93 /

Faculty of Natural and Applied Sciences

Time: 2 hrs

Final

Name : ***Section :*** A

Part I: Questions and Answers

1. When can the deadlock among processes occur?
   When can the starvation among processes occur? (5points)

5 ✓ - Deadlock occurs when a process wants a resource but can not get it because it is held by another process.

- Starvation can occur when a process is always the one who is waiting for others to free. Example: same process is always killed.

2. List and briefly, explain the methods used for handing deadlocks? (5 points)

- Ostrich Method → ignore the deadlock.
- Deadlock Avoidance → Banker's Algorithm
- Deadlock pevention → Negate only one of the deadlock conditions.

5 ✓

3. List and briefly, explain the necessary conditions of a deadlock? (5 points)

5

1- No preemption : process can only free resource voluntarily.

2- Hold and wait : process is always holding a resource until the other one he wants is freed.

3- Circular wait : P_1 waits for P_2 and P_2 waiting for P_3 while P_3 is waiting for P_1

4. What is a race condition? How can we prevent a race condition? (5 points)

5

when 2 processes are racing to occupy a critical section this can be solved by using mutual exclusion ⇒ only one process can use the critical section while the other one has to wait

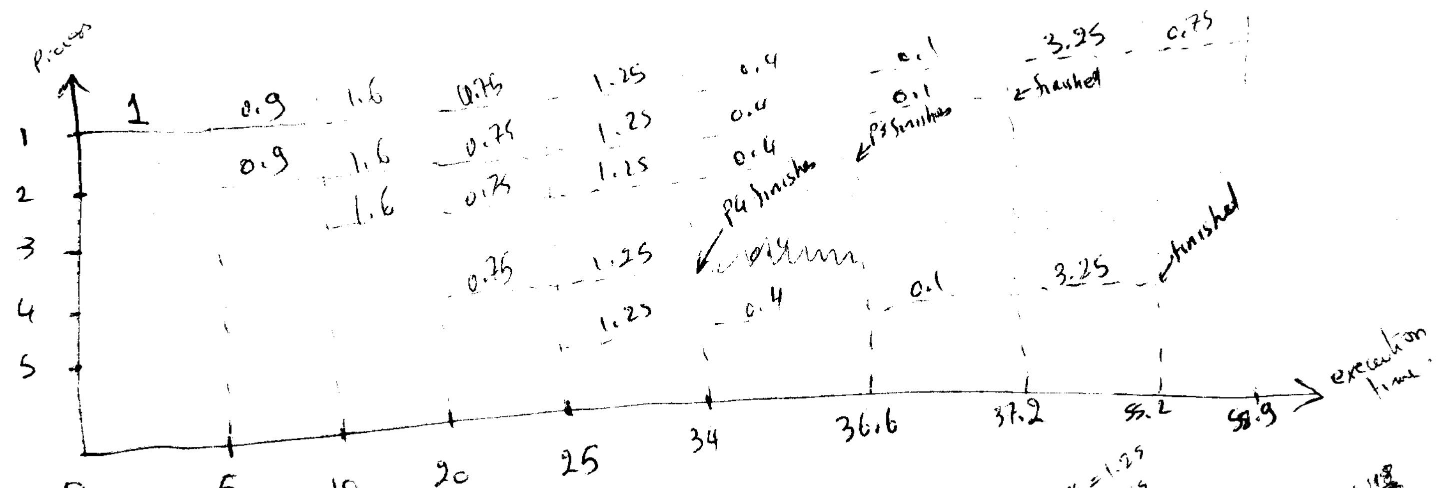
5. Consider the following set of processes, with the length of the CPU burst time given in milliseconds: (15 points)

15

Process	Burst Time	Arrival time
P1	10	10:00
P2	5	10:05
P3	4	10:10
P4	2	10:20
P5	5	10:25

Draw the chart illustrating the multiprogramming analysis and give the final execution time . Knowing that the performance of the cpu / process is

# 1	2	3	4	5
0.20	0.18	0.16	0.15	0.13



execution time for all processes to complete is almost 59 seconds

$0.20 \times x = 0.75$, $x = \frac{75}{20} \approx 3.7$

$0.18 \times x = 3.25$, $x = \frac{325}{18} \approx 18$

$0.13 \times x = 1.25$, $x = \frac{1.25}{0.13}$

$0.15 \times x = 0.4$, $x = \frac{0.4}{0.15} = 2.6$

6. P is a set of processes. R is a set of resources. E is a set of request or assignment edges. The sets P, R, and E are as follows: (5 points)

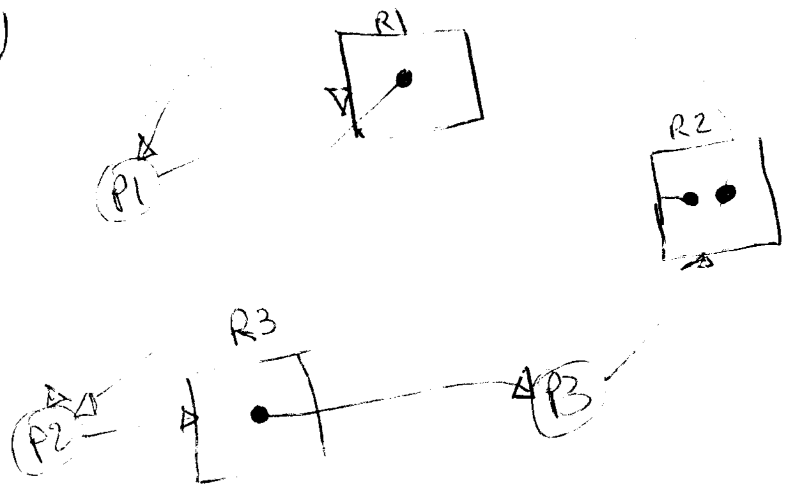
P = {P1, P2, P3}
R = {R1, R2, R3}
E = {P1 -> R1, P2 -> R3, P3-> R2, R1 -> P2, R2 -> P2, R2 -> P1, R3 -> P3}

5

R1 has one instance. R2 has two instances. R3 has one instance.

a. Draw the resource-allocation graph.
b. Is there any deadlock in this situation? Briefly Explain.

a)



b) yes we have a deadlock (all conditions are satisfied)
no process can execute because:
P1 is holding instance of R2;
P1 is waiting for R1.
R1 is held by P2 which is holding the other instance of R2 and Requesting R3
but R3 is held by P3 which will not free R3 unless it was given R2 but R2 is occupied.
all processes are waiting for each others
no one can execute, we have a circular loop.

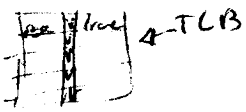
7. a) What is the difference between a virtual (logical) address and a physical address?
b) What is associative memory (translation look-aside buffer, TLB)? Why do some systems use associative memory in addition to page tables?
c) Explain the difference between internal and external fragmentation.
d) What type of fragmentation occurs in simple paging systems? Explain.
e) What type occurs in systems that use pure segmentation? Explain. (10 points)

a) - Virtual address is not real; it is the address that is fresh out of CPU before it goes into memory.
- physical Address is real address in Memory.

2

b) TLB is a Memory that stores pages and their corresponding frames. It has a [V] bit that tells if it is up to date or not

1



c) - external fragmentation is when we have hole in segments because of processes that do not use all the space they occupy.
- Internal fragmentation occurs withing the frames leaving holes for unused space

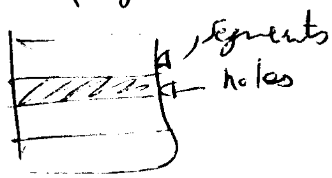
2

d) Internal because it is within pages.

2

e) Segments ⇒ external

2



space is externally fragmented

8. Consider a swapping system in which memory consists of 500K as shown below.(10 points)

considering process arrive here

```
+---+-----+------+--------+--------+-----+--+
|xxx| P3  | xxxx | P5     | xxxxxx | P7  |xx|
+---+-----+------+--------+--------+-----+--+
0   50   160    240      320      420  480 500
```

Note that P3, P5, and P7 are processes in memory. Assume that process P3 was just swapped into memory. Consider each of the four swapping algorithms discussed in class: first fit, next fit, best fit and worst fit, Complete the table showing where each of the processes will be loaded.

Assume that new processes arrive in the order P8, P9, P10, and are of size 50K, 70K, and 45K, respectively.

	P8	P9	P10	
First	0-50	160-240	320 - 420	Done
Next	~~160-240~~	~~320-420~~	~~out of memory~~	
Best	0-50	160-240	320 - 420	Done
Worst	320-420	160-240	0-50	Done

First: first location that fits
Next: Next location that fits after first
Best: smallest location that fits
Worst: Biggest location that fits.

... we could defrag ⇒ stick 0-50 to 480-500

If a process won't fit, write "out of memory" in the appropriate slot, then indicate if compaction could be used to correct the problem.

Done before

10

9. Suppose there are 16 virtual pages and 4 page frames. Determine the number of page faults that will occur with the reference string ~~1 2 3 4 2 1~~ 5 6 ~~2~~ 1 2 3 ~~7~~ 6 3 2 ~~1 2 3 6~~ if the page frames are initially empty, using each of the following page replacement algorithms: a. LRU b. FIFO (10 points)

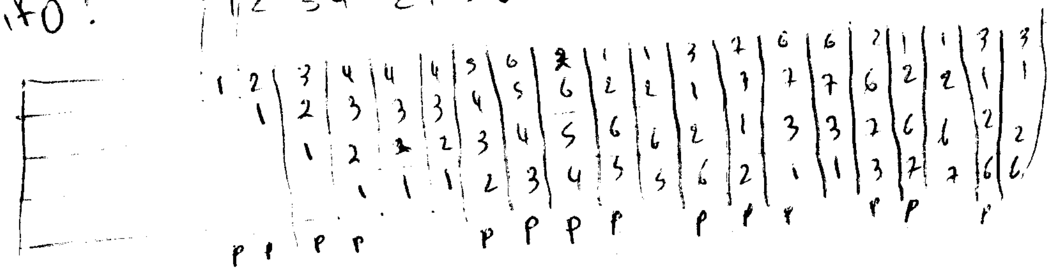
LRU

	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
page1 0	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
page 2 1		1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3
pg 3 2			1	2	3	4	2	1	5	6	6	1	2	3	7	6	3	3	1	2
pg 4 3				1	1	3	4	2	1	5	5	6	1	2	2	7	6	6	6	1
Virtual 0								3	4	4	4	4 5	6	1	1	1	7	2	7	7
1									3	3	3	3 4	5	5	5	5	5	5	5	5
2													4	4	4	4	4	4	4	4
3																				
4																				
5																				
6																				
7																				
8																				
9																				
10																				
11																				
12																				
13																				

LRU

=> 10 Page faults

b) Fifo: 1 2 3 4 2 1 5 6 2 1 2 3 7 6 3 2 1 2 3 6



14 page faults

10. A small computer has four page frames. At the first clock tick, the R bits are 0111 page 0 is 0, the rest are 1). At subsequent clock ticks, the values are 1011, 1010, 1101, 0010, 1010, 1100, and 0001.
If the aging algorithm is used with an 8-bit counter, give the values of the four counters after the last tick. (10 points)

0111 1011 1010 1101 0010 1010 1100 0001

counters	1011	1010	1101	0010	1010	1100	0001
P1	1000 0000	1000 0000	1000 0000	0000 0000	1000 0000	1000 0000	0000 0000
P2	0100 0000	0100 0000	1100 0000	0000 0000	0100 0000	1100 0000	0000 0000
P3	1010 0000	1010 0000	0110 0000	1000 0000	1010 0000	0110 0000	0000 0000
P4	1101 0000	0101 0000	1011 0000	0100 0000	0101 0000	0011 0000	0100 0000

2

11. A computer has four page frames. The time of loading, time of last access, and the R and M bits for each page are as shown below (the times are in clock ticks): (10 points)

5

Page	Loaded	Last Ref.	R	M
0	126	280	1	0
1	230	265	0	1
2	140	270	0	0
3	110	285	1	1

(a) Which page will NRU replace?
(b) Which page will FIFO replace?
(c) Which page will LRU replace?
(d) Which page will second chance replace?

a) not recently used ⇒ least loaded and referenced ⇒ Page 3.
b) First one loaded goes out ⇒ P3
c) least recently used ⇒ P1 referenced 265 times.
d) ... the first loaded so it it was freshly loaded and removes the second one

10

12. List the main differences between Unix , Linux and Windows (10 points)

Unix and Linux are Very much alike.

Windows NT (new Technology)

Unix	Linux	Windows NT
- stable system	- stable system	- Prone to crashes
- Works on only compatible systems such as Sun	- works on PC	- Works on PC
- Designed for multi-users	- Designed for multi-users	- Designed for single user
- No Viruses	- No Vires	- Viruses
- can see windows	- can see windows	- Can not see but itself
- work on any partition	- work on any partition	- works only on main partition
- supports TCP/IP	- Supports TCP/IP	- Supports TCP/IP
- multi tasking efficient	- Multitasking efficient	- Multitasking less efficient
- free, open source	- free, open source	- Expensive but Marketed More
- scheduling: aging - process work ⇒ mutex = 1 - process finishes ⇒ mutex = 0	- Scheduling: Loads and Manages processes into pages	- scheduling: Does not guarantee that the needed thread will execute in a given time.
- Memory Management: First in - 1: paging - 2: swapping. due to sharing, does not need to swap all processes ⇒ fast	- Memory Management: frees the space that is not needed to make a process work ⇒ parts of processes can work without having to be all loaded into Memory	- Memory Management: 1- Virtual alloc: allocates process to memory 2- Virtual free: Frees unneeded space
File Manipulation: Tree → Directories ↓ files; can be manipulated by user; - root is main; - can force Permissions; - [illegible]	same as Unix	files Management: - Tree - Directory. - Can be Manipulated by users; types: Fat 16, Fat 32 } Not secure; NTFS → can recover [illegible]

9

Part II: Unix – Linux commands (1 point each)

Explain the following commands

1. date > hello

insert the file date in file hello

2. grep track *.out

find string "track" in any file that has .out extension

3. alias v='vi nouhad'

command 'v' becomes 'vi nouhad' ⇒ when v is excuted it will open a file called nouhad in vi.

4. cat < rizk

read content of file rizk.

5. chmod 600 out

change permissions of file out to 600 ⇒ user group others
r x nothing nothing

read execute

6. who | sort

pipe the who and sort commands ⇒ sort the output of the who command, who tells which user you are

7. #!bin/ksh –f

start to write a script using the Kourne shell

8. Export PS1="byecsc414"

the prompt in the terminal becomes byecsc414 instead of $

9. cc -o ok myfile.cpp

make the file "myfile.cpp" executable by only typing 'ok' ⇒ file ok is an executable file linked to myfile.cpp

10. make

make compiles the files that are linked in the make file that has already been coded, after make, the file can be executed.

Good Luck