Exercises, Chap 2

Research Topics (Answers here will vary. Look for originality in the student's work.)

A. Research the maximum and minimum memory capacity of two models of laptop computers. Be sure not to confuse the computer's memory capacity with its secondary storage. Explain why memory capacity and secondary storage are often confused. Cite the sources of your research.

B. Identify an example of an online data source that's used by researchers to advance their profession. Consider interstellar databases, medical data sources, and so on. Explain what kinds of data are included in the online source, the researchers who access it, and scientific advances that may have stemmed from such a resource. Be sure to cite your sources.

Exercises

1. Consider the first and last memory allocation scheme described in this chapter. Describe their respective advantages and disadvantages. ANSWER:

	Single User Contiguous	Relocatable Dynamic Partitions
Advantages	• Requires minimal overhead • No external fragmentation	• Allows multiple jobs in memory at one time • Better use of the CPU • Improved use of memory space • Resolved external

		fragmentation
Disadvantages	• Does not allow multiple jobs in memory at one time • Poor use of the CPU resources	• Requires significant overhead for memory compaction and job management

2. In your own words, how often should memory compaction/relocation be performed? Describe the advantages and disadvantages of performing it even more often than recommended. (Answer)

We hope every student will have a unique answer to this question, which depends on the cost of the overhead operation in terms of processing time and use of system resources required for relocation (such as disk space). The question is designed to encourage students to speculate on the cost to throughput and system resources of relocation and compaction. Certainly, if those costs are not taken into consideration, some students might suggest that relocation be performed very, very often.

3. Give an example of computing circumstances that would favor first-fit allocation over best-fit. Explain your answer. (Answer: Students should conjure unique examples that emphasize data writing speed over data retrieval speed. Such circumstance would include data archival and massive storage of data that will seldom be retrieved. This question is designed to explore current technologies as they relate to speedy storage of data elements.)

4. Given the following: configuration with jobs arriving in order (Job A, B, C) and with blocks shown in order from low order memory to high order memory:

[Start UNT-p.53-1 here] Editor: these should appear as 2 tables

Job List:	
Job Number	**Memory Requested**
Job A	990K
Job B	275K
Job C	760K

Memory Block List:	
Memory Block	**Memory Block Size**
Block 1	900K
Block 2	960K
Block 3	300K

[End UNT-p.53-1 here]

a. Use the first-fit algorithm to indicate which memory blocks are allocated to each of the arriving jobs. ANSWER: Block 1 gets Job B, Block 2 gets Job C. Job A is not processed because it is too large for any of the memory blocks.

b. Use the best-fit algorithm to indicate which memory blocks are allocated to each of the arriving jobs. ANSWER: Block 3 gets Job C, Block 1 gets Job C. Job A is not processed because it is too large for any of the memory blocks.

c. Calculate the amount of internal fragmentation in Block 1 using the first-fit algorithm. ANSWER: Using first-fit, Block 1 (900K) is allocated to Job B (275K) so the fragmentation is 900 – 275 = 625K.

5. Given the following configuration with jobs arriving in order (Job A, B, C, D) and with blocks shown in order from low order memory to high order memory:

[Start UNT-p.53-2 here] Editor: these should appear as 2 tables

Job List:	
Job Number	**Memory Requested**
Job A	256K
Job B	900K
Job C	50K
Job D	350K

Memory Block List:	
Memory Block	**Memory Block Size**
Block 1	910K
Block 2	900K
Block 3	200K
Block 4	300K

[End UNT-p.53-2 here]

a. Use the best-fit algorithm to indicate which memory blocks are allocated to each of the arriving jobs.

ANSWER:

Job A (256K) gets Block 4 (300K)

Job B (900K) gets Block 2 (900K)

Job C (50K) gets Block 3 (200K)

Job D (350K) gets Block 1 (910K)

b. Use the first-fit algorithm to indicate which memory blocks are allocated to each of the arriving jobs.

ANSWER:

Job A (256K) gets Block 1, Job B (900K) gets Block 2.

Job C (50K) gets Block 3.

Job D is not processed immediately because it is too large for the last available block (which is Block 4). However, it will be processed after either Block 1 or Block 2 become available and is allocated to Job D, which is waiting.

c. Calculate the total amount of internal fragmentation in all four blocks using the best-fit algorithm.

ANSWER:

- -Block 4 (with Job A) has 44K fragmentation.
- -Block 2 (with Job B) has zero fragmentation because the job and block are the same size, 900K.
- -Block 3 (with Job C) has 150K in fragmentation.
- -Block 1 (with Job D) has 560K in fragmentation.
- -Total fragmentation for all four blocks (44+0+150+560) is 754K.

6. Next-fit is an allocation algorithm that starts by using the first-fit algorithm but it keeps track of the partition that was last allocated so that it can start its next search from that exact spot instead of restarting the search with Block 1. In other words, it starts searching from the most recently allocated block when the next job arrives. Using the following configuration with jobs arriving in order (Job A, B, C, D) and with blocks shown in order from low order memory to high order memory:

[Start UNT-p.54-1 here] Editor: these should appear as 2 side-by-side tables

Job List:	
Job Number	**Memory Requested**
Job A	625K
Job B	44K
Job C	900K
Job D	220K

Memory Block List:	
Memory Block	**Memory Block Size**
Block 1	250K
Block 2	900K
Block 3	280K
Block 4	200K
Block 5	50K

[End UNT-p.54-1 here]

(a) Indicate which memory blocks are allocated to each of the arriving jobs. ANSWER:

-Job A (625K) doesn't fit into Block 1 and so gets Block 2 (900K).

-When Job B (44K) arrives, it does not get Block 1 (250K) because the next-fit system chooses the next block after the one most recently allocated, which is Block 2. Therefore, Job B gets Block 3 (280K).

-Job C cannot be allocated and so has to wait because it is very large and the only block that's large enough (Block 2) has already been allocated to Job A. At this point, the last allocated job is Block 3 so the next incoming job,

-Job D, is allocated to the next block that's large enough to hold it, Block 1 (250K).

-Later, when Job A is done, Block 2 can be allocated to the waiting Job C.

(b) Explain in your own words what advantages the next-fit algorithm could offer.

ANSWER: This allocation scheme is designed to make processing time more efficient but does not necessarily reduce fragmentation. It would also allocate the blocks at the bottom of the list more often that first-fit or best-fit, both of which begin searching for available blocks from the first block.

7. Worst-fit is an allocation algorithm that allocates the largest free block to a new job. It is the opposite of the best-fit algorithm. Using the following configuration with jobs arriving in order (Job A, B, C, D) and with blocks shown in order from low order memory to high order memory:

[Start UNT-p.54-1 here] Editor: these should appear as 2 side-by-side tables

Job List:	
Job Number	**Memory Requested**
Job A	44K
Job B	220K
Job C	670K
Job D	64K

Memory Block List:	
Memory Block	**Memory Block Size**
Block 1	250K
Block 2	900K
Block 3	280K
Block 4	200K
Block 5	750K

[End UNT-p.54-1 here]

a) Indicate which memory blocks are allocated to each of the arriving jobs.

ANSWER:

-Job A (44K) will be allocated to the biggest available block, Block 2 (900K).

- Then Job B (220K) will get the biggest available block, Block 5 (750K).

-Job C (670K) will need to wait until a block that's large enough (Blocks 2 or 5) becomes available.

-Job D (64K) will be allocated Block 3 because it's the largest one available of the five blocks.

b) In this chapter we introduced external and internal fragmentation. Explain in your own words which one would be expected in a worst-fit memory allocation scheme

ANSWER: The worst-fit memory allocation scheme lends itself to internal fragmentation because the blocks allocated to the incoming jobs can be much larger than necessary, causing wasted space within the block (internal fragmentation) and not between blocks (external fragmentation).

8. Imagine an operating system that cannot perform memory deallocation. Name at least three effects on overall system performance that might result and explain your answer. (Answer)

Answers here will vary but could include the following. This question could invite a discussion of the problem of "memory leaks" where an application releases some but not all allocated memory space (this is a subject not discussed in this text).

- It would run out of available memory as soon as each available memory location was used once. Memory could never be reused by another program.
- If the only way to clear memory was to reboot, then the system would need to be rebooted often.
- Such a system would need vast memory reserves to work.
- Such a system would encourage the development of applications that needed very small memory resources.

9. In a system using the relocatable dynamic partitions scheme, given the following situation (and using decimal form): Job Q is loaded into memory starting at memory location 42K.

a. Calculate the exact starting address for Job Q in bytes. Answer: 43008 (42*1024 = 43008)

b. If the memory block has 3K in fragmentation, calculate the size of the memory block. Answer: 46080 (3*1024 = 3072) + (42*1024 = 43008) = 46080

c. Is the resulting fragmentation internal or external? Explain your reasoning. Answer: Look for a fundamental understanding of the differences between internal and external fragmentation.

10. In a system using the relocatable dynamic partitions scheme, given the following situation (and using decimal form): Job W is loaded into memory starting at memory location 5000K.

a. Calculate the exact starting address for Job W in bytes. Answer: 5120000 (5000*1024 = 5120000)

b. If the memory block has 3K in fragmentation, calculate the size of the memory block.

Answer: 5130240 (10*1024 = 10240) + (5000*1024 = 5120000) = 5130240

c. Is the resulting fragmentation internal or external? Explain your reasoning. Answer: Look for a fundamental understanding of the differences between internal and external fragmentation.

11. Using your own words, explain the role of the bounds register.

Answer: This hardware element, the bounds register, tracks the limits of addressable memory for this job and prevents it from trying to access memory that's allocated to another job.

12. Describe in your own words the role of the relocation register.

Answer: This hardware element, the relocation register, tracks the number of bytes that a job has been moved in main memory as a result of memory compaction. It is used to track each instruction so it can be found successfully after the job's relocation

13. If the relocation register holds the value -83968, was the relocated job moved toward lower or higher addressable end of main memory? Answer: It was moved toward the lower addressable end of memory) By how many kilobytes was it moved? Answer: It was moved 82K (-83968 / 1024 = -82) Explain your conclusion. Students should explain how they reached their conclusion by showing their work.

14. In a system using the fixed partitions memory allocation scheme, given the following situation (and using decimal form): After Job J is loaded into a partition of size 50K, the resulting fragmentation is 7168 bytes. Perform the following:

    a. What is the size of Job J in bytes? Answer: Job J is 44032 bytes (50*1024 = 51200) – 7168 = 44032 bytes

    b. What type of fragmentation is caused? Answer: Internal fragmentation

15. In a system using the dynamic partition memory allocation scheme, given the following situation (and using decimal form): After Job C of size 70K is loaded into a partition resulting in 7K of fragmentation, calculate the size (in bytes) of its partition (Answer: 71680 (70*1024 = 71680) and identify the type of fragmentation that is caused (External fragmentation). Explain your answer. (Students should show their calculations and explain how they know that the fragmentation is *between* partitions indicating external fragmentation and not within partitions, which would indicate internal fragmentation.)

Advanced Exercises

16. The relocation example presented in the chapter implies that compaction is done entirely in memory, without secondary storage. Can all free sections of memory be merged into one contiguous block using this approach? Why or why not?

(see attached document for answer to 2-16)

17. In this chapter we described one of several ways to compact memory. Some people suggest an alternate method: all jobs could be copied to a secondary storage device and then reloaded (and relocated) contiguously into main memory, thus creating a single free block after all jobs have been recopied into memory. Is this viable? Could you devise a better way to compact memory? Write your algorithm and explain why it is better. (Answer: 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.

18. Given the memory configuration in Figure 2.10 below, answer the following questions given that at this point, Job 4 arrives requesting a block of 100K.

    a. Can Job 4 (100K) be accommodated? Why or why not? ANSWER: Job 4 cannot be accommodated because there is not enough contiguous free memory available.

    b. If relocation is used, after memory compaction, what are the contents of the relocation registers for Job 1, Job 2, and Job 3? ANSWER:

Contents of Relocation Registers	Job Number
0	1
-20480 (-20K)	2
-30720 (-30K)	3

c. What are the contents of the relocation register for Job 4 after it has been loaded into memory? Answer: The relocation register for Job 4 is set to zero because it has just been loaded into memory.

d. An instruction that is part of Job 1 was originally loaded into memory location 22K. What is its new location after compaction? Answer Its location has not changed because Job 1 has not been relocated.

e. An instruction that is part of Job 2 was originally loaded into memory location 55K. What is its new location after compaction? Answer Its location is: 55K - 20K = 35K (or 35840)

f. An instruction that is part of Job 3 was originally loaded into memory location 80K. What is its new location after compaction? Answer Its location is: 80K - 30K = 50K (or 51200)

g. If an instruction was originally loaded into memory location 110K, what is its new location after compaction? Answer: Its location is: 110K - 30K = 80K (or 81920)

Programming Exercises

19. Here is a long-term programming project. Use the information that follows to complete this exercise.

[Start UNT-p.56-1 here]

Job List		
Job Stream Number	**Time**	**Job Size**
1	5	5760

2	4	4190
3	8	3290
4	2	2030
5	2	2550
6	6	6990
7	8	8940
8	10	740
9	7	3930
10	6	6890
11	5	6580
12	8	3820
13	9	9140
14	10	420
15	10	220
16	7	7540
17	3	3210
18	1	1380
19	9	9850
20	3	3610

21	7	7540
22	2	2710
23	8	8390
24	5	5950
25	10	760

[End UNT-p.56-1 here]

[Start UNT-p.56-2 here]

Memory List	
Memory Block	**Size**
1	9500
2	7000
3	4500
4	8500
5	3000
6	9000
7	1000
8	5500
9	1500
10	500

[End UNT-p.56-2 here]

At one large batch-processing computer installation, the management wants to decide what storage placement strategy will yield the best possible performance. The installation runs a large real storage computer (as opposed to "virtual" storage, which is covered in Chapter 3) under fixed partition multiprogramming. Each user program runs in a single group of contiguous storage locations. Users state their storage requirements and time units for CPU usage on their Job Control Card (it used to, and still does, work this way, although cards may not be used). The operating system allocates to each user the appropriate partition and starts up the user's job. The job remains in memory until completion. A total of 50,000 memory locations are available, divided into blocks as indicated in the previous table.

a. Write (or calculate) an event-driven simulation to help you decide which storage placement strategy should be used at this installation. Your program would use the job stream and memory partitioning as indicated previously. Run the program until all jobs have been executed with the memory as is (in order by address). This will give you the first-fit type performance results.

b. Sort the memory partitions by size and run the program a second time; this will give you the best-fit performance results. For both parts a. and b., you are investigating the performance of the system using a typical job stream by measuring:

    1. Throughput (how many jobs are processed per given time unit)
    2. Storage utilization (percentage of partitions never used, percentage of partitions heavily used, and so on)
    3. Waiting queue length
    4. Waiting time in queue

5. Internal fragmentation

Given that jobs are served on a first-come, first-served basis:

c. Explain how the system handles conflicts when jobs are put into a waiting queue and there are still jobs entering the system—which job goes first?

d. Explain how the system handles the "job clocks," which keep track of the amount of time each job has run, and the "wait clocks," which keep track of how long each job in the waiting queue has to wait.

e. Since this is an event-driven system, explain how you define "event" and what happens in your system when the event occurs.

f. Look at the results from the best-fit run and compare them with the results from the first-fit run. Explain what the results indicate about the performance of the system for this job mix and memory organization. Is one method of partitioning better than the other? Why or why not? Could you recommend one method over the other given your sample run? Would this hold in all cases? Write some conclusions and recommendations.

20. Suppose your system (as explained in Exercise 19) now has a "spooler" (a storage area in which to temporarily hold jobs), and the job scheduler can choose which will be served from among 25 resident jobs. Suppose also that the first-come, first-served policy is replaced with a "faster-job, first-served" policy. This would require that a sort by time be performed on the job list before running the program. Does this make a difference in the results? Does it make a difference in your analysis? Does it make a difference in your conclusions and recommendations? The program should be run twice to test this new policy with both best-fit and first-fit.

21. Suppose your spooler (as described in the previous exercise) replaces the previous policy with one of "smallest-job, first-served." This would require that a sort by job size be performed on the job list before running the program. How do the results compare to the previous two sets of results? Will your analysis change? Will your conclusions change? The program should be run twice to test this new policy with both best-fit and first-fit.

Answers to 19, 20, 21: The three Programming Exercises 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 submitted.

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	2		457
05	300	free		1	
06	900	busy	07	1	006
07	100	free			
08	550	free			
09	150	free			
10	050	free			