NATIONAL UNIVERSITY OF SINGAPORE

CS2106 – INTRODUCTION TO OPERATING SYSTEMS

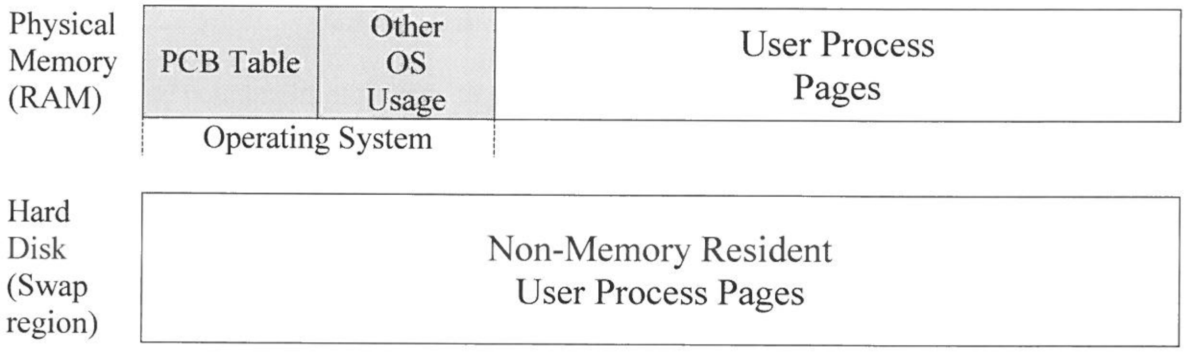
(Semester 1: AY2017/18)

Time Allowed: 2 Hours

INSTRUCTIONS TO CANDIDATES

1. This assessment paper consists of **EIGHTEEN (18)** questions and comprises **SIX (6)** printed pages on **THREE (3)** sheets of paper.
2. This is a **CLOSED BOOK** assessment. One A4 reference sheet is 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 8 - 13: Below is an illustration of virtual memory organization on a certain machine. Note that the size of each region is not to scale (i.e. the size of each region is not accurately represented by size of the boxes). **Each question is worth 3 marks.**



For ease of reference, use the following short form for each of the regions:

a. **OS-P**: OS memory region for PCB tables.
b. **OS-O**: OS memory region for other usages.
c. **UPP**: User Process Pages in physical ram.
d. **SWAP**: Non-memory resident user process pages.
e. **NIL**: Not part of the virtual memory space.

For each of the following items, indicate the region(s) it can reside. State any assumptions if necessary.

8. Page table of a process.
9. Open file table.
10. Dynamically allocated linked list nodes in a program.
11. File descriptor returned from an `open(...)` system call.
12. The code for the Process Scheduler.
13. The binary of a compiled program, e.g. `a.out`.

---

14. **[6 marks]** Give pseudo-code of a simple program to **find out the page size of a machine**. You should state clearly how to run the program and interpret the result.

15. **[9 marks]** Upon the termination of a process **P**, operating system is supposed to "clean up the resources used by **P**". Answer the following briefly:
    a. **[3 marks]** How is the OS notified of **P**'s termination?
    b. **[6 marks]** Indicate the resources to be cleaned up and how can the OS find out the relevant information to perform the cleaning. You should be as specific as possible and refer to ideas covered in this course.

16. **[9 marks]** Indicate whether the working set W(T, Δ) increases (more pages are needed), decreases (less pages are needed) or stable (no change in number of pages needed) for the following scenarios:
    a. **[3 marks]** We are executing a loop to calculate values to be stored in a **large array**.

    b. **[3 marks]** We are currently executing a **recursive function**.

    c. **[3 marks]** We are currently blocked on a critical section implemented by **busy waiting**.

    State any assumption if necessary.

17. **[12 marks]** Given the following two code fragments:

Code One	**Code Two**

`Wait( S1 )`
`A`
`B`
`Signal( S1 )`
	`Wait(` **<X>** `)`
	`C`
	`D`
	`Signal(` **<X>** `)`

Suppose each code is executed by many concurrent tasks, we want to examine whether the following execution sequence:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
A	C	B	D	C	A	D	B	A	A	C	B	B	C	D	D

is possible under the following scenario:

a. <X> is S1 (i.e. same semaphore used in Code One). S1 is a binary semaphore initialized to 1.

b. <X> is S1. S1 is a general semaphore initialized to 2.

c. <X> is S2 (i.e. a different semaphore). Both S1 and S2 are binary semaphores initialized to 1.

d. <X> is S2. Both S1 and S2 are general semaphores initialized to 2.

For each scenario, indicate the **earliest (**i.e. **leftmost)** sequence number where impossible execution is detected, e.g. if you think D (at 4) is impossible due to the semaphore restriction, indicate "4" in the answer booklet. Briefly explain your reasoning.

If there is nothing wrong with the sequence, just indicate **0** in the answer. There is no need to explain for such case(s).

18. **[18 marks]** Below is the File Allocation Table (**FAT**) and the directory entries of the **root directory** on a mysterious disk:

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

FAT (16 disk blocks)

`easy.txt`	6
`exam.c`	1
`kidding.bmp`	8

Directory Entry (for **root**)

a. **[2 marks]** Give the file size for the file "`easy.txt`" if each disk block is 1KB.

b. **[5 marks]** Convert the file "`exam.c`" into I-Node representation. Note the following:

- Use the same disk block numbers in the same sequence currently occupied by "`exam.c`".
- The I-Node structure is defined as:
  - 3 direct data block pointers
  - 1 indirect data block pointer (pointing to 3 direct pointers)
  - 1 double indirect pointer (pointing to 3 indirect pointers)
  - 1 triple indirect pointer (pointing to 3 double indirect pointers)

Show the I-Node for "`exam.c`".

c. **[5 marks]** Perform a **defragmentation** on the disk. So that:

- Files uses consecutive blocks.
- Free space (if any) are at the end of the disk.
- The order of the files is the same as the directory entries (i.e. "`easy.txt`" followed by "`exam.c`" followed by "`kidding.bmp`").

Show the FAT and directory entries after defragmentation.

d. **[3 marks]** As the OS, what is the main difficulty of performing (c) for this disk?

e. **[3 marks]** Suppose the hard disk has only 4 disk blocks (sectors) per track. After defragmentation, can we say that the file "`exam.c`" does not incur additional **rotational delay** once we located the first disk block? You can assume we allocate all disk blocks in adjacent track(s).

~~~~ **End of Paper** ~~~~
~~~~

<Blank Page>