鄭琳玲
108590056
資工三

Operating system - HW3

7.6

a) This could be changed without any problems.

b) This could have an effect on the system & introduce the possibilities of deadlock as the safety of the system assumed there were a certain number of available resources.

c) This could give an impact on the system & the posibilities of deadlock.

d) could be changed without any problems.

e) It is allowed since the resources were allocated to new process such that the system doesn't enter an unsafe state.

f) could be changed without any problems.

7.13

a)

	A	B	C	D
P0	2	2	1	1
P1	2	1	3	1
P2	0	2	1	3
P3	0	1	1	2
P4	2	2	3	3

need(P0) < Available, so P0 can take all resources

Available = (3 3 2 1) ÷ (2 0 0 1) (P0's allocation)

= (5 3 2 2)

need (P3) < Available, so P3 will go next.

Available = (5 3 2 2) ÷ (1 3 1 2) = (6 6 3 4)

so, the save seqvence will be P0, P3, P4, P1 & P2

b)

Reqvest from P1 is (1 1 0 0) & available is (3 3 2 1)

7.11 As reqvest (P1) < Available, reqvest can be granted.

c) Reqvest from P4 is (0 0 2 0) & available is (3 3 2 1)

As reqvest (P4) < Available, reqvest can be granted.

7.15

```
semaphore ok_to_cross = 1;
void enter_bridge () (
        P (ok_to_cross);
)
void exit_bridge () (
        V (ok_to_cross);
)
```

8.1

Internal fragmentation is the area that a process occupies but can't utilize. The system wouldn't be be able to use this space until the operation completes. While Extrenal fragmentation occurs when total free memory is sufficient for the new process but isn't contiguous & hence can't meet the request. Storage is divided into little holes.

8.9

paging needs greater memory overhead to keep the translation structures up to date. Segmentation requires just two registers per segment: one to keep the segment's base & the other to maintain the segment's extend. While paging, with this entry including the physical location of the page.

8.16

a) $\dfrac{2^{32}}{2^{12}} = 2^{20}$

$2^{32} \rightarrow$ total virtual memory size

$4KB / 2^{12} \rightarrow$ size of single page

$\dfrac{2^{32}}{2^{12}} = 2^{20} \rightarrow$ total number of pages of virtual memory.

b) $2^{29} \rightarrow$ total physical memory

$2^{12} \rightarrow$ page size / frame size

$\dfrac{2^{29}}{2^{12}} = 2^{17} \rightarrow$ total number of frames in physical memory.

9.8

a) LRU replacement

7
7 2
7 2 3
1 2 3
1 2 5
3 2 5
3 4 5
3 4 6
7 4 6
7 1 6
7 1 0
5 1 0
5 4 0
5 4 6
2 4 6
2 3 6
2 3 0
1 3 0

18 page faults

b) FIFO replacement

7
7 2
7 2 3
1 2 3
1 2 5
3 2 5
3 4 5
3 4 6
7 4 6
7 1 6
7 1 0
5 1 0
5 4 0
5 4 6
2 4 6
2 3 6
2 3 0
1 3 0

17 page faults.

c) optimal replacement.

7
7 2
7 2 3
1 2 3
1 5 3
1 5 4
1 5 6
1 5 7
1 5 0
1 4 0
1 6 0
1 2 0
1 3 0

13 page faults.

9.8 LRU replacement

	7	2	3	1	2	5	3	4	6	7	7	1	5	0	4	6	2	3	0	1
F1	7	7	7	1	1	1	3	3	3	7	7	7	5	5	5	6	6	6	0	0
F2		2	2	2	2	2	2	4	4	4	4	4	4	0	6	0	2	2	2	1
F3			3	3	3	5	5	5	6	6	6	1	1	1	4	4	4	3	3	3
	PF	PF	PF	PF	Hit	PF	PF	PF	PF	PF	Hit	PF	PF	PF	PF	PF	PF	PF	PF	PF

Page fault : 18
Hits : 2

FIFO Replacement

	7	2	3	1	2	5	3	4	6	7	7	1	5	0	4	6	2	3	0	1
F1	7	7	7	1	1	1	1	1	6	6	6	6	5	5	5	6	6	6	0	0
F2		2	2	2	2	5	5	5	5	7	7	7	7	0	0	0	2	2	2	1
F3			3	3	3	3	3	4	4	4	4	1	1	1	4	4	4	3	3	3
	PF	PF	PF	PF	Hit	PF	Hit	PF	PF	PF	Hit	PF	PF	PF	PF	PF	PF	PF	PF	PF

Page fault = 17
Hits = 3

Optimal replacement

	7	2	3	1	2	5	3	4	6	7	7	1	5	0	4	6	2	3	6	1
F1	7	7	7	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
F2		2	2	2	2	5	5	5	5	5	5	5	5	5	4	6	2	3	3	3
F3			3	3	3	3	3	4	6	7	7	7	7	0	0	0	0	0	0	0
	PF	PF	PF	PF	Hit	PF	Hit	PF	PF	PF	Hit	Hit	Hit	PF	PF	PF	PF	PF	Hit	Hit

Page fault = 13
Hit = 7

9.11

LFU → checks the old page & frequency of that page, if the frequency of the page is larger than old page; we can't remove it.

LRU → an algorithm where the page to be replaced is least recently used.

Ex 1.

LFU

	1	1	2	3	4	5	1
F1	1	NF					NF
F2			2			5	
F3				3			
F4					4		

Faults = 5

LRU

	1	1	2	3	4	5	1
F1	1	NF				5	1
F2			2				
F3				3			
F4					4		

Faults = 6

Assuming that there are 4 frames, for first entry, first frame is marked. For entry 1, no page faults occurs. For the next three entries, they are filled. For page entry 5, LFU is accessed, so, either 2, 3, 4 will be replaced. Then for the last entry, 1 no page fault occurs. Whereas, if we used LRU, page entry 5 would be in the first frame, thus causing page fault to occur in the last entry.

ex 2

Assuming that there are 4 frames,

LFU

	1	1	2	3	4	5	2
F1	1	NF					
F2			2			5	2
F3				3			
F4					4		

Faults = 6

LRU

	1	1	2	3	4	5	2
F1	1	NF				5	
F2			2				NF
F3				3			
F4					4		

Faults = 5

1st entry → no page fault

3 next entries → filled

Page entry 5

↳ LFU is accessed. Thus either 2,3,4 is replaced. Allowing frame 2 used.

last entry

↳ 2 page fault occurs, if we had used LRU, the page entry 5 would be in first frame, causing no page faut to occur in last entry.

9.17

① i) 0

a)
ii) when a new page is associated with the frame

ii) when a page associated with the frame is no longer required

iii) page frame with the smallest counter is replaced; if tie occurs, FIFO replacement will be used.

b)

	1	2	3	4	5	3	4	1	6	7	8	7	8	9	5	4	5	4	2
F1	1(1)				5(2)								8(3)						
F2		2(1)						1(2)							5(3-1=2)			NF	
F3			3(1)			NF			6(2-1=1)		8(2-1=1)			NF					
F4				4(1)			NF			7(2)		NF							

Faults = 14

c)

	1	2	3	4	5	3	4	1	6	7	8	7	8	9	7	8	9	5	4	5	4	2
F1	1							NF	6		8		NF			NF						
F2		2			5													NF		NF		2
F3			3			NF				7	8	NF			NF				4		NF	
F4				4			NF							9			NF					

Faults = 11