NATIONAL UNIVERSITY OF SINGAPORE

CS2106 – Introduction to Operating Systems

(Semester 2: AY 2024/2025)

Time Allowed: 2 Hours

Sample Answers

INSTRUCTIONS TO STUDENTS

1. Please complete this exam on ExamSoft.
2. This assessment paper contains **NINETEEN (19)** questions and comprises **SEVEN (7)** printed pages (including this cover page).
3. Total marks in this question paper is **100**.
4. Students are required to answer **ALL** questions.
5. This is a **CLOSED BOOK** assessment.
6. Only one A4 sheet of paper is allowed.

Examiner's Use Only

Question No.	**Maximum Marks**
MCQs 1 – 15	**60**
16	**10**
17	**10**
18	**10**
19	**10**
Total	**100**

Question 1

Time interval from a process being submitted to the time of its completion is called

(a) Waiting Time
(b) Wasting Time
(c) Turnaround Time
(d) Response Time
(e) Throughput Time

Option (c).

Question 2

Which of the following is true about the multilevel feedback scheduling algorithm?

(a) There is one queue for all priority levels
(b) The number of queues depends on the number of processes
(c) Processes cannot move between different queues
(d) It is not very suited for real-time and multimedia apps
(e) None of the above

Option (d). Since it takes a while for this to reach steady state.

Question 3

What is a race condition?

(a) Two threads racing to find out who is the fastest
(b) Many threads read and write shared data – result is based on thread interleaving
(c) Many threads access shared data – who is the first to acquire it
(d) Threads outside a critical section race to get access to the critical section
(e) None of the listed

Option (b).

Question 4

Consider a 16-bit virtual memory address and a page size of 8 KB. How many pages can a process potentially have?

(a) 8
(b) 2
(c) 10
(d) 4
(e) 6

Option (a). Total address/page size = 2^{16} / 2^{13} = 8.

Question 5

Consider the pseudo-code of three concurrent processes P0, P1, and P2 shown below. S0, S1 and S2 are three binary semaphores that are initialised as S0 = 0, S1 = 1, S2 = 0. **Note**: For the binary semaphore in this case, it is valid to increment the semaphore twice, and it will increment to 1 and remain there for any further `up` operations.

```
Process P0                 Process P1            Process P2
{                          {                     {
  while (true) {             down (S0);            down (S2);
    down (S1);               up (S1);              up (S1);
    print 'exam';          }                     }
    up (S0);
    up (S2);
  }
}
```

How many times will process P0 print the string *exam*?

(a) 0 or 1
(b) 1 or 2
(c) 2 or 3
(d) 3 or 4
(e) 4 or 5

Option (c**).** At least twice and at-most thrice.

P0 completes loop, prints `exam,` and waits on S1.

P1 runs and wakes P0 so P0 prints `exam`

P2 runs and wakes P0 so P0 prints `exam` i.e. `exam`, `exam`, `exam`. However, if P2 runs immediately after P1, then S1 will still be 1, so P0 runs only once more and prints `exam`, `exam`.

Question 6

In a system using virtual memory , increasing the RAM size usually helps with performance, why?

(a) Increase in virtual memory size
(b) Faster physical memory
(c) Not as many memory segmentation faults
(d) Reduced Thrashing
(e) Fewer memory leaks

Option (d). Increasing RAM leads to more physical pages being available, which reduces the number of page faults and hence reduced thrashing.

Question 7

Consider a virtual memory system that uses 32-bit virtual address, 8-bit page offset and a 3-level paging system. The address is broken up as follows:

| 10-bit | 8-bit | 6-bit | 8-bit |

What is the memory footprint (in bytes) of page directories/tables for a process with a size of 256 KB?

(a) 2560
(b) 4608
(c) 4096
(d) 1024
(e) None of the above

Option (b). Process has 2^{10} pages (256 KB/256). Number of inner-level pages required $= 2^{10}/2^6 = 2^4$. Number of 2nd-level pages required $= 2^4/2^8 = 1$ and number of third-level pages is 1. Total size $= (1 * 2^{10} + 1 * 2^8 + 2^4 * 2^6) * 2 = 4608$ bytes.

Question 8

Which of the following statements is true?

(a) Paging does not suffer from internal fragmentation
(b) Segmentation does not suffer from external fragmentation
(c) In UNIX, symbolic links are allowed to link to a directory, thus possibly creating loops
(d) The TLB must always be fully flushed when a new file is opened
(e) The maximum partition size managed by the FAT file system is 32 GB

Option (c).

Question 9

Consider a file system that uses i-nodes. Suppose that, for a given file, all the blocks stemming from the doubly indirect pointers are filled up. Assume that the i-node and free block bitmap are both completely in memory. How many disk accesses will it take to write one more byte to the file?

(a) 0
(b) 1
(c) 2
(d) 3
(e) 4

Option (e). No disk access for allocating 4 necessary blocks as free block bitmap is in main memory. Accessing triple indirect pointer is free as i-node is in memory. You then need to get 1st block of indirect pointers, then a 2nd block and then the 3rd block, so

3 in total. Finally, you need to write the one byte to the disk ➔ total 4.

Question 10

Which of the following does the OS not have to do on a context switch?

(a) Flush some entries from the TLB
(b) Load the page table for the process that is to start running
(c) Set the page table base register to point to the correct page table
(d) Retain the page table of the process to be swapped out in memory
(e) None of the above

Option (d) is not necessary always. (a) – (c) are all done by the OS on a context switch.

Question 11

What is the access time for a four-level paging system assuming a TLB hit ratio of 80%, TLB access time of 20 ns and a memory access time of 100 ns? You can assume the page directory and all the inner pages are in memory.

(a) 100 ns
(b) 200 ns
(c) 180 ns
(d) 196 ns
(e) None of the above

Option (b). Four-level paging TLB miss: 20 (TLB) + 100 (level 1 page-table) + 100 (level 2) + 100 (level 3) + 100 (level 4) + 100 (page from memory) = 520 ns. Hence, access time = 0.8 * 120 + 0.2 * 520 = 200 ns.

Question 12

What is the internal fragmentation (in bytes) on a system with page size of 2 KB when a process is of size 72,766 bytes?

(a) 2048
(b) 1086
(c) 962
(d) 1024
(e) None of the above

Option (c). Bytes left-over in last page = 72766 % 2048 = 1086. Internal fragmentation = 2048 – 1086 = 962 bytes.

Question 13

Consider the pseudo code for the following three threads, which share variables a and b

T1	T2	T3

a = 1;
b = 2;
	b = 1;	a = 2;

What is the probability to have a = 2 and b = 2 after all threads complete execution?

(a) 0.58
(b) 0.5
(c) 0.33
(d) 0.41
(e) None of the above

Option (d). In total we have 12 variations of possible thread execution

a = 1; b = 2; b = 1; a = 2; ➔ (a = 2, b = 1)

a = 1; b = 2; a = 2; b = 1; ➔ (a = 2, b = 1)

a = 1; b = 1; b = 2; a = 2; ➔ (a = 2, b = 2)

a = 1; a = 2; b = 2; b = 1; ➔ (a = 2, b = 1)

a = 1; b = 1; a = 2; b = 2; ➔ (a = 2, b = 2)

a = 1; a = 2; b = 1; b = 2; ➔ (a = 2, b = 2)

b = 1; a = 1; b = 2; a = 2; ➔ (a = 2, b = 2)

a = 2; a = 1; b = 2; b = 1; ➔ (a = 1, b = 1)

b = 1; a = 1; a = 2; b = 2; ➔ (a = 2, b = 2)

a = 2; a = 1; b = 1; b = 2; ➔ (a = 1, b = 2)

b = 1; a = 2; a = 1; b = 2; ➔ (a = 1, b = 2)

a = 2; b = 1; a = 1; b = 2; ➔ (a = 1, b = 2)

5 of the possible thread executions produce a = 2, b = 2 ➔ 5/12 = 0.41

Question 14

Which of the following is true?

(a) Interactive systems use non-preemptive scheduling
(b) Turnaround times in preemptive systems are more predictable
(c) A weakness of priority scheduling is it may lead to process starvation
(d) If all processes are I/O-bound, ready queue will almost always be full
(e) Preemptive scheduling suspends a running process before its time slice expires

Option (c).

Question 15

In a particular OS, an i-node contains 6 direct pointers, 1 single indirect block, 1 doubly indirect block, and 1 triply indirect block. Each of these pointers is 8 bytes long. Assume a disk block is 1024 bytes and that each indirect block fills a single block. What is the maximum file size in such a file system?

(a) Approx 1 GB
(b) Approx 2 GB
(c) Approx 3 GB
(d) Approx 4 GB
(e) Not enough information to calculate this

Option (b). Amount of data that can be accessed is: 6 x 1024 (data directly indexed) + 128 x 1024 (data referenced by single indirect) + 128^2 x 1024 (data referenced by double indirect) + 128^3 x 1024 (data referenced by triple indirect) = 2.02 GB.

Question 16

Assume a memory access reference string for the following page numbers: 1, 2, 3, 4, 6, 2, 5, 1, 4, 3, 6, 1, 3, 5. Assuming that the number of memory frames is 3, calculate the number of page faults for the LRU and Clock page replacement algorithms. You do not need to show your working.

(10 marks)

Using LRU (13 faults)

1	2	3	4	6	2	5	1	4	3	6	1	3	5
1	1	1	4	4	4	5	5	5	3	3	3	3	5
	2	2	2	6	6	6	1	1	1	6	6	6	6
		3	3	3	2	2	2	4	4	4	1	1	1
Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y

Using Clock (13 faults)

1	2	3	4	6	2	5	1	4	3	6	1	3	5
1	1	1	4	4	4	5	5	5	3	3	3	3	3
	2	2	2	6	6	6	1	1	1	6	6	6	5
		3	3	3	2	2	2	4	4	4	1	1	1
Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y

Question 17

Consider a disk with a block size of 1024 bytes with disk addresses being 4 bytes. How many block reads are needed to access the 510100th data byte, if the file system uses

(a) Basic Linked List (5 marks)

(b) I-Node with 6 direct pointers, 1 single indirect pointer and 1 double indirect pointer. Assume I-Node is not in memory and that each indirect block fills a single block. (5 marks)

You do not need to show your working.

Answer:

(a) Basic Linked List – there are 1020 data bytes per block. The 510100th data byte is in the 501st disk block (500 x 1020 = 510000) $\rightarrow$ 501 block reads.

(b) I-Node:
The 510100th data byte is in the 499th data block (510100/1024). This means that the pointer to this block is in the double indirect pointers. So, 1 disk read for I-Node, 2 for the block containing the pointer and 1 more for the data. Total of 4 block reads.

Question 18

Consider a virtual memory system that uses paging using a one-level page table. The maximum size of the virtual address space is 32 MB. The page table for a running process is provided below.

Virtual Page	Physical Page
0	5
1	11
2	16
3	23
4	41
5	20
6	12

Assuming the page size is 2 KB, and the maximum physical memory size of the machine is 2 MB, answer the questions below. You do not need to show your working.

(10 marks)

(a) How many bits are required for each frame number?	10 bits
(b) What is the maximum number of entries in a page table?	2^{14}
(c) How many bits are there in a virtual address?	25
(d) To which physical address will the virtual address 5090 translate to?	33762
(e) Which virtual address will translate to physical address 41063?	10343

(a) There can be at most 2^^{21}/ 2^^{11} physical pages in main memory, so a page entry will require 10 bits for a physical page number

(b) Max entries in page table is when a process uses all pages, $2^{25} / 2^{11} = 2^{14}$

(c) Virtual address has 25 bits (32 MB = 2^{25})

(d) Virtual address 1524 is on page 2 which lies in physical page 16. Offset is 5090 % 2048, or 994. So, 5090 maps to physical address 16 * 2048 + 994 = 33762

(e) Physical address 41063 is on physical page 20, which is mapped to by virtual page 5. So, 41063 is from virtual address 5 * 2048 + 103 = 10343

Question 19

The following are all True/False questions, so you only need to state True or False.

(10 marks)

(a) The terms *process* and *program* can be used interchangeably
(b) The FCFS scheduling algorithm works very well when tasks are I/O-bound
(c) Based on the lectures, a semaphore variable can never have a negative value
(d) The FIFO page replacement algorithm always suffers from Belady's anomaly
(e) When several threads in a process access a shared variable, mutual exclusion must be ensured
(f) Each Kernel-level thread has its own stack, while user-level threads do not
(g) Contiguous file allocation can lead to disk fragmentation
(h) When using virtual memory, there are no limits to how large a process can be
(i) User-level threads cannot be scheduled by the Kernel
(j) Using Bitmaps for free space management will always take up less space than when using Linked Lists for free space management

Answers:

(a) False, as a process is an instance of a running program
(b) False, as for when a process does I/O, the CPU is idle and FCFS does not switch to the next process, leading to low CPU utilisation
(c) True
(d) False, only in certain cases
(e) False. If all are reading, then no need for mutual exclusion
(f) False
(g) True
(h) False, the length of the virtual memory address limits process size
(i) True, as Kernel is not aware of them
(j) False, as Bitmap size is constant while for linked list, it varies and the more full the disk is, the less space it takes to maintain information about free blocks

– END OF PAPER –