OS HW3

108820018 蔡翔宇

- 7.6

(a), this change could be made safely without any problems.

(d), this change could be made safely without any problems.

(e), it could be allowed if resources were allocated to new processes without causing the system entering unsafe state.

(f), this change could be made safely without any problems.

- 7.13

a.

	need					available					new available			
P0	2	2	1	1	<=	3	3	2	1	T	5	3	2	2
P1	2	1	3	1	<=	5	3	2	2	F				
P2	0	2	1	3	<=	5	3	2	2	F				
P3	0	1	1	2	<=	5	3	2	2	T	6	6	3	4
P4	2	2	3	3	<=	6	6	3	4	T	7	10	6	6
P1	2	1	3	1	<=	7	10	6	6	T	10	11	8	7
P2	0	2	1	3	<=	10	11	8	7	T	12	12	8	10
safe seq. : <0, 3, 4, 1, 2> => safe.														

b. Yes

	need					available					new available			
P0	2	2	1	1	<=	2	2	2	1	T	4	2	2	2
P1	2	1	3	1	<=	4	2	2	2	F				
P2	0	2	1	3	<=	4	2	2	2	F				
P3	0	1	1	2	<=	4	2	2	2	T	5	5	3	4
P4	2	2	3	3	<=	5	5	3	4	T	6	9	6	6
P1	2	1	3	1	<=	6	9	6	6	T	10	11	8	7
P2	0	2	1	3	<=	10	11	8	7	T	12	12	8	10
safe seq. : <0, 3, 4, 1, 2> => safe.														

c. No.

	need					available					new available			
P0	2	2	1	1	<=	3	3	0	1	F				
P1	2	1	3	1	<=	3	3	0	1	F				
P2	0	2	1	3	<=	3	3	0	1	F				
P3	0	1	1	2	<=	3	3	0	1	F				
P4	2	2	3	3	<=	3	3	0	1	F				
safe seq. : <> => unsafe.														

- 7.15

```
semaphore ok_to_cross = 1
void EnterBridge(){
        cross.wait()
}

void ExitBridge(){
        cross.signal()
}
```

- 8.1

External fragmentation occurs when the total free space in the memory is enough for a process, but it is not contiguous so that it can't be distributed to a new process. It can be solved by paging.

Internal fragmentation occurs when the memory size distributed to a process is larger than the process needed, these space can't be used by the process, and also cannot be used by the system until the process release it.

- 8.9

Paging requires more memory overhead to maintain the translation structures. Segmentation requires just two registers per segment, one to maintain the base of the segment and the other to maintain the extent of the segment. Paging on the other hand requires one entry per page, and this entry provides the physical address in which the page is located.

- 8.16

a. conventional single-level page table: 2^32 / 2^12 = 2^20
b. 512M = 2^29
   inverted page table: 2^29 / 2^12 = 2^17

- 9.8

a.

LRU																		
7	2	3	1	2	5	3	4	6	7	7	1	0	5	4	6	2	3	0	1
7	7	7	2		2	2	5	3	4		6	7	1	0	5	4	6	2	3
	2	2	3		1	5	3	4	6		7	1	0	5	4	6	2	3	0
		3	1		5	3	4	6	7		1	0	5	4	6	2	3	0	1
18 page faults																			

b.

FIFO																		
7	2	3	1	2	5	3	4	6	7	7	1	0	5	4	6	2	3	0	1
7	7	7	2		3		1	5	4		6	7	1	0	7	4	6	2	3
	2	2	3		1		5	4	6		7	1	0	7	4	6	2	3	0
		3	1		5		4	6	7		1	0	7	4	6	2	3	0	1
17 page faults																			

c.

Optimal																			
7	2	3	1	2	5	3	4	6	7	7	1	0	5	4	6	2	3	0	1
7	7	7	1		1		1	1	1			1		1	1	1	1		
	2	2	2		5		5	5	5			5		4	6	2	3		
		3	3		3		4	6	7			0		0	0	0	0		
13 page faults																			

- 9.11

The case LFU that performs better:

LFU							LRU						
1	1	2	3	4	5	1	1	1	2	3	4	5	1
1		1	1	1	1		1		1	1	1	2	3
		2	2	2	3				2	2	2	3	4
			3	3	4					3	3	4	5
				4	5						4	5	1
5 page faults							6 page faults						

The case that LRU performs better:

LFU						LRU					
1	1	2	3	4	2	1	1	2	3	4	2
1		1	1	1	1	1		1	1	2	
		2	2	3	4			2	2	3	
			3	4	2				3	4	
5 page faults						4 page faults					

- 9.17
- 9.19

Thrashing is caused by under allocation of the minimum number of pages required by a process, forcing it to continuously page fault. The system can detect thrashing by evaluating the level of CPU utilization as compared to the level of

multiprogramming. It can be eliminated by reducing the level of multiprogramming.