NATIONAL UNIVERSITY OF SINGAPORE

CS2106 – INTRODUCTION TO OPERATING SYSTEMS

(Semester 1: AY2016/17)

Time Allowed: 2 Hours

INSTRUCTIONS TO CANDIDATES

1. This assessment paper consists of **FOURTEEN (14)** questions and comprises **EIGHT (8)** printed pages on **FOUR** (**4**) sheets of paper.
2. This is a **CLOSED BOOK** assessment. Two handwritten A4 reference sheets are allowed. Calculators are not allowed.
3. Answer all questions and write your answers in the **ANSWER BOOKLET** provided.
4. Fill in your Student Number with a **pen, clearly on odd-numbered pages** of your ANSWER BOOKLET.
5. You may use pencil to write your answers.
6. Marks allocated to each question are indicated. Total marks for the paper is **100**.
7. You are to submit only the **ANSWER BOOKLET** and no other document.

Questions 1 - 8: Each question has only one correct answer. Write your answers in the boxes provided in the Answer Booklet. Two marks are awarded for a correct answer and no penalty for wrong answer.

1. Given the following complete inverted page table on a system with frame size = 16 bytes, what is the translated physical address of 8-bit logical address 0110 1011_2 for process **pid 5**? Note that all values in the table are in binary.

	PID	P#
0	1110	1011
1	0101	1110
2	0101	0110
3	0011	0101

	PID	P#
4	1100	0111
5	0110	0101
6	0101	1000
7	0110	1011

   a. Physical address = 0000 1011_2
   b. Physical address = 0010 1011_2
   c. Physical address = 0111 1011_2
   d. Physical address = 1011 1011_2
   e. None of the above.

2. Which of the following statement regarding symbolic and hard link is/are TRUE?

   i. Symbolic link will use an additional i-node compared to hard link.
   ii. Symbolic link is not affected by the deletion of the linked file.
   iii. Hard link can link to both directory and file, while symbolic link is limited to files only.

   a. (i) only.
   b. (ii) only.
   c. (ii) and (iii) only.
   d. (i) and (iii) only.
   e. (i), (ii) and (iii).

3. Which of the following statement regarding semaphore is FALSE?

   a. Tasks utilizing semaphore(s) can still violate mutual exclusion.
   b. Tasks utilizing semaphore(s) can still deadlock.
   c. A task can never be blocked on two semaphores at the same time.
   d. A task should always call wait() on a semaphore before signalling it.
   e. None of the above

4. Given 3 physical frames, which of the page replacement algorithm(s) give the same number of page fault as the optimal page replacement algorithm (OPT) for the following memory reference string?

23, 26, 18, 22, 17, 24, 30, 27, 16, 19

i. LRU
ii. FIFO
iii. Second-Chance (CLOCK)

a. (i) only.
b. (ii) only.
c. (ii) and (iii) only.
d. (i) and (iii) only.
e. (i), (ii) and (iii).

5. On a system with 32-bit virtual memory space, 4KB (2^{12}) page size and 4-byte Page Table Entry (PTE) with no additional booking bits, which of the following process(es) have a lower overhead if 2-level paging is used instead of direct paging?

Note that GB = 2^{30} Bytes, MB = 2^{20} Bytes, KB = 2^{10}Bytes.

i. Process A, which uses (4GB – 3MB) memory
ii. Process B, which uses (4GB – 4MB) memory
iii. Process C, which uses (4GB – 5MB) memory

a. (i) only.
b. (ii) only.
c. (ii) and (iii) only.
d. (i) and (ii) only.
e. (i), (ii) and (iii).

6. What the minimal set of information we should keep for a zombie process in order for us to support the **`wait()`**correctly? Select the best answer that does not include irrelevant information.

a. PID, return value.
b. PID, return value, stack region.
c. PID, return value, code region.
d. PID, return value, all memory regions
e. None of the above.

7. Which of the following statement(s) regarding **batch processing scheduling** is/are TRUE?

   i. Responsiveness is less critical as compared to an interactive system.
   ii. High system utilization rate is more critical as compared to an interactive system.
   iii. Priority scheduling can be deployed on a batch processing system.

   a. (i) only.
   b. (ii) only.
   c. (ii) and (iii) only.
   d. (i) and (iii) only.
   e. (i), (ii) and (iii).

8. Given the following programs:

//compiled as **a.out**

```
int main()
{
  int i;

  for (i = 0; i < 3; i++){
    fork();
    printf("Hello World!\n");
    execl("./Echo", "Echo",
                  "Hello World!");

    printf("Hello World!\n");
  }
  return 0;
}
```

//compiled as **Echo**

```
int main( int argc,
          char**argv)
{
   printf("%s\n",argv[1]);

   return 0;
}
```

If the two programs are compiled as **`a.out`** and **`Echo`** respectively in the same directory. How many "**`Hello World!\n`**" messages will be printed if we execute "**`a.out`**"?

a. 6.
b. 9.
c. 16.
d. 24.
e. None of the above.

9. [**9 marks**] As covered in this course, operating system manages three main resources of a system: **process (CPU time), memory** and **file.** For each of these resources, use your own words to **briefly** describe how OS is designed to provide **abstraction** and **protection.**

---

10. [**14 marks**] Suppose we have the following machine setup:

    - Pure segmentation scheme, with four default segments:
      (0 = text, 1 = data, 2 = heap, 3 = stack).
    - A special set of four hardware registers known as **segment base registers (SBR):**
      `SBR0`, `SBR1`, `SBR2` and **`SBR3`.** Each **`SBR`** has two fields:
        - **`SAddr`** = point to the starting address of a segment
        - **`Limit`** = maximum valid offset of a segment.

      For simplicity, you can use array indexing syntax to use one of the **`SBRs`**, e.g. **`SBR[0]`** refers to the **`SBR0`**. So, **`SBR[1].SAddr`** refers to the starting address as stored in **`SBR1`**.

    a. [**4 marks**] Illustrate (draw) a sample process with the following running information:
       - SBR0 stores the stack region {SAddr = 64, Limit = 24}
       - SBR1 stores the data region {SAddr = 8, Limit = 16}
       - SBR2 stores the heap region {SAddr = 32, Limit = 12}
       - SBR3 stores the text region {SAddr = 44, Limit = 16}

       There is no need to worry about the scale of region (how large the region is), however, you need to indicate all addresses / limits correctly as well as the relationship between the SBRs and the regions.

    b. [**3 marks**] Briefly describe how to handle **malloc()** (dynamic memory allocation) for the sample process in (a).

    c. [**3 marks**] Do we need caching mechanism like TLB in this machine? Briefly explain.

    d. [**4 marks**] Briefly explain how **thread switching,** i.e. switching between two threads in the same process, can be supported on this machine **efficiently**. Your answer only need to focus on the memory context (i.e. ignore hardware context etc) and minimally should contain the following 2 points:

       i. What should be saved as part of the **thread context**?
       ii. Use (a) as an example to illustrate the basic ideas.

11. [**16 marks**] Given below is the enhanced version of Reader-Writer synchronization solution. Note that we added a ***DoingSomethingA()*** **and** ***DoingSomethingB()*** in the two codes to represent some other computations performed by the tasks.

```
while (TRUE) {
    DoingSomethingA( )

    wait( RevDoor );
    wait( roomEmpty );

    Modifies data

    signal( roomEmpty );
    signal( RevDoor );
}
```

Writer Process

```
RevDoor = Semaphore(1)
roomEmpty = Semaphore(1)
mutex = Semaphore(1)

nReader = 0
```

Initialization

```
while (TRUE) {
    DoingSomethingB( )
    wait( RevDoor );
    signal( RevDoor );

    wait( mutex );
    nReader++;
    if (nReader == 1)
        wait( roomEmpty );
    signal( mutex );

    Reads data

    wait( mutex );
    nReader--;
    if (nReader == 0)
        signal( roomEmpty );
    signal( mutex );
}
```

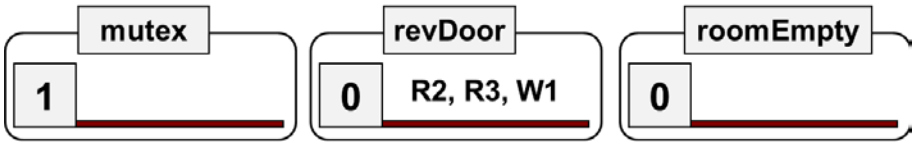
Reader Process

Suppose there are two writer processes **W1**, **W2** and three reader processes **R1**, **R2**, **R3** running. For each of the execution snapshots **in the answer sheet,** indicate:

- Whether the execution snapshot can happen i.e. legal or illegal, AND
- If the snapshot is legal, deduce the status of the missing tasks (from the original five tasks) and state any assumption.
- If the snapshot is illegal, briefly explain **why.**

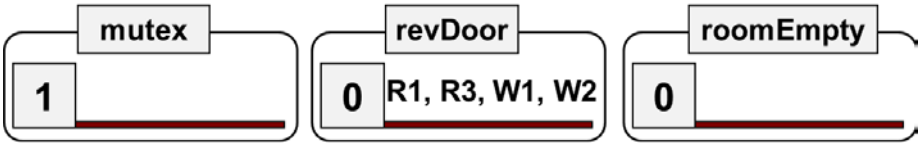
Note that the blocked queue in each semaphore is a FIFO queue, i.e. tasks are queued in the order of their **wait()** call.

For example, the following snapshot is **legal:**



Missing Tasks: **R1** should be in ***DoingSomethingB***( ) and **W2** is writing data.
Note: there may be alternative answers, you only need to give one.

The following snapshot is **illegal**:



Reason: Task **R2** passed through the ***wait*****(roomEmpty)** and signalled the **mutex** already ➔ **R2** is reading data. There is no way for **R1** and **R3** to be blocked on **RevDoor** as **W1** and **W2** arrived later than **R1** and **R3** on **RevDoor**.

12. [**14 marks**] Most OSes perform higher level I/O scheduling other than just trying to minimize hard disk seeking time. For example, one common hard disk I/O scheduling algorithm is described below:

    i. User process submit file operation requests in the form of

    ***operation*(starting hard disk sector, number of bytes)**

    ii. OS **sorts** the requests by hard disk sector.

    iii. OS **merges** requests that are nearby into a larger request, e.g. several requests asking for tens of bytes from nearby sectors merged into a request that reads several nearby sectors.

    iv. OS then issue the processed requests whenever the hard disk is ready.

    a. [**2 marks**] How should we decide whether to merge two user requests? Suggest two simple criteria.

    b. [**2 marks**] Give one advantage of the algorithm as described.

    c. [**4 marks**] Give one disadvantage of the algorithm and suggest one way to mitigate the issue.

    d. [**3 marks**] Strangely enough, OS tends to **intentionally delay** serving user Disk I/O request. Give one reason why this is actually beneficial using the algorithm in this question for illustration.

    e. [**3 marks**] On modern harddisk, algorithm to minimize disk head seek time (e.g. SCAN variants, FCFS etc) are **built into the** hardware controller. i.e. when multiple requests are received by the harddisk hardware controller, the requests will be reordered to minimize seek time. Briefly explain how the high level I/O scheduling algorithm described in this question may conflict with the harddisk built in scheduling algorithm.

---

13. [**9 marks**] In virtual memory scheme, some memory pages are stored in a **swap file** on the secondary storage. This question explore the relationship in more details.

    a. [**2 marks**] Should the swap file be handled as a normal file? Briefly explain.

    b. [**4 marks**] What is the relationship between page size and hard disk cluster size? Briefly explain.

    c. [**3 marks**] On most OS, there is only a system-wide swap file shared by all processes. Why do you think the alternative, i.e. per-process swap file, is not a good idea?

14. [**22 marks**] Given below is a tiny FAT filesystem snapshot. The file allocation table is given on the right, while the disk blocks are shown in the middle column and continues to the right column. In a disk block, the directory entries are simplified to show only: name of file / directory, a single bit to represent type (0 = file, 1 = directory) and a single number to represent the starting disk block number. Note that if the disk block belongs to a file, then the directory entries are simply "content" of that file.

FAT	
0	EOF
1	EOF
2	8
3	12
4	EOF
5	FREE
6	14
7	EOF
8	EOF
9	EOF
10	FREE
11	2
12	EOF
13	EOF
14	EOF
15	3

Disk Blocks			
0	A.txt	0	3
	B.txt	0	1
1	C.txt	1	1
	D.txt	1	5
2	E.txt	1	4
	F.txt	0	14
3	G.txt	0	11
	H.txt	1	10
4	J.txt	0	9
	K.txt	1	3
5	L.txt	0	13
	M.txt	0	0
6	N.txt	0	15
	P.txt	1	9
7	Q.txt	1	6
	R.txt	0	4

Disk Blocks			
8	T.txt	0	12
	U.txt	1	3
9	V.txt	0	0
	W.txt	0	?
10	X.txt	1	13
	Y.txt	0	12
11	Z.txt	0	7
	AA.txt	1	2
12	BB.txt	0	0
	CC.txt	0	5
13	DD.txt	1	8
	EE.txt	0	6
14	FF.txt	0	1
	GG.txt	0	13
15	HH.txt	1	8
	JJ.txt	1	2

a. [**4 marks**] Given that root directory is stored in **disk block 7**, list the directory content of the subfolder "Q.txt", i.e. similar to "ls Root / Q.txt". There is no need to list the content of any further subdirectory (if any).

b. [**4 marks**] Give the absolute path for the file "`N.txt`" and "`V.txt`".

c. [**4 marks**] Give the disk blocks **in the allocation order** for the file "`N.txt`" and "`V.txt`".

d. [**3 marks**] Suppose we want to implement **hard link** in this file system, show how you would link the file "`N.txt`" as the file "`R.txt`" (i.e. `R.txt` now points to `N.txt`). Give only the affected FAT entries and the disk block content (if any).

e. [**3 marks**] A disk scanner utility found that a minor disk corruption occurs at the entry for file "`W.txt`". The starting disk block number for that file is no longer readable (shown as "?"). What do you think is the **most likely** disk block number for this entry assuming FAT is correct?

f. [**4 marks**] Suppose a **new disk block** is added to the **beginning** of file "`N.txt`", show all affected FAT entries and the disk block content (if any). You should use the free disk block with the smallest disk number if possible.

~~~~ End of Paper ~~~~