ⓘ Students have either already taken or started taking this quiz, so take care when editing it. If you change any quiz questions in a significant way, you might want to consider re-grading students' quizzes who took the old version of the quiz.

Points 8 Published

Details Questions

Show question details

Question 1 pts

In lottery scheduling, CPU time allocation to processes cannot be guaranteed due to,

- Randomness
- Timesharing
- Context switching
- I/O operations

Question 1 pts

In Completely Fair Scheduler (CFS), if the time slice is large, it results in,

- Unfair scheduling
- Scheduling overhead
- Pure randomness
- High response time

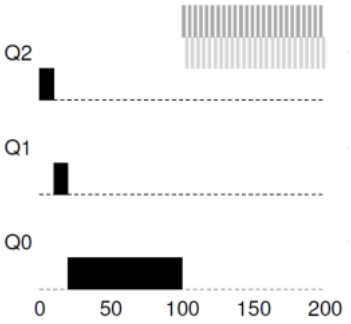
Question 1 pts

The Stride scheduler, each process has a "pass value" that increases in "strides" when a process is scheduled. The stride scheduler selects the process which has,

- The lowest pass value
- The highest pass value
- The highest stride value
- The lowest stride value

Question 1 pts

In multi-level feedback queue scheduler, a CPU intensive job is relegated to low priority, but an interactive job isn't. If there are too many interactive jobs, they can starve the CPU intensive job:



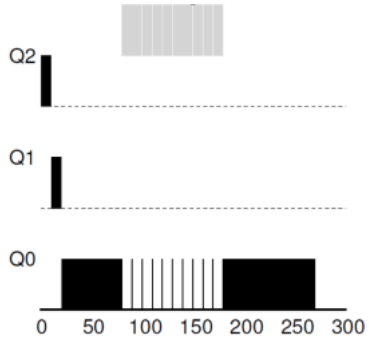
How does the scheduler solve this problem?

iswer

- Periodically boost the priority of all jobs
- Periodically schedule lower priority jobs
- Reduce the priority of interactive jobs if they are too many
- Do not allow so many interactive jobs to run at the same time

Question 1 pts

In multi-level feedback queue scheduling, a CPU intensive job is relegated to low priority, but an interactive job isn't. This can be used to "game" the system as follows:



How does the scheduler solve this problem?

iswer

- Reduce priority of a task based on total time used even if it was in many short bursts
- Forcefully reduce priority of processes that are trying to "game" the system
- Boost the priority of processes that being starved by the processes which are "gaming" the system
- Stop the processes which are trying to "game" the system

Question 1 pts

In Completely Fair Scheduler, suppose 10 processes are running and all have the same "nice" value.
Scheduling latency = 50ms
Minimum granularity = 6ms
What will be the time share of each process?

iswer

- 6ms
- 5ms
- 0.6ms
- 0.5ms

Question 1 pts

Which of the following is NOT a benefit of dynamic libraries?

iswer

- Lesser dependency on other libraries during run-time
- The executable files generated are smaller in size
- Updates in shared libraries are easy to use
- Low usage of the memory by the generated executable files

Question 1 pts

What is the the following compile command doing:

```
gcc -Wall -o a.out main.o libmylibrary.a
```

iswer

- link main.o with static library "mylibrary.a"
- link main.o with dynamic library "mylibrary.a"
- link main.o with static library "libmylibrary.a"
- link main.o with dynamic library "libmylibrary.a"

+ New question | + New question group | Find questions

Notify users this quiz has changed

Cancel | Save