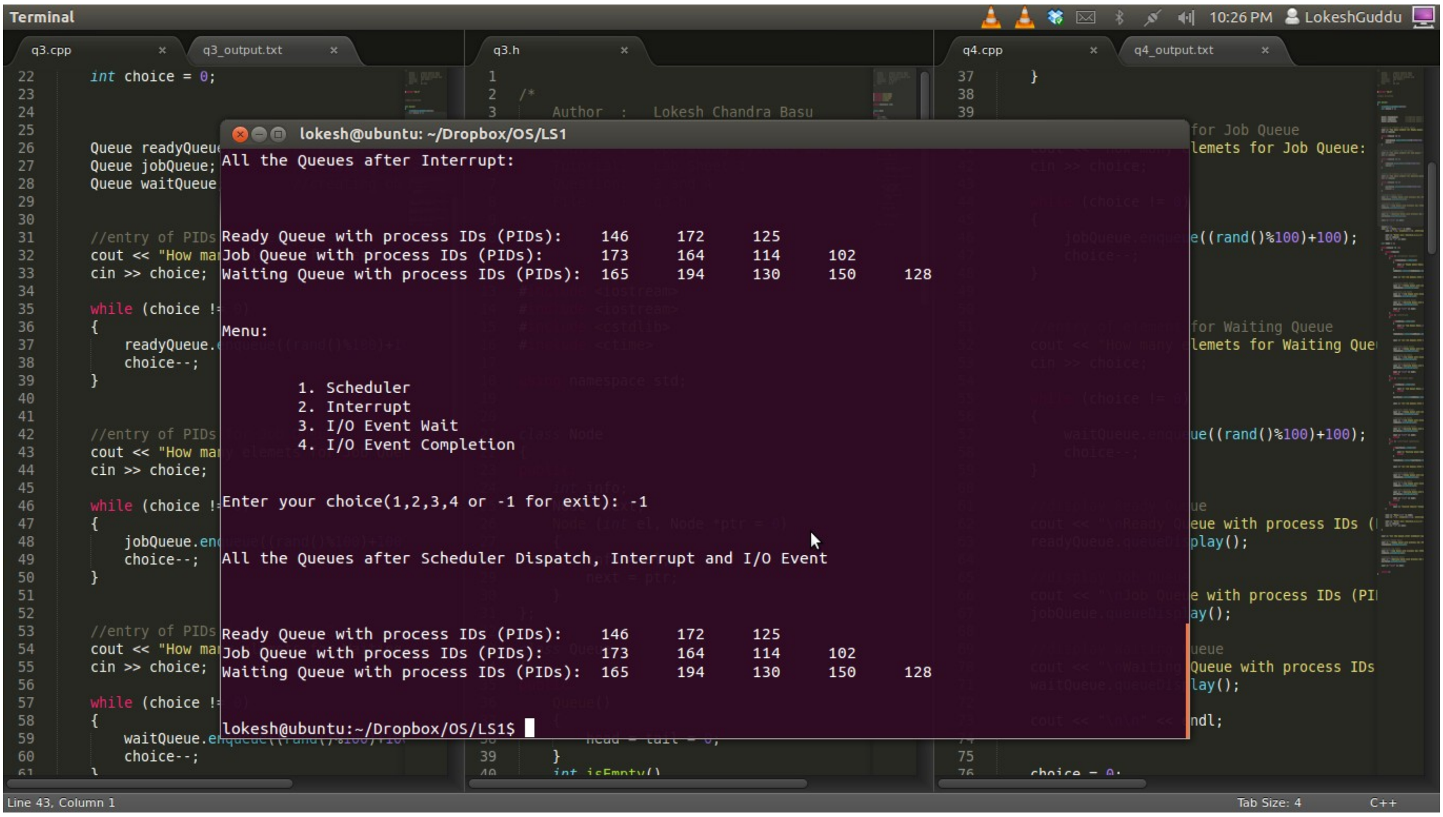
take input from the user for the operation to perform

```
switch choice (1 to 4)
    perform the operation according to choice

    1. scheduler will dequeue Ready Queue and enqueue Running Queue
    2. interrupt will dequeue Running Queue and enqueue Ready Queue
    3. I/O Event wait will dequeue Running Queue and enqueue Waiting Queue
    4. I/O Event wait will dequeue Waiting Queue and enqueue Ready Queue



if the choice is other then 1 to 4 then the process ends and displays the contents of the queues

//to display the Ready Queue
initialize count to zero again
    while count is less then length.readyQueue
        display info.readyQueue

//to display the Job Queue
initialize count to zero again
    while count is less then length.jobQueue
        display info.jobQueue

//to display the Waiting Queue
initialize count to zero again
    while count is less then length.waitQueue
        display info.waitQueue
```

Output



```
lokesh@ubuntu:~/Dropbox/OS/LS1$ g++ q4.cpp -o q4.out
lokesh@ubuntu:~/Dropbox/OS/LS1$ ./q4.out
How many elemets for Ready Queue:
3
How many elemets for Job Queue:
4
How many elemets for Waiting Queue:
5

Ready Queue with process IDs (PIDs):    102     146     172
Job Queue with process IDs (PIDs):      125     173     164     114
Waiting Queue with process IDs (PIDs):  165     194     130     150     128



          Menu:


          1. Scheduler
          2. Interrupt
          3. I/O Event Wait
          4. I/O Event Completion


Enter your choice(1,2,3,4 or -1 for exit): 1


All the Queues after Scheduler Dispatch:



Ready Queue with process IDs (PIDs):    146     172
```



```
All the Queues after Interrupt:



Ready Queue with process IDs (PIDs):    146     172     125
Job Queue with process IDs (PIDs):      173     164     114     102
Waiting Queue with process IDs (PIDs):  165     194     130     150     128


Menu:


          1. Scheduler
          2. Interrupt
          3. I/O Event Wait
          4. I/O Event Completion


Enter your choice(1,2,3,4 or -1 for exit): -1


All the Queues after Scheduler Dispatch, Interrupt and I/O Event



Ready Queue with process IDs (PIDs):    146     172     125
Job Queue with process IDs (PIDs):      173     164     114     102
Waiting Queue with process IDs (PIDs):  165     194     130     150     128


lokesh@ubuntu:~/Dropbox/OS/LS1$
```

gcc version 4.6.3 (Ubuntu/Linaro 4.6.3-1 ubuntu5)