Answers to Exercises

Chapter 1

Research Topics

Answers will vary. Good sources of information are IEEE Computer Journal, Communications of the ACM, and other computer journals and magazines that have articles comparing new operating systems. Be sure to emphasize the need for credible sources, especially on the Internet.

Exercises

1. Answers will vary and may include: UNIX, Linux, Macintosh, Windows Vista/XP/2000, VMS, MVS, MS-DOS, OS/390.

2. Memory management: how are data/programs loaded into memory.

Process management: how are programs executed.

File management: how are data/programs stored in secondary storage devices.

I/O management: how are I/O requests satisfied, how are I/O devices controlled.

Network management: how are on-line requests fulfilled, how are security issues handled, how is synchronization implemented.

The operating system is the manager of all resources in a computer system. It manages and protects resources in a multi-user environment. It allows for sharing of costly resources and information. It keeps track of who is using what resource, resolves conflicts among users, allocates and deallocates resources and charges users for its services.

3. As computer hardware became more reliable and was bought by increasingly larger numbers of customers, the separation between operators, programmers and users began to blur. Functions that were not separate in the days of first generation computers (vacuum tube technology), became very distinct during the second generation (transistor technology) and a great deal of computer time was wasted as operators walked from device to device loading decks of cards and tearing printouts. To reduce this waste of time operating systems were designed to process batches of programs without operator's intervention.

Memory hardware changed and made it possible to carry out multiprogramming, this necessitated changes in the operating system

During third generation computers (small scale integrated circuits technology) protection hardware made it possible to have time-sharing and the operating system had to be able to satisfy the needs of multiple users with on-line terminals.

Fourth generation computers (large scale integrated circuits and chips) signaled the beginning of the personal computer era. As operating systems for personal computers evolved, the user interface became more user friendly. During the late 1980s there was a surge of personal computers running network operating systems and distributed operating systems, which often allow programs to run on several processors at the same time, therefore requiring more complex scheduling algorithms than uniprocessor operating systems.

4. Opinion falls on both sides of this question so there is no wrong answer. Look for credible resources to back up the student's opinion. When Gordon Moore first described the trend, he was observing historical improvements in chip design. However, after the publication of Moore's Law, chip designers reportedly became driven to design chips to meet the expected prediction. Therefore, although it was initially based merely on one man's observations, it has since become a motivator for many chip designers. (Note: too that although it is commonly known as a "law," it is only a "theory" in strict scientific terminology.)

5. In a batch system programs to be run were collected in a designated area, read onto a magnetic tape using a small, relatively inexpensive computer. After collecting jobs in this manner for a prescribed period of time, the tape was rewound and mounted on a tape drive connected to the main computer where the programs were run one after the other. The output from each program was collected onto a second tape. When the entire batch was processed, the operator removed both tapes and passed the one with the outputs on to be printed by the same small computer connected to a printer. The operator then mounted another tape with another batch of programs to be run.

Interactive systems are a variation of multiprogramming systems because they allow several users to be on-line at the same time. The computer can provide fast interactive service to several users, and programmers can debug their programs in a shorter period of time than was possible with batch systems. This is because not all users are issuing commands to the computer at the same time. For example, of 300 users logged on there may be only 20 percent actively working at any given time allowing the CPU to handle their requests in a timely manner. In addition, some of the commands may take a small amount of CPU time to complete, for example to compile a student program, while others may take a longer time to finish, thus balancing the workload.

Real-time systems are used in time critical environments such as vehicle control or patient monitoring where failures in the system could lead to loss of life or major destruction of property. They are usually considered dedicated systems which spend most of the time on a single job. Work such as laboratory experiment monitoring, or environment control within buildings requires continuous processing, with little opportunity to use the computer for unrelated purposes.

6. Some examples of real-time applications: monitor the status of air defense systems, monitor and control the progress of experiments in laboratories, automate and control manufacturing processes in an industrial plant, vehicle control, environmental control, patient monitoring, nuclear system monitoring, air traffic control, robotics, telecommunications, stock exchange, emergency dispatch system, space craft landing, weather (tornado) tracking system

7. Answers will vary. The following examples show "batching" of data before executing an application, not batching of several applications. (1) Preparation of a weekly payroll for an organization: time cards are collected, data entered and processed, paychecks and reports are printed. (2) Preparation of report cards at end of term: grades are collected, data entered and processed, report cards are printed. (3) Customer billing done by utilities: meter readings are collected at end of month, data entered and processed, bills are printed.

Examples of batch processing, defined as collecting several programs to be run at one time, is not very common. However, some small educational institutions may still collect all student programs and run them at a given time because they are sharing that resource with the school's administrative functions which control the use of the computer equipment.

8. Any hardware component qualifies for inclusion in this list. Answers may include, CPU, monitor, keyboard, hard drives, removable storage media (such as flash memory, floppy disks, optical discs, etc.), scanners, cameras, and so on. Incorrect answers would include files, software, etc.

9. In active multiprogramming the operating system controls the amount of time that the CPU is allocated to each program already accepted for execution. When the preset time slice expires, the operating system issues an interrupt and allocates the CPU to another program. This insures a fairer utilization of resources and, in certain cases, faster program completion.

In passive multiprogramming the operating system had no control over the amount of CPU allocated to each program already accepted for execution. A program that required extensive processing could keep the processor for as long as it needed until it issued an I/O interrupt. This scenario did not optimize utilization of resources and, in some cases, would considerably delay completion of programs.

10. Although a single powerful computer could accomplish all of the processing output required of a bank (large or small), because it is a single machine it would be vulnerable to downtime should that one computer fail. Therefore, it would be wiser from a system reliability standpoint to have duplicate computing systems in case one should fail. (Availability and Reliability are discussed in chapter 12.)

11. For this answer, students should explore the ways that computers such as tablets, PDAs, navigation systems, etc. are used in everyday life to improve efficiency. Answers here can include collecting information from the Web (students should note specific types of data collected here), inventory management, changing assignments, source validation, etc. Encourage them to add detail to their answers -- more than merely: “web browsing or email.”

Advanced Exercises

12. Answers will vary.

13. 1. User issues command: BACKUP <path><filenames><drive>

2. User interface interprets command
   - awakens File Manager

3. File Manager resolves address of file to be backed up
   - awakens Device Manager

4. Device Manager opens access path to drive from where files are to be backed up
   - reads file (or portion of it) into I/O buffer
   - awakens Memory Manager

5. Memory Manager allocates memory for file in I/O buffer
   - awakens Processor Manager

6. Processor Manager loads file from I/O buffer to memory
   - awakens File Manager

7. File Manager resolves address of file to be copied to new disk
   - awakens Processor Manager

8. Processor Manager copies file from memory to I/O buffer
   - awakens Device Manager

9. Device Manager opens access path to drive where file will be copied
   - reads file from I/O buffer to new disk
   - awakens File Manager

10. File Manager determines if file is completely copied
    - if yes, writes Trailer Label
      - jump forward to step 11
    - if no, resolves address of next section of file to be backed up

      - awakens Device Manager returns to step 4

11. File Manager determines if this is the last file to be backed up
    - if no, awakens Processor Manager
      - Processor Manager activates user interface
        - sends *File N has been backed up* message to user
      - jump back to step 3

```
- if yes, awakens Processor Manager
    - Processor Manager activates user interface
        - sends *All files have been backed up*
          message to user
-   Continue with other functions
```

14. The clock rates and processors change technology advances. The faster processors more easily accommodate multiprocessing because they can quickly move data into and out of secondary storage. Therefore, the implications include increased speed, improved throughput, reduced overhead cost.

15. The only secure computing environment is one that is not networked and dedicated to only a single task or a single user. Among the security issues that arise in a shared processing environment are:

- inadvertent or intentional access to user ids and passwords
- unauthorized access to private data
- unauthorized access to, or alteration of, the operating system software
- inadvertent or intentional spreading of viruses and other destructive programming

16. Answers will vary but two examples are web browsing and database searching.

17. Once the process terminates, the threads will terminate because control of each thread is held by the parent process.

18. Because control of threads is held by the process, the threads do not have additional powers not granted to the process. Therefore, if the process is put on wait, then the threads are also waiting.

Chapter 2

Research Topics

Answers will vary. Look for credible references and evidence that the student thoughtfully evaluated the available research.

Exercises

1 a. In a multiprogramming environment several programs reside simultaneously in main memory and share the CPU. Multiprogramming is used to keep the processor busy: time is saved by having a program ready to execute as soon as another needs to wait. It allows more efficient utilization of system resources and improves system throughput.

b. Internal fragmentation occurs when the amount of main memory requested by a program is less than the size of the fixed partition allocated to it. The unused space within the partition is the fragment of memory that is wasted because it cannot be allocated to another program.

c. External fragmentation is the development of unusable portions of memory in a dynamic partition memory management scheme. As programs terminate they release their allocated section of main memory which is promptly allocated to other programs ready to execute. If the size of the new program is smaller than that of the dynamic partition, then a hole develops between two allocated memory partitions. This hole is the external fragment.

d. Compaction is used to consolidate all the external fragments (free areas in memory) into one contiguous block. In some cases this enhances throughput by allowing more programs to be active at the same time.

e. Relocation is the process by which programs are repositioned in main memory to allow compaction of free memory areas. When relocation takes place, the addresses specified by a program for either branching or data reference are modified, during execution, to allow the program to execute correctly. Relocation can be time consuming, therefore it should be done sparingly. Some suggestions are: (1) when a certain percentage of main memory is used up, (2) when the number of programs waiting for execution reaches a prescribed upper limit, (3) when a prescribed amount of time has elapsed, (4) combinations of 1, 2 and 3.

2. Major disadvantages for all: (1) the program must be entirely loaded into memory. (2) it must be stored contiguously and (3) it must remain in memory until completion. In addition:

Single-user systems:	Often large sections of memory are wasted. CPU is idle during I/O. Peripheral devices are wasted when program is using CPU.
Fixed partitions:	Internal fragmentation.
Dynamic partitions:	External fragmentation.

Relocatable dynamic partitions: Time wasted in relocation. Additional hardware support needed. Even after compaction, main memory may not be big enough to accommodate incoming jobs and goes to waste.

3. Major advantages:

Single-user systems:	Good for its simplicity. Memory management is easy to maintain and understand. Suitable for small personal computers where simplicity is the goal.
Fixed partitions:	Allows for multiprogramming.
Dynamic partitions:	Allows for multiprogramming. Removes internal fragmentation.

Relocatable dynamic partitions: Allows for multiprogramming. Removes external fragmentation. May improve throughput.

4.

Job List	Memory List
J1 740K	B1 610K
J2 500K	B2 850K
J3 700K	B3 700K

Best Fit:	J1 into B2	850 - 740 = 110 internal fragmentation.
	J2 into B1	610 - 500 = 110 internal fragmentation.
	J3 into B3	700 - 700 = 0 internal fragmentation.
First Fit:	J1 into B2	850 - 740 = 110 internal fragmentation.
	J2 into B1	610 - 500 = 110 internal fragmentation.
	J3 into B3	700 - 700 = 0 internal fragmentation.

5.

Job List		Memory List	
J1	700K	B1	610K
J2	500K	B2	850K
J3	740K	B3	700K

Best Fit: J1 into B3 700 - 700 = 0 internal fragmentation.
J2 into B1 610 - 500 = 110 internal fragmentation.
J3 into B2 850 - 740 = 110internal fragmentation.

First Fit: J1 into B2 850 - 700 = 150 internal fragmentation.
J2 into B1 610- 500 = 110 internal fragmentation.
J3 has to wait because it does not fit into B3.

6 a. Memory is used more evenly because the search for a free partition does not always start at the beginning of the list (forcing higher use of these partitions and lower use of the partitions at the end of the list).

b. Next Fit compared to exercise 4.

J1 into B2 850 - 740 = 110 internal fragmentation.
J2 into B3 700 - 500 = 200 internal fragmentation.
J3 has to wait because it does not fit into B1.

It is worse that either Best Fit or First Fit because J3 has to wait.

c. Next Fit compared to exercise 5.

J1 into B2 850 - 700 = 150 internal fragmentation.
J2 into B3 700 - 500 = 200 internal fragmentation.
J3 has to wait because it does not fit into B1.

It is worse than Best Fit because J3 has to wait. It is worse than First Fit because there is more internal fragmentation (350 is greater than 260).

7 a. If the internal fragments cannot be allocated to other programs, then there are no advantages to this scheme. However, it usually produces internal fragments that are likely to be large enough to be useful if the policy of the Memory Manager is to release internal fragments to incoming jobs (this is not the policy treated in the chapter).

b. Worst Fit compared to exercise 4.

J1 into B2 850 - 740 = 110 internal fragmentation.
J2 into B3 700 - 500 = 200 internal fragmentation.
J3 has to wait because it does not fit into B1.

It is worse than either Best Fit or First Fit because J3 has to wait.

c. Worst Fit compared to exercise 5.

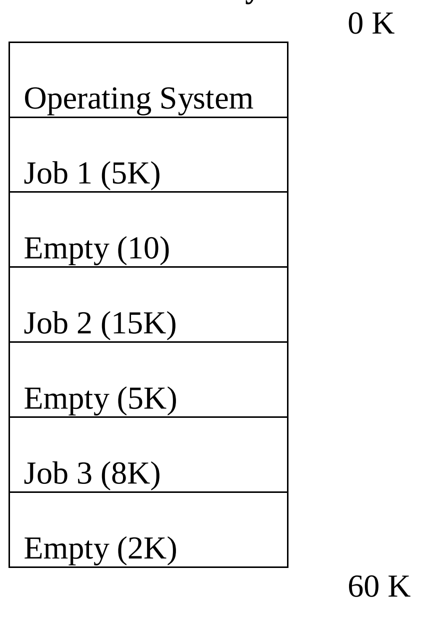
J1 into B2 850 - 700 = 150 internal fragmentation.
J2 into B3 700 - 500 = 200 internal fragmentation.
J3 has to wait because it does not fit into B1.

It is worse than Best Fit because J3 has to wait. It is worse than First Fit because there is more internal fragmentation (350 is greater than 260).

8. This question can be answered using the following pseudocode and example which show that all free sections of memory can be merged into one contiguous block without using secondary storage as a temporary holding area. It could generate a discussion on additional hardware components needed to implement this method of compaction.

Example:

Main Memory



Job 4 requires 15K. Compaction is needed.

```
Pseudocode:
compaction-case = 0
k = 1
DO-WHILE there are free-partitions
      j = 1
    DO-WHILE there are active jobs
      IF free-partition-size(k) = active-job-size(j)
          THEN compaction-case = 1
              relocate active job(j) into free-partition(k)
          ELSE
             IF free-partition-size(k) > active job-size(j)
                THEN compaction-case = 2
                   relocate active-job(j) into free-partition(k)
                  free-partition-size (k) = free-partition-size (k) - active-job-size(j)
             ELSE compaction-case = 3
                   difference = active job-size(j) - free-partition-size(k)
                   beginning = 0
                  DO-WHILE beginning not = active job-size(j)
                       ending = beginning + free-partition-size(k)
                       relocate active job(j) from beginning to ending
                       release active job(j) from beginning to ending
                      beginning = ending
                      difference = difference - free-partition-size(k)
                      IF difference < 0
                         THEN difference = difference - free-partition-size(k)
                              free-partition-size(k) = difference
                     END-IF
                 END-DO-WHILE
              END-IF
           END-IF
           j = j + 1
       END-DO-WHILE
           k = k + l
     END-DO-WHILE
     IF compaction-case = 0
        THEN "Compaction is not possible"
     ELSE
          perform update of free and busy lists
     END-IF
```

9. This type of compaction may prove to be more time consuming than the previous one because it requires access to a secondary storage device. If the operating system and computer are equipped with "block" transfer and buffers (these are discussed later in the text), then the transfer time could be optimized. In addition, if the disk was a device dedicated only to compaction, then its hardware components could also be set to optimize the seek time. When using secondary storage one needs to remember that a store and load operation are performed every time increasing the time needed to complete compaction.

Advanced Exercises

10 a. Job 4 cannot be accommodated because there is not enough contiguous free memory available.

b.

Contents of Relocation Registers	Job Number

0	1
-20480 (-20K)	2
-30720 (-30K)	3

c. The relocation register for Job 4 is set to zero because it has just been loaded into memory.
d. Its location has not changed because Job 1 has not been relocated.
e. Its location is: 55K - 20K = 35K or 35840
f. Its location is: 80K - 30K = 50K or 51200
g. Its location is: 110K - 30K = 80K or 81920
h. That value does not change. The **true** address is computed when execution occurs.

11. Although the following is not an exhaustive list, it is representative of answers provided by students.

a. Is the environment still static? Is the workload still predictable? Are the characteristics of the jobs still the same?

b. Collect data as jobs are being submitted. What are typical memory requests? Are there jobs whose memory requests exceed the largest partition available? Keep track of the memory requests of jobs that have to wait. Keep track of internal fragmentation as waiting queues develop. Keep track of partition utilization. Are there some which are never used?

c. Examine new environment requirements. Is it still static but memory requirements have changed? Then reconfigure memory partitions to accommodate these changes. Has it become more dynamic? Then it might be better to go into a dynamic memory allocation scheme.

Programming Exercises

Exercises **12**, **13** and **14** are best explained during a few class sessions using a sample paper-and-pencil model to highlight the structure of the program modules and their interrelationships. Sample outputs are very helpful to the students as well as a sample of the analysis that can be performed on the data collected.

Typically the procedures used in the simulation are:

i. Main Procedure: contains the calls to all other procedures.
ii. Input Procedure: where all data is inputted.
iii. Time Procedure: where the system clock is updated to indicate how long it took to process all jobs and where the time to run a job (job clock) is updated.
iv. Waiting Procedure: used if a job is put on a waiting queue. The job's waiting clock is updated and the waiting queue count is increased.
v. Snapshot Procedure: used to print out the status of the job table and the memory table at even intervals. This gives a good trace of how the system is behaving.
vi. Winding Procedure: used to handle all jobs still active or waiting after there are no more incoming jobs.
vii. Statistics Procedure: used to produce the typical measures requested by the exercise.

A policy has to be developed to handle the conflict of which job will be served first: the one in the waiting queue or the incoming one. A related policy must be defined to determine whether no more incoming jobs will be accepted until the ones in the waiting queue are served or the system will flip-flop between incoming and waiting jobs.

The data for the jobs is best stored into a table with columns indicating: job size, job time to run, waiting time and status (new, running, waiting, too big or done). The data for the memory blocks is best stored into a table with columns indicating: block size, status (busy or free), number of jobs currently using block, counter for number of times block has been used.

Sample Snapshot:

Job Table

Job Number	Run Time	Job Size	Job Status	Wait Time	Completion Time
01	05	576	done	00	05
02	04	419	done	00	04
03	08	329	done	00	08
04	02	203	done	00	02
05	02	255	done	00	02
06	06	699	done	00	06
07	08	894	running	00	
08	10	074	running	00	
09	07	393	running	00	
10	06	689	running	01	
11	05	658	waiting	03	
12	08	382	running	00	
13	09	914	new		
14	10	042	new		
15	10	022	new		
16	07	754	new		
17	03	321	new		
18	01	138	new		
19	09	985	new		
20	03	361	new		
21	07	754	new		
22	02	271	new		
23	08	839	new		
24	05	595	new		
25	10	076	new		

Memory Table

Block Number	Block Size	Block Status	Job Number	Number of Times Used	Internal Fragmentation
01	950	busy	10	3	261
02	700	busy	08	2	626
03	450	busy	12	2	068
04	850	busy	9	2	457
05	300	free		1	
06	900	busy	07	1	006
07	100	free			
08	550	free			
09	150	free			
10	050	free			