Testing Type Details: Unit Testing
Tester Name: Group1
Test Environment Details

Scen #	Scenario Description	Req #	Cond #	Test Data	Test Conditions/Steps	Expected Results/Comments	Post-Conditions	Actual Results	Pass/Fail (Y/N)
1	Enter priority number 0			**1-call Operation constructor with priority number 0**	1- initialize operation constructor	Throw illegal argument exception	exit	Throw illegal argument exception	Y
2	Enter priority number -1			**1-call Operation constructor with priority number -1**	1- initialize operation constructor	Throw illegal argument exception	exit	Throw illegal argument exception	Y
3	Enter priority number 1			**1-call Operation constructor with priority number 1**	1- initialize operation constructor 2- Call getpriority()	work normally	exit	Work normally	Y
4	Enter priority number 10			**1-call Operation constructor with priority number 10**	1- initialize operation constructor 2- Call getpriority()	Work normally	exit	Work normally	Y
5	Enter priority number 100			**1-call Operation constructor with priority number 100**	1- initialize operation constructor	Throw illegal argument exception	exit	Throw illegal argument exception	Y
6	Enter priority number 9	b		**1-call Operation constructor with priority number 9**	1- initialize operation constructor 2- Call getpriority()	Work normally	exit	worknormally	Y

Testing Type Details: Unit Testing
Tester Name: Group1
Test Environment Details

Scen #	Scenario Description	Req #	Cond #	Test Data	Test Conditions/Steps	Expected Results/Comments	Post-Conditions	Actual Results	Pass/Fail (Y/N)
7	Enter priority number 5			**1-Intailze Operation constructor with priority number 5**	1- initialize operation constructor 2- Call getpriority()	Work normally	exit	Work normally	Y
8	Enter priority number 10 and increment it			**1-Inalize Operation constructor with priority number 10 2-incrementprioity(20)**	1- initialize operation constructor 2- Call incrementPrioity(20)	Throws illegal argument exception	exit	Throw illegal argument exception	Y
9	Enter ID negative number			**1-Inalize Operation constructor with ID number -1**	1- initialize operation constructor with Test data	Throws illegal argument exception	exit	Doesn't throw anything and crash	Y
10	Enter arrival Time negative number			**1-Inalize Operation constructor with arrivalTime number -2**	1- initialize operation constructor with Test Data	Throws illegal argument exception	exit	Throw illegal argument exception	Y
11	Enter exeTime negative number	c		**1-Inalize Operation constructor with exeTime -1**	1- initialize operation constructor with Test Data	Throws illegal argument exception	exit	Throw illegal argument exception	Y

Testing Type Details: Unit Testing
Tester Name: Group1
Test Environment Details

Scen #	Scenario Description	Req #	Cond #	Test Data	Test Conditions/Steps	Expected Results/Comments	Post-Conditions	Actual Results	Pass/Fail (Y/N)
12	Enter exeTime zero			**1-Inalize Operation constructor with exeTime 0**	1- initialize operation constructor with Test Data	Throws illegal argument exception	exit	Throw illegal argument exception	Y
13	Enter valid data of the constructor			**1-Inalize Operation constructor with all valid data**	1- initialize operation constructor with Test Data	Work normally	exit	Work normally	Y
14	Enter exetime 1 and decrement it two times with decrementTimeLeft(2)			**1-Inalize Operation constructor with exeTime 1** **2- decrementTimeleft(2)**	1- initialize operation constructor with Test Data	Throws illegal argument exception	exit	Throw illegal argument exception	Y
15	Enter a valid constructor and call getWaiting() as Response time is MAX_VALUE so it return -1 in getTAT() but getwait will be a negative number			**1-Inalize Operation constructor with valid input**	1- initialize operation constructor with Test Data 2- call getwaiting()	Throws illegal argument exception	exit	Throw illegal argument exception	Y

Testing Type Details: Unit Testing
Tester Name: Group1
Test Environment Details

Scen #	Scenario Description	Req #	Cond #	Test Data	Test Conditions/Steps	Expected Results/Comments	Post-Conditions	Actual Results	Pass/Fail (Y/N)
16	Enter a valid constructor And setResponse time with negative number			**1-Inalize Operation constructor with valid input** **2-setResponseTime(-2)**	1- initialize operation constructor with Test Data 2- call setResponse(-2)	Throws illegal argument exception	exit	Throw illegal argument exception	Y
17	Enter a valid constructor And setResponse time with negative number And call getTAT will be negative number not -1			**1-Inalize Operation constructor with valid input** **2-setResponseTime(-2)**	1- initialize operation constructor with Test Data 2- call getTATime()	Throws illegal argument exception	exit	Throw illegal argument exception	Y
18	FCFS enqueue it when the arrival time in the past and the timer has gone			**1-Inalize Operation constructor with arrival time less than timer** **2-Inalize 1 Operations with valid input**	**1-Inalize Operation constructor with arrival time 0 timer** **2-Inalize 1 Operation with valid input** **3-call enqueue()** **4-call consumeTimeUnit()** **5-call enqueue with the operation arrivalTime less than timer**	Throws illegal argument exception	exit	Throw illegal argument exception	Y
19	if the queue in FCFSQ is null and call consumeTimeUnit			**1-intalize FCFSQ**	1- **intalize FCFSQ** 2- **call consumeTimeUnit()**	Return null	exit	Return null	Y

Testing Type Details: Unit Testing
Tester Name: Group1
Test Environment Details

Scen #	Scenario Description	Req #	Cond #	Test Data	Test Conditions/Steps	Expected Results/Comments	Post-Conditions	Actual Results	Pass/Fail (Y/N)
20	Put one operation in queue and consume time unit with exeTime 1			**1-intalize FCFSQ**	1- **intalize FCFSQ** 2- **inalize Operation with exeTime 1** 3- **call consumeTimeUnit()** 4- **call consumeTimeUnit() again**	Return null	exit	Return null	Y
21	Empty iterator in FCFS			**1-inalize FCFSQ**	1-inalize FCFSQ 2-getiterator()	Work normally	exit	Work normally	Y
22	SJFQ enqueue it when the arrival time in the past and the timer has gone			**1-Inalize Operation constructor with arrival time less than timer** **2-Inalize 1 Operations with valid input**	**1-Inalize Operation constructor with arrival time 0 timer** **2-Inalize 1 Operation with valid input** **3-call enqueue()** **4-call consumeTimeUnit()** **5-call enqueue with the operation arrivalTime less than timer**	Throws illegal argument exception	exit	Throws illegal argument exception	Y
23	Temp queue is empty			1- **inalize SJFSQ**	1-inalize SJFSQ 2-inalize 3 operation with arrivalTime=Timer 3-enqueue them 4-call consumeTimeUnit()	Work normally and decrement the first Operation as it is lowest exetime	exit	Work normally	Y
24	In SJF queue is empty			**1-inalize SJFSQ**	1- Inalize SJFSQ 2- Call consumeTime() 3- Call consumeTime()agai	Return null	Exit	Return null	Y

Testing Type Details: Unit Testing
Tester Name: Group1
Test Environment Details

Scen #	Scenario Description	Req #	Cond #	Test Data	Test Conditions/Steps	Expected Results/Comments	Post-Conditions	Actual Results	Pass/Fail (Y/N)
					n				
25	We will put all arrivalTime queues more than timer so it will be in Temp queues			1- **Inalize SJFSQ** 2- **Inalize 4 Operations with valid data but arrival time bigger than timer**	1- Inalize SJFSQ 2- Inalize 4 operations with valid data but arrival time bigger than timer 3- Enqueue it 4- Call 2 times ConsumeTime()	Return null in first call but second call work normally	exit	Return null in first call but second call work normally	Y
26	One process at queue			1- **inalize SJFSQ** 2- **inalize one operation with 1 exeTime**	1- inalize SJFSQ 2- inalize Operation with valid input but exeTime 1 3- enqueue it 4- ConsumeTimeUnit( ) 5- Call consumeTimeUnit( ) again	Return null	exit	Return null	Y
27	In PreemptiveSJFQ queue is empty			**1-inalize** PreemptiveSJFQ	1- Inalize PreemptiveSJFQ 2- Call consumeTime	Return null	exit	Return null	Y
28	We will put all arrivalTime queues more than timer so it will be in Temp queues			1- **Inalize** PreemptiveSJFQ 2- **Inalize 4 Operations with valid data but**	1- Inalize PreemptiveSJFQ 2- Inalize 4 operations with valid data but arrival time bigger than timer	Return null in first call but second call work normally	exit	Return null in first call but second call work normally	Y

Testing Type Details: Unit Testing
Tester Name: Group1
Test Environment Details

Scen #	Scenario Description	Req #	Cond #	Test Data	Test Conditions/Steps	Expected Results/Comments	Post-Conditions	Actual Results	Pass/Fail (Y/N)
				arrival time bigger than timer	3- Enqueue it 4- Call consumeTimeUnit( ) two times				
29	One process at queue			1- **inalize** Preemptive **SJFQ** 2- **inalize one operation with 1 exeTime**	1- inalize SJFSQ 2- inalize Operation with valid input but exeTime 1 3- enqueue it 4- ConsumeTimeUnit( ) 5- Call consumeTimeUnit( ) again	Return null	exit	Return null	Y
Round Robin									
30	RoundRobin enqueue it when the arrival time in the past and the timer has gone			**1-Inalize Operation constructor with arrival time less than timer** **2-Inalize 1 Operations with valid input**	**1-Inalize Operation constructor with arrival time 0 timer** **2-Inalize 1 Operation with valid input** **3-call enqueue()** **4-call consumeTimeUnit()** **5-call enqueue with the operation arrivalTime less than timer**	Throws illegal argument exception	exit	Throws illegal argument exception	Y

Testing Type Details: Unit Testing
Tester Name: Group1
Test Environment Details

Scen #	Scenario Description	Req #	Cond #	Test Data	Test Conditions/Steps	Expected Results/Comments	Post-Conditions	Actual Results	Pass/Fail (Y/N)
31	In RoundRobin queue is empty			**1-inalize RoundRobin**	1- Inalize RoundRobin 2- Call consumeTime() 3- Call consumeTime()again	Return null	exit	Return null	Y
32	We will put all arrivalTime queues more than timer so it will be in Temp queues			**1- Inalize** RoundRobin **2- Inalize 4 Operations with valid data but arrival time bigger than timer**	1- Inalize RoundRobin 2- Inalize 4 operations with valid data but arrival time bigger than timer 3- Enqueue it 4- Call consumeTimeUnit( ) two times	Return null in first call but second call work normally	exit	Return null in first call but second call work normally	y
33	One process at queue			**1- inalize SJFSQ** **2- inalize one operation with 1 exeTime**	1- inalize RoundRobin 2- inalize Operation with valid input but exeTime 1 3- enqueue it 4- ConsumeTimeUnit( ) 5- Call consumeTimeUnit( ) again	Return null	exit	Return null	Y
34	We will put all arrivalTime queues more than timer so it will be in Temp			**1- Inalize** RoundRobin **2- Inalize 3**	1- Inalize RoundRobin 2- Inalize 3 operations with	Work normally	Exit	Work normally	Y

Testing Type Details: Unit Testing
Tester Name: Group1
Test Environment Details

Scen #	Scenario Description	Req #	Cond #	Test Data	Test Conditions/Steps	Expected Results/Comments	Post-Conditions	Actual Results	Pass/Fail (Y/N)
	queues and operation in queue will be less round robin but not equal zero			**Operations with valid data but arrival time bigger than timer** **3- Inaltize operation with exeTime less than quantum but same arrival time in ready queue**	valid data but arrival time bigger than timer 3- Enqueue it 4- Inalize Operation with arrival time equal timer with exe Time less than quantum 5- consumeTimeUnit( ) two times				
Same for Preemptive and Priority queues and success									
35	In PreemptivePriority Queue different priority but same arrival time so the highest priority will be on cpu			**1-inalize Preemptive Priority queue** **2- inalize two operation with same arrival time but different piroity**	**1-inalize Preemptive Priority queue** **2- inalize two operation with same arrival time but different piroity** **3- enqueued them** **4- check asseration**	Highest pirioity on cpu	exit	Highest pirioity on cpu	N

Testing Type Details: Unit Testing
Tester Name: Group1
Test Environment Details

Scen #	Scenario Description	Req #	Cond #	Test Data	Test Conditions/Steps	Expected Results/Comments	Post-Conditions	Actual Results	Pass/Fail (Y/N)
36	In PreemptivePriority Queue different priority and same arrival time and same priority so shortest job must be first will be on cpu			**1-inalize Preemptive Priority queue** **2- inalize Three operation with same arrival time and same piroity**	**1-inalize Preemptive Priority queue** **2- inalize three operation with same arrival time and same piroity** **3- enqueued them** **4- check asseration**	FCFS	exit	FCFS	y
37	Same id								
38	In Preemptive Shortestjob first Queue different arrival but same brust after consumeTimeUnit so the earliest arrival Time will be on CPU			**1-inalize Preemptive ShortestJob first queue** **2- inalize three operation with different arrival time but same brust after consume**	**1-inalize Preemptive SJF queue** **2- inalize three operation with different arrival time but same brust after consume** **3- enqueued them** **4- check asseration**	FIRST ARRIVAL STILL ON CPU	exit	FIRST ARRIVAL STILL ON CPU	y
39	In Preemptive Shortestjob first Queue same arrival and same brust but different piroity after consumeTimeUnit so highest piroity must enter			**1-inalize Preemptive PShortestJob first queue** **2- inalize two operation with same arrival time and same brust**	**1-inalize Preemptive PShortestJob first queue** **2- inalize two operation with same arrival time and same brust** **3- enqueued them** **4- check asseration**	Highest piroity on Cpu	exit	First one with medium piroity	Y (cancelled)

Testing Type Details: Unit Testing

Tester Name: Group1

Test Environment Details

Scen #	Scenario Description	Req #	Cond #	Test Data	Test Conditions/Steps	Expected Results/Comments	Post-Conditions	Actual Results	Pass/Fail (Y/N)
40	Duplicate process and ID			**1-inalize 3 operation with same all valid data**	**1-inalize 3 operation with same all**	Illegal argument excpetion	exit	Work normally	Y(cancelled)
41	Huge number in exeTime in FCFS or PIROIty queue					Throw illegal argument exception	Exit	WorkNormally	Y(handled by Andrew)

Testing Type Details: Unit Testing

Tester Name: Group1

Test Environment Details

Scen #	Scenario Description	Req #	Cond #	Test Data	Test Conditions/Steps	Expected Results/Comments	Post-Conditions	Actual Results	Pass/Fail (Y/N)
	112								