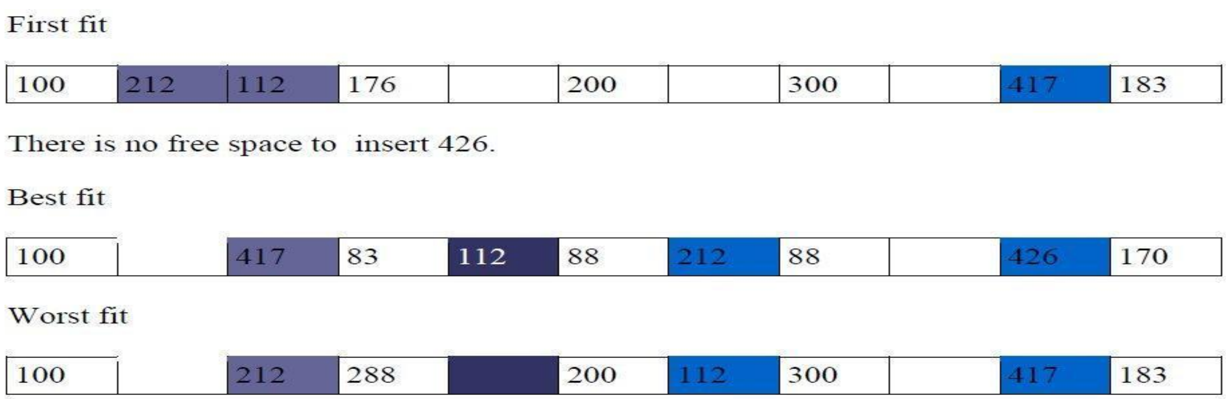
1**. Given five memory partitions of 100Kb, 500Kb, 200Kb, 300Kb, 600Kb (in order), how would the first-fit, best-fit, and worst-fit algorithms place processes of 212 Kb, 417 Kb, 112 Kb, and 426 Kb (in order)? Which algorithm makes the most efficient use of memory?**

First-fit:
212K is put in 500K partition
417K is put in 600K partition
112K is put in 288K partition (new partition 288K = 500K - 212K)
426K must wait

Best-fit:
212K is put in 300K partition
417K is put in 500K partition
112K is put in 200K partition
426K is put in 600K partition

Worst-fit:
212K is put in 600K partition
417K is put in 500K partition
112K is put in 388K partition
426K must wait

OR



There is no free space to insert 426.

In this example, best-fit turns out to be the best.

2. Assuming a 4 KB page size, what are the page numbers and offsets for the following address references (provided as decimal numbers):
a. 2375
b. 19366
c. 30000
d. 256
e. 16385

Assumption:
- (max) Physical/logical Address Space = 32,768 because part c has the highest address 30,000,

which would fit in an addressible space of this size. This would require a 15 bit address size.

- A Physical/logical address of 32,786 would have 'm' = 15 (2^15) => 32,768

Page size: 2^n = 4KB = 4,096 bytes

n = 12

Page count = 8, bits required: 3, 0 indexed: 0-7

Page Number   Page Offset

| 3 bits | 12 bits |

(m-n)   n

a. 3275 => (binary) 110011001011

000110011001011 (zero padded to 15 bits)

Page: (binary) 000 -> (decimal) 0 [the first page]

Offset: (binary) 110011001011 -> (decimal) 3,275

b. 19366 => (binary) 100101110100110

Page: (binary) 100 -> (decimal) 4

Offset: (binary) 101110100110 -> (decimal) 2,982

000000100000000

c. 30000 => (binary) 111010100110000

Page: (binary) 111 -> (decimal) 7

Offset: (binary) 010100110000 -> (decimal) 1,328

d. 256 => (binary) 000000100000000 (zero padded to 15 bit)

Page: (binary) 000 -> (decimal) 0

Offset: (binary) 000100000000 -> (decimal) 256

e. 16385 => (binary) 100000000000001

Page: (binary) 100 -> (decimal) 4

Offset: (binary) 000000000001 -> (decimal) 1

3. Consider a logical address space of 64 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?

Answer:

2^n = 2048 ==> n = 11 (page size)

Logical -> | 0| 1| 2| 3| 4| 5| 6| 7|
(pages)
Physical-> | 0| 1| 2| 3| 4| 5| 6| 7| 8| 9|10|11|12|13|14|15|
(frames)

a. How many bits are required in the logical address?
How many for page number and how many for page offset?

8 pages * 2048 bytes = 16,384 bytes total logical memory

16,384 = 2^14 ==> 14 bits

8 pages can be accessed with 3 bits (2^3 = 8)

2048 bytes per page can be access with 11 bits.

Page Number   Page Offset

| 3 bits | 11 bits |

b. How many bits are required in the physical address?
How many for frame number and how many for page offset?

16 frames * 2048 bytes per frame = 32,768 bytes total physical memory
32,768 = 2^15 ==> 15 bits
16 pages can be accessed with 4 bits (2^4 = 4)
2048 bytes per page can be access with 11 bits.
Frame Number   Frame Offset

| 4 bits | 11 bits |

a. Logical address: 16 bits
b. Physical address: 15 bits

4. Consider a logical address space of 32 pages with 1024 words per page; mapped onto a physical memory of 16 frames.
a. How many bits are required in the logical address?
b. How many bits are required in the physical address?

Answer:

a. 2^5 + 2^10 = 15 bits.

b. 2^4 + 2^10 = 14 bits.

5. Consider a paging system with the page table stored in memory.

a. If a memory reference takes 200 nanoseconds, how long does a paged memory reference take?

b. If we add associative registers, and 75 percent of all page-table references are found in the associative registers, what is the effective memory reference time? (Assume that finding a page-table entry in the associative registers takes zero time if the entry is there.)

Answer:

a. 400 nanoseconds: 200 nanoseconds to access the page table and 200nanoseconds to access the word in memory.

b. Effective access time = 0.75 × (200 nanoseconds) + 0.25 × (400nanoseconds) = 250 nanoseconds.

6. Consider the following segment table:

Segment	**Base**	**Length**
0	**219**	**600**
1	**2300**	**14**
2	**90**	**100**
3	**1327**	**580**
4	**1952**	**96**

What are the physical addresses for the following logical addresses?

a. 0,430
b. 1,10
c. 2,500
d. 3,400
e. 4,112

Answer:

a. Logical address < Length of the segment
   430 < 600
   Add base address with Logical address
   219 + 430 = 649

b. Logical address < Length of the segment
   10 < 14
   Add base address with Logical address
   2300 + 10 = 2310

c. Logical address < Length of the segment
   500 > 100
   illegal reference, trap to operating system

d. Logical address < Length of the segment
   400 < 580
   Add base address with Logical address

$1327 + 400 = 1727$

e. Logical address < Length of the segment

$112 > 96$

illegal reference, trap to operating system