NATIONAL UNIVERSITY OF SINGAPORE

CS2106 – INTRODUCTION TO OPERATING SYSTEMS

(Semester 2: AY2014/15)

Time allowed: 2 hours

INSTRUCTIONS TO STUDENTS

1. Please write your Matriculation Number. Do not write your name.
2. This assessment consists of **fifteen** questions and **eighteen** printed pages, including this page.
3. This is a **CLOSED BOOK** assessment. You may use any approved calculators but not any PDA or laptop, especially those capable of external connectivity or communication.
4. Write your answers on this script only. Answer in the spaces given.
5. All question carry the **same weight**. I.e., the marks for each question will be scaled to 10, for a maximum possible total of 150.

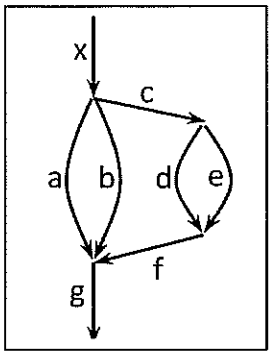
MATRICULATION NO: _____

This portion is for examiners' use only

Question	Marks	Question	Marks	Question	Marks
1		6		11	
2		7		12	
3		8		13	
4		9		14	
5		10		15	

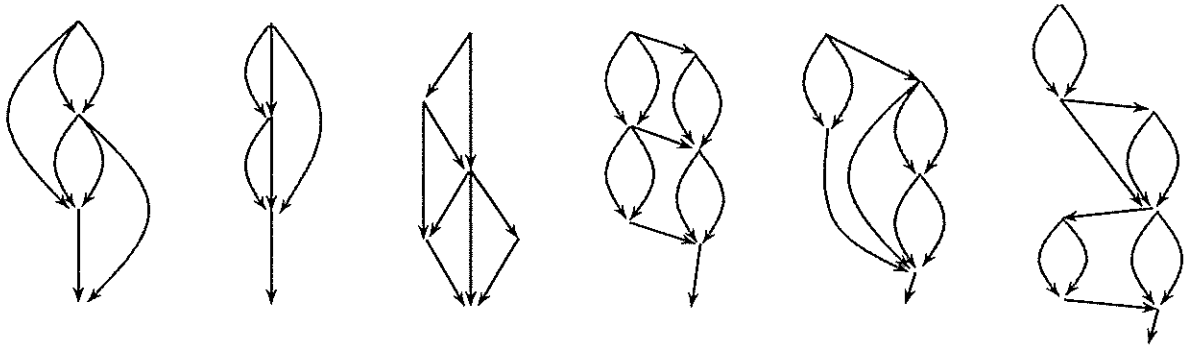
TOTAL (Scaled): _____/150

1. [12 marks]
   a. Express the process flow graph shown below using cobegin/coend notation. (Use only the smallest possible number of cobegin/coned statements.)



   b. Consider each of the following process flow graphs. Determine which of them can be expressed using cobegin/coend:

(1) (2) (3) (4) (5) (6)



	(1)	(2)	(3)	(4)	(5)	(6)
expressible using cobegin/coend (yes/no)						

2. [12 marks] Consider the program shown below, where the a, b, …, h represent some arbitrary processes. Draw the corresponding process flow graph.

```
    n1 = 2; n2 = 4;
    fork A;
    fork B;
    c;
    fork D;
    fork E;
    f;
    join n2, H; quit;
A:  a; join n2, H; quit;
B:  b; join n2, H; quit;
D:  d; join n1, G; quit;
E:  e; join n1, G; quit
G:  g; join n2, H; quit;
H:  h; quit;
```

3. [8 marks] Consider the 3 processes below, where A, B, C are arbitrary computations:

process 1:	process 2:	process 3:
while(1){ P(s1); A; V(s2); }	while(1){ P(s2); B; V(s1); }	while(1){ P(s1); C; V(s1); }

The initial semaphore values are: s1 = 0, s2 = 1
Assuming that all processes run concurrently, which of the following are possible sequences of executions of A, B, C:

	yes/no
B A B A B A ...	
B C C C C C ...	
A B C A B C ...	
B C A B C A ...	
C A B A B C ...	
B C C C A A ...	
B A B C A B ...	
B C A B A B ...	
All of the above	
None of the above	

4. [24 marks] Consider the following two solutions to the readers/writers problem. CS is the critical section where either *one* writer or *one or more* readers should be allowed to enter:

Solution 1

```
reader() {
  P(mutex1);
  rc++;
  if(rc==1) P(cs);
  V(mutex1);
  CS;
  P(mutex1);
  rc--;
  if(rc==0) V(cs);
  V(mutex1); }
```

```
writer() {
  P(cs);
  CS;
  V(cs); }
```

Solution 2

```
reader() {
  P(mutex1);
  rc++;
  if(rc==1) P(cs);
  V(mutex1);
  CS;
  P(mutex1);
  rc--;
  if(rc==0) V(cs);
  V(mutex1); }
```

```
writer() {
  P(mutex2);
  wc++;
  if(wc==1) P(cs);
  V(mutex2);
  P(w)
  CS;
  V(w)
  P(mutex2);
  wc--;
  if(wc==0) V(cs);
  V(mutex2); }
```

Initially: mutex1 = mutex2 = cs = w = 1; rc = wc = 0;

a. Answer the following questions for each solution (yes/no):

	Solution 1	Solution 2
Can multiple readers enter CS concurrently?		
Can multiple writers enter CS concurrently?		

b. Assume that the following read/write requests arrive while **r1** is in CS: **w1, w2, r2.** Which semaphore, if any will the corresponding process be blocked on?

	Solution 1	Solution 2
w1 blocked on:		
w2 blocked on:		
r2 blocked on:		

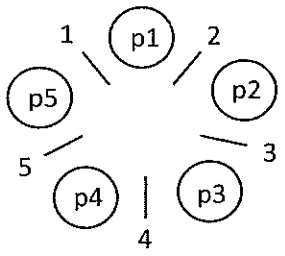
Which request will enter CS next:		
Can writers starve?		

c. Assume now that the following read/write requests arrive while **w1** is in CS: **r1, r2, w2.** Which semaphore, if any will the corresponding process be blocked on?

	Solution 1	Solution 2
r1 blocked on:		
r2 blocked on:		
w2 blocked on:		

Which request will enter CS next:		
Can readers starve?		

5. [30 marks] The diagram on the right shows the Dining Philosophers problem with 5 philosophers (p1 through p5) and 5 forks (1 through 5).



Consider the following strategies for picking up the forks:

A. Each philosopher picks up the *right* fork first, then the *left* one

B. Each *odd*-numbered philosopher picks the *lower*-numbered fork first, then the higher-numbered one. Each *even*-numbered philosopher picks the *higher*-numbered fork first, then the *lower*-numbered one.

C. Each philosopher picks either the left or right fork at *random*, then it picks the other fork.

For each of the strategies, determine if it possible to **violate concurrency** in that only 1 philosopher is eating while the remaining 4 are blocked. If so then show **one** possible scenario in the table below. Specifically, state which forks each philosopher is holding and which fork he is requesting at the time of the violation:

	Strategy A		Strategy B		Strategy C	
	Holding	Requesting	Holding	Requesting	Holding	Requesting
p1						
p2						
p3						
p4						
p5						

6. [15 marks] Consider the following elevator algorithm from the book:

```
monitor elevator {
    int direction=1, up=1, down=-1, position=1, busy=0;
    condition upsweep,downsweep;

request(int dest) {
    if (busy) {
        if ((position<dest) || ((position==dest) && (direction==up))) upsweep.wait(dest);
        else downsweep.wait(-dest);
    }
    busy = 1;
    position = dest;
}

release() {
    busy = 0;
    if (direction==up) {
        if (!empty(upsweep)) upsweep.signal;
        else {
                direction = down;
                downsweep.signal;
        }
     }
    else if (!empty(downsweep)) downsweep.signal;
        else {
                direction = up;
                upsweep.signal;
        }
}
}
```

This code services all requests in both directions, i.e., when moving up and when moving down.
On the next page, rewrite this code to perform the "circular scan" algorithm, which services all requests in **only one direction** (say up). When there are no more requests in that direction, it returns to the lowest request and again continues servicing all requests in the same upward direction.

For example, assume the following requests arrive while the elevator is at position 50:

10, 20, 50, 60, 5, 55, 70

It would service them in the order of 50, 55, 60, 70, and then continue with 5, 10, 20.

Hints:

- Instead of upsweep and downsweep, use two queues named upsweep1 and upsweep2.
- Instead of direction use phase (which can be 1 or 2).
- Eliminate the variables up and down.
- Feel free to abbreviate (e.g., p for position, d for destination, sig for signal, etc.)

7. [18 marks] Consider the implementations of P/V operations on semaphores where
   - P(s) is busy-waiting when s = 0
   - all P and V operations are critical sections (surrounded by binary mutex semaphores, also implemented using busy-wait)

   Assume 2 processes (p1 and p2) start executing the following code at the same time (CS is a critical section): **P(s); CS; V(s);**

   Make the following assumptions:
   - Initially s = 1
   - Executing CS takes c time units
   - Executing a P operation takes p time units
   - Executing a V operation takes v time units
   - There is only a single CPU
   - The time quantum is very small compared to c, p, or v and the time of a context switch is 0

   a. Assume both processes start executing concurrently at time 0. Process p1 wins the race and enters the CS first. Determine at what time each process completes each of its operations:

      - p1 completes P(s) at time: _____
      - p1 completes CS at time: _____
      - p1 completes V(s) at time: _____
      - p2 completes P(s) at time: _____
      - p2 completes CS at time: _____
      - p2 completes V(s) at time: _____

   b. Consider now a different implementation of P and V where P blocks the process when s = 0, instead of busy waiting. The timing of the process that wins the race is unaffected but the process that blocks executes for an additional b units of time at the end of its P operation to block itself (where b<c). Assuming again that p1 and p2 start concurrently at time 0 and p1 wins the race, determine the following:

      - p1 completes P(s) at time: _____
      - p1 completes CS at time: _____
      - p1 completes V(s) at time: _____

8. [12 marks] Consider a file system analogous to the last project of this course. The following series of commands is issued. Show the corresponding output for each of the read operations:

```
in
cr a
op a
cr b
op b
wr 1 x 2
wr 1 y 64
wr 2 z 192
sk 1 0
rd 1 5      _____
cr c
cr d
op c
rd 1 5      _____
cl 1
op a
sk 1 64
rd 1 5      _____
sk 2 64
wr 2 w 3
sk 2 62
rd 2 7      _____
```

a. List all files that exist at the end of the sequence _____

b. Which of these files are still open _____

9. [15 marks] Consider three processes, p1, p2, p3, with the following arrival times and total service times (t):

process	arrival	t
p1	0	5
p2	2	7
p3	3	2

a. Determine the start and end time of each process under the following scheduling disciplines:

Round robin with a quantum of 1.
Assume that any new process starts running immediately at the time of its arrival.

	p1	p2	p3
Start time			
End time			

MLF (multi-level feedback) with 5 priority levels and T=1
Assume that at each level, processes are running in FIFO order.

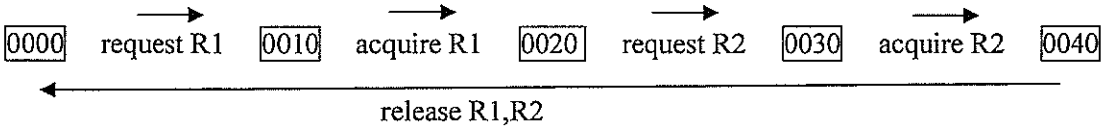
	p1	p2	p3
Start time			
End time			

b. At which priority level will each process terminate under the above MLF discipline:

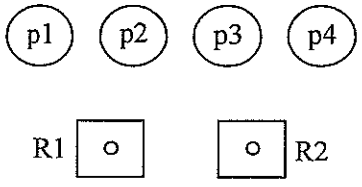
	p1	p2	p3
Final priority level			

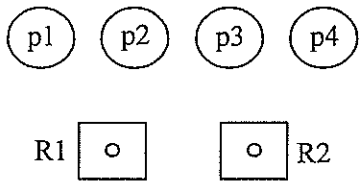
10. [35 marks] Consider 4 processes, p1, p2, p3, p4, and 2 resources, R1, R2.
    - p1 and p3 first request (and acquire) R1, then R2, then release both resources at the same time
    - p2 and p4 first request (and acquire) R2, then R1, then release both resources at the same time

    Since there are 4 processes, the system state is represented by a 4-tuple, where the first digit represents the operations of p1, the second digit represents the operations of p2, etc. For example, when only p3 is active, the system would cycle through the following states:



a. Complete the following state diagrams corresponding to the states **3010** and **1141**, respectively:





b. Determine which of the states listed in the table are reachable. For each reachable state answer the questions below:

	1234	1411	2020	3301	1311
Is the state reachable? If yes then:					
Is it a deadlock state?					
Is p1 blocked?					
Is p2 blocked?					
Is p3 blocked?					
Is p4 blocked?					

11. [12 marks] Consider the portion of main memory shown below.
    - The memory uses variable partitions where tags and sizes are replicated at *both ends* of each block.
    - Each tag/size pair is represented by a single integer: A positive integer, n, represents an occupied block of size n; a negative integer, -n, represents a free block (hole) of size n.
    - The size n is the total size occupied by the block or hole. For example, addresses 0-5 contain an occupied block of size 6.
    - Each hole contains a forward and a backward link. For example, the hole at address 6 is linked to 22 and 146.

    a. Assume that the block starting at address 12 is to be deleted. In column (a), show all changes to memory that are necessary to implement this deletion (show only what's truly necessary for correctness, no optional changes)
    b. Now assume that the block starting at address 52 is to be deleted. Starting with the original configuration, show all changes in column (b).

		(a)	(b)
0	6		
1	0		
2	0		
3	0		
4	0		
5	6		
6	-6		
7	22		
8	146		
9	23		
10	4		
11	-6		
12	4		
13	56		
14	7		
15	4		
16	6		
17	66		
18	6		
19	6		
20	6		
21	6		
22	-4		
23	62		

		(a)	(b)
24	6		
25	-4		
26	8		
27	1		
28	122		
29	1		
30	1		
31	1		
32	1		
33	8		
34	6		
35	9		
36	567		
37	66		
38	33		
39	6		
40	6		
41	0		
42	9		
43	3		
44	5		
45	6		
46	-6		
47	74		

		(a)	(b)
48	62		
49	34		
50	8		
51	-6		
52	10		
53	99		
54	99		
55	88		
56	54		
57	2		
58	55		
59	88		
60	3		
61	10		
62	-8		
63	46		
64	22		
65	3		
66	3		
67	3		
68	3		
69	-8		
70	4		
71	55		

		(a)	(b)
72	54		
73	4		
74	-12		
75	110		
76	46		
77	0		
78	0		
79	0		
80	0		
81	0		
81	0		
82	0		
83	0		
84	-12		
85	54		
86	12		
87	13		
88	14		
89	15		
90	16		
...	...		

12. [38 marks] Consider a virtual memory system using paging (no segmentation) with the following specifications:

- Size of virtual address (VA) = 10 bits
- Size of physical address = 8 bits
- Size of the offset w = 5 bits
- There are currently 2 processes. The contents of the two page tables, PT1 and PT2, along with their physical addresses (in hex) are as shown in the diagram on the right
- The page tables contain frame *numbers* (not memory addresses). Hence only 0 through 7 represent valid frame numbers. Any number greater than 7 is an invalid frame number.

	PT1		PT2
00	5	80	8
01	B	81	1
02	F	82	A
03	6	83	6
04	9	84	9
05	A	85	F
06	F	86	A
07	2	87	7
08	F	88	C
09	8	89	D
0A	9	8A	5
0B	F	8B	E
0C	F	8C	9
0D	F	8D	F
0E	F	8E	F
0F	F	8F	F
10	A	90	9
11	B	91	A
12	C	92	B
13	D	93	F
14	E	94	F
15	F	95	F
16	F	96	F
17	8	97	F
18	9	98	F
19	A	99	F
1A	8	9A	F
1B	8	9B	A
1C	E	9C	A
1D	B	9D	8
1E	C	9E	9
1F	A	9F	C

a. Complete the table below by determining the contents of each frame. (State which page of which process resides in that frame or enter "free" if the frame is not occupied.)

Frame#	Contents
0	PT1
1	
2	
3	
4	PT2
5	
6	
7	

b. For each process, translate the following virtual addresses (given in hexadecimal) into the corresponding physical addresses (also in hexadecimal). If an address cannot be translated, enter "illegal"

Virtual address	Physical address under process 1	Physical address under process 2
070		
0BB		
0FF		
141		

c. Now assume that the above memory system is to be extended to use both segmentation and paging. The virtual address size will be extended to 18 bits. The sizes of pages and page tables remain as before. Answer the following questions:

i. How many segments can be implemented? (give as power of 2) _____

ii. How many pages can each segment have? (give as power of 2) _____

iii. What is the total virtual memory size? _____ (number of addressable words)

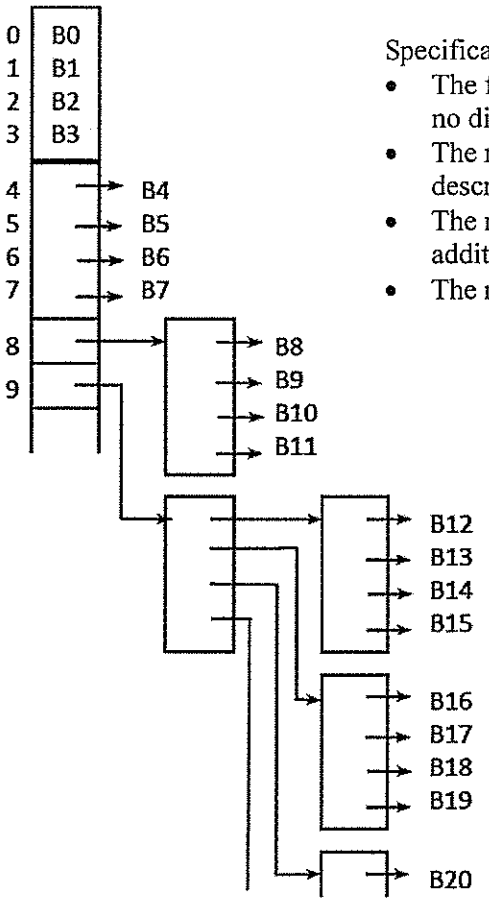
d. With the above extension, the segment table does not fit into a single frame. How many pages does it have to be divided into?

_____

e. What is the maximum size of the virtual address such that the segment directory and all page tables fit into a single frame each?

_____ (bits)

13. [12 marks] Consider an expanding file index similar to Unix but modified as shown in the figure below.



Specifically:

- The first 4 blocks (0-3) resides inside the descriptor (requiring no disk access)
- The next 4 blocks (4-7) are pointed to directly from the descriptor (requiring 1 disk access each)
- The next 4 blocks (8-11) require 1 level of indirection, i.e., 1 additional disk access
- The next 16 blocks require 2 levels of indirection, and so on

Answer the following questions for each given file size:

a. How many disk accesses are necessary to read the *entire file sequentially*
b. How many disk accesses are necessary (on average) to *seek to a random location* within the file.

	File size					
	2 blocks	7 blocks	12 blocks	15 blocks	17 blocks	29 blocks
a.						
b.						

14. [20 marks] Consider the following disk organization:
    - the disk rotates at 600 rpm
    - there are 10 blocks/track
    - all blocks are numbered sequentially starting from 0 (i.e., track 0 holds blocks 0-7, track 1 holds blocks 8-15, etc.)
    - no track skew is used
    - the average seek time to a different track is 6 ms
    - the time for any CPU calculations is negligible
    - at time 0 the disk head is at the beginning of block 12

    Requests to access the following blocks are received in the given order: **12, 13, 15, 39, 5, 41**

    a. In which **order** will the requests be serviced (starting with the current sector 12) under:

    - FIFO _____
    - SSTF (Shorted Seek Time First) _____
    - Scan (Elevator Algorithm; assume r/w head is moving up)

    _____

    b. What will be the total **distance** (number of tracks) traveled by the read/write head to service all requests (again starting from sector 12) under:

    - FIFO _____
    - SSTF _____
    - Scan _____

    c. For each request, determine the time at which the r/w head reaches the end of that block:

Block number	12	15	39	41
Time at end of block				

    d. Repeat the above question but assuming seek time of 12 ms

Block number	12	15	39	41
Time at end of block				

15. [10 marks] Consider a biometric user authentication system that generates values between 0 and 1 where 1 is a perfect match and 0 a perfect mismatch. For valid attempts, the probability of getting a value x is approximated by the following function:

$g = 0$	for x the interval [0 : 0.5)
$g = 4x^2 - 4x + 1$	for x in the interval [0.5 : 1]

For invalid (imposter) attempts, the probability of getting a value x is approximated by the following function:

$f = -2.5x + 1$	for x in the interval [0 : 0.4]
$f = 0$	for x in the interval (0.4 : 1]

a. Is it possible to set the threshold value for x such that all genuine attempts are accepted and all imposter attempts are rejected?

_____ (yes/no)

b. What should be the maximal threshold value for x such that all genuine attempts are accepted?

_____

c. What should be the minimal threshold value for x such that all imposter attempts are rejected?

_____

d. Assume the threshold value is set to 0.2. What percentage of genuine attempts will be rejected?

_____ What percentage of imposter attempts will be accepted? _____

END OF PAPER