1 processes P1, P2, and P3 ，Define semaphores, and synchronize the execution of P1, P2, and P3 by wait() and signal() on these semaphores.

2 Draw three Gantt charts，

What is the turnaround time of each process for SJF, and RR

What are the average waiting time and the average turnaround time

3 Consider the following page-reference string:

(1) LRU page replacement

(2) Optimal page replacement

(3) FIFO replacement algorithm

page fault times，Page faults rate

4 (1) Is the system in a safe or unsafe state? Why?

(2) If P_i request resource of (0, 1, 0, 0), can resources be allocated to it? Why?

In a demand paging system, the page size is 1k bytes, and the page table is as follows(assuming use decimal values), translate a logical address 10 into its corresponding physical address, and why?

frame
2
1
6

A．8202　　B．4106　　C．2058

D．1034

In a demand paging system, the page size is 1024 bytes, and the page table is as follows, would the following virtual addresses (assuming use decimal values) result in a page fault? And why?

(1) 2500

(2) 5100

valid/invalid	# frame
i	**20**
v	**15**
v	**20**
v	**29**
i	**80**
i	**53**

Page table

(1) 2500=2*1024+452 page 2 is valid, so no page fault

(2) 5100=4*1024+1004 page 4 is invalid, a page fault occurs

Consider a simple paging system with a page table containing 1024 entries of 14 bits (including one *valid/invalid* bit) each, and a page size of 1024 bytes

(a) how many bits are in the logical address?

(b) how many bits are in the physical address?

(c) what is the size of the logical address space?

(d) how many bits in the logical address specify the page number?

(e) how many bits in the physical address specify the offset within the frame?

1) 20

2) 23

3) 2^{20}B

4) 10

5) 10

(a) There are 10+10=20 bits in the logical address

(b) There are 13+10=23 bits in the physical address

(c) The size of the logical address space is 2^{20} bytes

(d) There are 10 bits in the logical address specifying the page number

(e) There are 10 bits in the physical address specifying the offset within the frame

Consider a system using segmentation with paging management scheme, whose physical memory is of 2^{35} bytes. The logical address space consists of up to 8 segments. Each segment can be up to 2^{13} pages, and page size of 512 bytes,

(1) How many bits in the logical address specify the page number?

(2) How many bits are there in the entire logical address?

(3) What is the size of a frame?

(4) How many bits in the physical address specify the frame number?

(5) How many bits in the physical address specify the frame offset?

(6) How many entries are there in the page table for each segmentation, i.e. how long is the page table for each segmentation?

(1) Each segment consists of up to 2^{13} pages, so $\log_2(2^{13}) = 13$ bits in the logical address specify the page number

(2) The entire logical address consists of segmentation and offset in segmentation, and offset in segmentation is divided into page number and page offset, so entire logical address has

$$\log_2(8) + \log_2(2^{13}) + \log_2(512) = 3 + 13 + 9 = 25 \text{ bits}$$

(3) the size of a frame is equal to that of a page, i.e., 512 bytes

(4) the total size of physical memory is 2^{35} bytes, and each frame is $2^9 = 512$ bytes, so there are 26 = (35 – 9) bits in the physical address specify the frame number

(5) the number of bits in the physical address specifying the frame offset depends on the size of the frame, so $\log_2(2^9) = 9$ bits in the physical address specify the frame offset

(6) each entry in the page table corresponds to a page in the segment, each segment can be up to 2^{13} pages, so there are 2^{13} entries in the page table for each segment.

A file system uses 256-byte physical blocks. Each file has a directory entry giving the file name, location of the first block, length of file, and last block position. Assuming the last physical block read is 100, block 100 and the directory entry are already in main memory.

For the following two file allocation algorithms, how many physical blocks must be read to access the specific block 600 (including the reading of block 600 itself), and why?

(1) contiguous allocation

(2) linked allocation

(1) 1

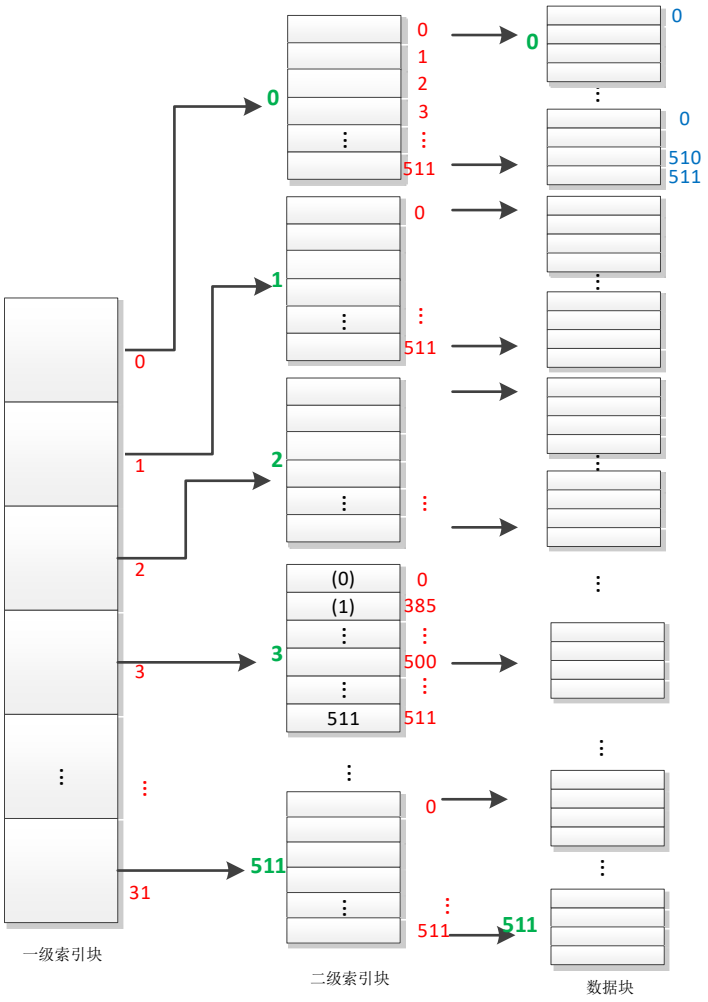
Contiguous allocation supports direct access on arbitrtary physical blocks in a file. To access physical block 600, the bolck 600 can be directory read into memory.

(2) 500

Linked allocation is used only for sequential-access files, file system can only read blocks one by one, directed by file pointers.

To find the phyical block 600 in the file, file system locates on the ***101th*** block following the ***100th*** block just read, and reads block 101, block 102, …, until block 600.

Consider a file system in which a directory entry can store up to 32 disk block addresses. For the files that is not larger than 32 blocks, the 32 addresses in the entry serves as the file's index table. For the files that is larger than 32 blocks, each of the 32 addresses points to an indirect block that in turn points to 512 file blocks on the disk, and the size of a block is 512-bytes. What is the largest size of a file?


0
1
2
3
31
一级索引块
0
1
2
3
511
二级索引块
(0)
(1)
511
0
385
500
511
0
511
0
510
511
数据块

32 x 512 =16384 file blocks

16384 x 512 = 8388608 bytes

or

2^5 x 2^9 x 2^9 = 2^{23} bytes

In the file system on a disk with physical block sizes of 512 bytes, a file is made up of 128-byte logical records, and each logical record cannot be separately stored in two different blocks. The disk space of the file is organized on the basis of indexed allocation, and a block address is stored in 4 bytes. Suppose that 2-level index blocks be used to manage the data blocks of the file, answer the following questions:

1) What is the largest size of the file?
2) Given 2000, the number of a logical record in the file, how to find out the physical address of the record 2000 in accordance with the 2-level index blocks.

1)

512/4=128

128*128*512=128*64 KB=8192KB

or $=2^{23}$ bytes = 8 MB

or = 8388680

2)

512/128=4

2000/4=500

in 3 block in the first –level index block

in 116 block in the second-level index block