

Spencer Shimko  
CMSC 421  
HW 3

**1. a) Given memory partitions of 100KB, 500KB, 200KB, 300KB and 600KB (in order), how would each of the First-fit, best-fit, and worst-fit algorithms place processes of 212KB, 417KB, 112KB, 426KB and 50KB (in order)? Which algorithm makes the most efficient use of memory?**

Note: any process for which space is not available will be placed in a “wait” state for the solutions here.

- First-fit: 212KB->500KB; 417KB->600KB; 112KB->288KB (288KB fragment left after partial fit of 212KB into 500KB); 426KB->wait (no partition or fragment of that size available); 50KB->100KB
- best-fit: 212KB->300KB; 417KB->500KB; 112KB->200KB; 426KB->600KB; 50KB->100KB
- worst-fit: 212KB->600KB; 417KB->500KB; 112KB->388KB (388KB fragment left after partial fit of 212KB into 600KB); 426KB->wait (no partition or fragment sufficient); 50KB->300KB

Best fit “wastes” the smallest amount of memory (most efficient use of memory) and is the only method capable of meeting all memory requests. However its overall performance is poor because it must scan all blocks and compaction must be performed often.

**b) Exercise 9.8 on page 314 of the text: Consider a logical address space of eight pages of 1024 words each, mapped onto a physical memory of 32 frames. (a) How many bits are there in the logical address? (b) How many bits are there in the physical address?**

- a)  $8\text{pages} * 1024\text{words (offset)} = 8192 = 2^{13}$   
13 bits to represent the logical address
- b)  $32\text{frames} * 1024\text{words (offset)} = 32,768 = 2^{15}$   
15 bits to represent the physical address

**2. Consider the following page-reference string:**

**1,1,2,3,4,3,4,2,1,5,1,2,3,4,5,6,6,1,6,3,2,1,6,3,4,6.**

**How many page faults would occur for the following replacement algorithms, assuming one, three, five, six or eight frames? Remember that all the frames are initially empty, so you first unique pages will all cost one fault each.**

	<i>1</i>	<i>3</i>	<i>5</i>	<i>6</i>	<i>8</i>
LRU	24	17	9	6	6
FIFO	24	16	11	6	6
Optimal	24	13	6	6	6

### 3. Problem 12.6 (page 450) from the text.

Consider a file system on a disk that has both logical and physical block sizes of 512 bytes. Assume that the information about each files i already in memory. For each of the three allocation strategies (contiguous, linked, and indexed), answer these questions:

a) How is the logical-to-physical address mapping accomplished in this system? (For the indexed allocation, assume the file is always less then 512 bytes long).

b) If we are currently at logical block 10 (the last block accessed was 10) and want to access logical block 4, how many physical blocks must be read from disk?

Note: For answering part “a”, I assume (L and S are defined):

L = the logically address to be mapped

S = the initial block number

B = block

O = offset

#### Contiguous:

a)  $L / 512 = B \text{ r } O$  (quotient=B, remainder=O)

$S + B =$  physical block number

O is the offset into that block

So:  $S + B + O =$  physical address

b) 1 (direct block access for contiguous storage)

#### Linked:

a)  $L / 511 = B \text{ r } O$  (quotient=B, remainder=O)

Follow the linked list for (B + 1) blocks. (O + 1) should be the offset into the last physical block.

b) 4

**Indexed:**

a)  $L / 512 = B \text{ r } O$  (quotient=B, remainder=O)

-Load the index block from disk (if not already in memory).

-The physical address is stored at in the index block B.

-O is the offset into the physical block pointed to by index block B.

b) 2

**4. Problem 13.2 (pages 487-488) from the text.**

Professor Joshi informed our section that this problem did NOT need to be completed and would not be graded.