Schedule Project – Answer Document

Submissions

1. Place the design of your implementation in this item. You should study the problem and create a design before trying to modify the code.
   1. Place diagrams of the data structures and how they are related here.
   2. Place pseudo code of your algorithms and how they affect the data structures here.

Diagram, timeline

Description automatically generated

* 1. Place a list of source files in the code base that you updated.

Main.c

Proc.c

Def.h

Proc.h

1. Place your test cases here. Your collection of test cases should demonstrate you have tested the entire program. You will have a collection of test cases for each of you schedulers. For each test case, first describe the objective of your test case. Then  insert a copy/paste of you applying your test cases to your program. The copy/paste must demonstrate that actual output achieved the described objective.

Test Case 1- have two procs, each had 50 tickets. Scheduled 5 times

Text

Description automatically generated

Out of the 5 test cases each proc was selected almost 50% of the time

Test Case 2 – have 3 procs, each with 50 tickets. Scheduled 10 times

Graphical user interface, text

Description automatically generated

I expected it to run evenly between the three but PID 3 was only scheduled once. This shows that even though the amount of tickets are the same the the PID picked is still random to a degree.

Test Case 3- have 2 procs, one with 10 tickets the other with 100. Scheduled 10 times

Text

Description automatically generated

As expected PID 2 was primary proc ran. I thought PID 1 would be ran at least once, but again this proves that the amount of tickets owned isn’t a guarantee but rather a chance multiplier.

Test Case 4- 4 procs 15, 30, 60 120 tickets held. Scheduled 21 times

Text

Description automatically generated

Text

Description automatically generated

PID 4 ran 10 out of 21 times. PID 2 and 3 both ran 5, and PID 1 ran once. As expected PID 4 ran double everything else and PID 1 ran the least

Test Case 5- 3 procs. 10, 11, 12 tickets respectively. Scheduled 15 times.

Text

Description automatically generated

Text

Description automatically generated

PID 2 ran 8 times PID 3 ran 5 and PID 1 ran 2. I expected PID 3 to run the most and PID 1 to run the least. PID 2 ran the most which is not surprising due to how close the number of tickets was.

LCFS Test 1

● Create a two proc scenario where each proc has a value of 0 (weight of 1024).

● Run to demonstrate the procs run equally. The two procs should alternate.

● Use ps command to show they each have the same timeslice with is half of the schedule

latency.

A picture containing graphical user interface

Description automatically generated

LCFS Test 2

● 3 procs: proc 1 nice -10, proc 2 nice 5, proc 3 nice 1

● Execute: timer 10 LCFS

Text

Description automatically generated

Graphical user interface

Description automatically generated with low confidence

LCFS Test 3

● 4 procs: proc 1 nice -1, proc 2 nice 3, proc 3 nice -3, proc 4 nice 1

● Execute: timer 15 LCFS

Text

Description automatically generated with low confidenceGraphical user interface, text

Description automatically generatedGraphical user interface

Description automatically generated with medium confidence

LCFS Test 4

● 4 procs: proc 1 nice 2, proc 2 nice -5, proc 3 nice 8, proc 4 nice 6

● Execute: timer 5 LCFSGraphical user interface

Description automatically generated with medium confidenceGraphical user interface

Description automatically generated with low confidence

LCFS Test 5

● 10 procs: proc 1 nice 0, proc 2 nice 1, proc 3 nice -1, proc 4 nice 0, proc 5 nice 1, proc 6 nice 1, proc 7 nice -1, proc 8 nice -2, proc 9 nice 0, proc 10 nice 1

● Execute: timer 25 LCFS

A picture containing table

Description automatically generated

1. On Canvas submit the code (.c, .h, and makefile) for your two scheduling algorithms. You must submit all code, even code that you did not modify. Your modified source files must include comments in the code that describes your modifications. If you implement two programs - one for each scheduler, you should submit two zip files - one for each solution.