**Operating System Concepts 10th Edition**

**Chapter 2 Exercises**

**2.13** Would it be possible for the user to develop a new command interpreter using the system-call interface provided by the operating system?

Yes, it would be possible.

**2.15** What are the two models of interprocess communication? What are the strengths and weaknesses of the two approaches?

The two models of interprocess communication are shared-memory model and the message-passing model.

shared-memory model: 1.strengths: low-overhead: a few system calls initially, and then none; more convenient for the user since we’re used to simply reading/writing from/to RAM. 2.weaknesses: more difficult to implement in the OS.

message-passing model: 1.strengths: useful for exchanging small amounts of data; simple to implement in the OS; sometimes cumbersome for the user as code is sprinkled with send/receive operations. 2.weakness: high-overhead: one system call per communication operation.

**Chapter 3 Exercises**

**3.12** Describe the actions taken by a kernel to context-switch between processes.

When a context switch occurs, the kernel saves the context of the old process in its PCB and loads the saved context of the new process scheduled to run.

补充：Describe the differences among short-term, medium-term, and long-term scheduling.

Long-term scheduling - job scheduling, select job from external storage to memory and create a process -----invoked very infrequently (seconds, minutes)  (may be slow)

Short-term scheduling - process scheduling, select the ready process to run on the processor-------- invoked very frequently (milliseconds)  (must be fast)

Medium-term scheduling - solves the problem of insufficient memory, using secondary storage to alleviate (controls the degree of multiprogramming)

**Chapter 4 Exercises**

**4.10** Which of the following components of program state are shared across threads in a multithreaded process?

a. Register values

b. Heap memory

c. Global variables

d. Stack memory

b、c

**4.11** Can a multithreaded solution using multiple user-level threads achieve better performance on a multiprocessor system than on a single-processor system? Explain.

A multithreaded system comprising of multiple user-level threads cannot make use of the different processors in a multiprocessor system simultaneously. The operating system sees only a single process and will not schedule the different threads of the process on separate processors. Consequently, there is no performance benefit associated with executing multiple user-level threads on a multiprocessor system

**4.13** Is it possible to have concurrency but not parallelism? Explain.

For a single core multitasking processor, there is only concurrency, not parallelism

**Chapter 5 Exercises**

**5.11** Of these two types of programs:

a. I/O-bound

b. CPU-bound

which is more likely to have voluntary context switches, and which is more likely to have nonvoluntary context switches? Explain your answer.

I/O-bound is more likely to have voluntary context switches, CPU-bound is more likely to have nonvoluntary context switches. A voluntary context switch occurs when a process has given up control of the CPU because it requires a resource that is currently unavailable (such as blocking for I/O.) A nonvoluntary context switch occurs when the CPU has been taken away from a process, such as when its time slice has expired or it has been preempted by a higher-priority process.

**5.12** Discuss how the following pairs of scheduling criteria conflict in certain settings.

a. CPU utilization and response time

b. Average turnaround time and maximum waiting time

c. I/O device utilization and CPU utilization

a: To improve CPU utilization, we should minimize the overhead of context switching, but at the same time, it will increase the response time of the process; Conversely, if you want to shorten the response time of the process, you need to increase context switching, which will reduce CPU utilization.

b: To reduce the average turnaround time, it is necessary to make the tasks with the shortest time execute first, but this will increase the waiting time of long-running tasks.

c: To improve CPU utilization, it is necessary to run CPU tasks for a long time, while to improve I / O utilization, I / O tasks need to be given priority.

**5.17** Consider the following set of processes, with the length of the CPU burst given in milliseconds:

Process Burst Time Priority

*P*1 5 4

*P*2 3 1

*P*3 1 2

*P*4 7 2

*P*5 4 3

The processes are assumed to have arrived in the order *P*1, *P*2, *P*3, *P*4, *P*5, all at time 0.

a. Draw four Gantt charts that illustrate the execution of these processes using the following scheduling algorithms: FCFS, SJF, nonpreemptive priority (a larger priority number implies a higher priority), and RR (quantum = 2).

b. What is the turnaround time of each process for each of the scheduling algorithms in part a?

c. What is the waiting time of each process for each of these scheduling algorithms?

d. Which of the algorithms results in the minimum average waiting time (over all processes)?

**5.18** The following processes are being scheduled using a preemptive, priority-based, round-robin scheduling algorithm.

Process Priority Burst Arrival

*P*1 8 15 0

*P*2 3 20 0

*P*3 4 20 20

*P*4 4 20 25

*P*5 5 5 45

*P*6 5 15 55

Each process is assigned a numerical priority, with a higher number indicating a higher relative priority. The scheduler will execute the highest priority process. For processes with the same priority, a round-robin scheduler will be used with a time quantum of 10 units. If a process is preempted by a higher-priority process, the preempted process is placed at the end of the queue.

a. Show the scheduling order of the processes using a Gantt chart.

b. What is the turnaround time for each process?

c. What is the waiting time for each process?

**5.20** Which of the following scheduling algorithms could result in starvation?

a. First-come, first-served

b. Shortest job first

c. Round robin

d. Priority

b、d

**5.22** Consider a system running ten I/O-bound tasks and one CPU-bound task. Assume that the I/O-bound tasks issue an I/O operation once for every millisecond of CPU computing and that each I/O operation takes 10 milliseconds to complete. Also assume that the context-switching overhead is 0.1 millisecond and that all processes are long-running tasks. Describe the CPU utilization for a round-robin scheduler when:

a. The time quantum is 1 millisecond

b. The time quantum is 10 milliseconds

**5.25** Explain the how the following scheduling algorithms discriminate either in favor of or against short processes:

a. FCFS

b. RR

c. Multilevel feedback queues

a: discriminates against short jobs since any short jobs arriving after long jobs will have a longer waiting time.

b: in favor of short processes since RR gives all jobs equal bursts of CPU time, short jobs will be able to leave the system faster.

c: in favor of short processes since short jobs will be able to leave the system faster.