Processes Scheduling

Exercises

1. \_\_\_\_ is the number of processes that are completed per time unit.

A) CPU utilization

B) Response time

C) Turnaround time

D) Throughput

2. \_\_\_\_\_\_\_\_\_\_is Linux scheduling algorithm designed especially for real-time systems.

A) SJF

B) SCHED\_OTHER

C) SCHED\_RR

D) Multilevel queue

3. Which of the following scheduling algorithms must be non-preemptive?

A) SJF

B) RR

C) FCFS

D) Preemptive priority

4. The default priority of Linux interruptable processes(TS) can be changed by changing

\_\_\_\_\_ value

A) share

B) active

C) nice

D) runqueue

5. \_\_\_\_\_\_ allows a thread to run on specific processor(s).

A) Processor affinity

B) Processor set

C) NUMA

D) Load balancing

6. What is the numeric priority of a Windows thread in the NORMAL\_PRIORITY\_CLASS with NORMAL relative priority?

A) 24

B) 10

C) 8

D) 13

7. What is the numeric priority of a Windows thread in the REAL-TIME\_CLASS with NORMAL relative priority?

A) 31

B) 24

C) 5

D) 8

8. A significant problem with priority scheduling algorithms is \_\_\_\_\_.

A) complexity

B) starvation

C) determining the length of the next CPU burst

D) determining the length of the time quantum

9. In Windows user threads can pre-empt kernel threads. T/F

10. Round-robin (RR) scheduling degenerates to first-come-first-served (FCFS) scheduling if the time quantum is too big. T/F

11. What role does the dispatcher play in CPU scheduling?

12. Explain the difference between throughput and turnaround time (scheduling criteria).

13. Suppose that the following processes arrive to ready queue at the times indicated. Each process will run for the amount of time listed.

Process Arrival Time (ms) Burst Time (ms) Priority

*P*1 0 5 3

*P*2 3 4 1

*P*3 1 3 1

1. What is the RR average turnaround time, if the quantum is 2ms?

14. Explain the scheduling algorithm implemented by Linux operating systems. CFS

**Threads- Exercises**

1) A program with a single thread of execution

A. cannot include recursive functions

B. does one thing at a time

2) The function pthread\_create returns

A. 0 if a new thread is created

B. 0 if the new thread is the child

3) All the threads in a process have access to

A. only static global variables

B. all global variables in the process

4) Race conditions arise when threads share variables because

A. the threads may run on different processors

B. of unpredictable scheduling of thread execution

5) When one thread calls exit

A. only that thread stops running

B. all threads in the process stop

6) Which of the following would be an acceptable signal handling scheme for a multithreaded program?

A) Deliver the signal to the thread to which the signal applies.

B) Deliver the signal to every thread in the process.

C) Deliver the signal to only certain threads in the process.

D) All of the above

7) Each thread has its own register set and stack. T/F

8). Provide examples in which multithreading provides better performance than a single-threaded solution.

**Answer:**

1. A Web server that services each request in a separate thread.

2. A parallelized application such as matrix multiplication where different parts of the matrix may be worked on in parallel.

3. An interactive GUI program where a thread is used to monitor

user input, another thread represents the running application, and a third thread monitors performance.

9). What resources are used when a thread is created? How do they differ from those used when a process is created?

**Answer:**

Because a thread is smaller than a process, thread creation typically uses fewer resources than process creation. Creating a process requires allocating a process control block (PCB), a rather large data structure. The PCB includes a memory map, list of open files, and environment variables. Allocating and managing the memory map is typically the most time-consuming activity. Creating either a user or kernel thread involves allocating a small data structure to hold a register set, stack, and priority.

10). List the benefits of multithreaded programming.

**Answer:**

The benefits of multithreaded programming fall into the categories: responsiveness, resource sharing, economy, and utilization of multiprocessor architectures. Responsiveness means that a multithreaded program can allow a program to run even if part of it is blocked. Resource sharing occurs when an application has several different threads of activity within the same address space. Threads share the resources of the process to which they belong to. As a result, it is more economical to create new threads than new processes. Finally, a single-threaded process can only execute on one processor regardless of the number of processors actually present.