

NATIONAL UNIVERSITY OF SINGAPORE
SCHOOL OF COMPUTING

FINAL ASSESSMENT PAPER

AY2020/21 Semester 1

CS2106 – INTRODUCTION TO OPERATING SYSTEMS

November 2020

Time Allowed: **2 hours**

INSTRUCTIONS

1. Submit by **Wed, 25 Nov, 7:30pm** on **Luminus Files->Student Submissions -> Exam** a PDF file:
 - Name your PDF file using your student number: A1234567Z.pdf
 - The format of the answers has to **strictly** follow the format provided in the question.PDF file. **For example, if Question 23 (requirement and answer space) is on page 8 in the questions PDF file, it should be on page 8 in the answers PDF file submitted by you.**
 - In case you scan your answers, make sure that they can be read once included in the PDF. We will not grade answers that cannot be read.
 - When multiple submissions are made, we will grade only your most recent submission.

If Luminus is not accessible, submit your PDF using this form: <https://forms.gle/fv9U4yLidycwsWya6>

2. To solve the questions. You can either:
 - write your answers on the PDF directly,
 - print out, hand-write your answers, and scan into a PDF
 - hand-write your answers on clean paper, and scan into a PDF
 - write your answers in the DOCX, and save as PDF
 - have any other approach that would result in a PDF file
 - if you scan your answers, you may use Drive application on Android to add a new document, and choose Scan.
3. This assessment paper contains 42 questions and comprises FIFTEEN (15) pages. The total number of marks in the paper is 80 and the paper counts towards 40% of your final grade.
4. Students are required to answer **ALL** questions within the space in this file.
5. Failing to submit your answers to Luminus folder means you failed your exam (0 marks will be awarded). No late submission is allowed.
6. Address any questions you might have in Zoom chat or call 65168850.
7. Write your student number below. Do not write your name.

STUDENT NO: _____

Section 1: MCQ questions.

1. [2 marks] How many times will message "All good!" be printed by the following code fragment (assume `ls` is located in `/bin`):

Line#	Code Snippets
1	<code>pid = fork();</code>
2	<code>if (pid == 0)</code>
3	<code> execl("/bin/ls", "ls", "-l", NULL);</code>
4	<code>printf("All good!\n");</code>

- A. 1
- B. 2
- C. 4
- D. "All good!" is never printed.
- E. The code snippet will produce a compilation error (when part of a program).

Answer:

2. [2 marks] Assume you are trying to create a shared memory space to be used by two processes. `shm_open` is failing because you have too many files open by the process. What can you do to make `shm_open` call succeed (choose the least invasive option)?
- A. Close a file before calling `shm_open`.
 - B. Once another process closes a file, `shm_open` will succeed.
 - C. Use `shm_unlink` to destroy a shared memory object.
 - D. Reboot the system.
 - E. Reinstall your operating system.

Answer:

3. [2 marks] Assume you are trying to create a process using `fork`. `fork` call is failing because you have too many orphan processes in the system. What can you do (choose the least invasive option)?
- A. Run a command in the interpreter to terminate a child process of process `init`.
 - B. Terminate all zombie processes.
 - C. Write a script to terminate an orphan process.
 - D. Reboot the system.
 - E. Reinstall your operating system.

Answer:

4. [2 marks] In a 32-bit system, a program is calling `malloc()` in a loop to allocate chunks of 1MiB (2^{20} bytes). Assume byte addressing. After how many loops will `malloc()` return NULL?
- A. `malloc()` will never return NULL because the space for heap can grow indefinitely.
 - B. `malloc()` will never return NULL because pages are added to accommodate the need for more memory space.
 - C. `malloc()` will never return NULL because `mmap()` is always successful.
 - D. `malloc()` will return NULL before 4096 calls.
 - E. It depends on how many other processes run at the same time in the system.

Answer:

5. [2 marks] A computer supports 64-bit virtual memory addresses. The page size is 1KiB (2^{10} bytes), the size of a page table entry is two bytes. Assuming the addressing is at the word (8 bytes) level, calculate the size of the standard page table (in bytes).
- A. 2^{58}
 - B. 2^{57}
 - C. 2^{54}
 - D. 2^{55}
 - E. 2^{52}

Answer:

6. [2 marks] A computer supports 32-bit virtual memory addresses. The page size is 1KiB (2^{10} bytes), the size of a page table entry is four bytes. Assuming the addressing is at the byte level, calculate the overhead incurred by paging when using 2-level paging in a process that uses only three pages (in bytes).
- A. $2^9 * 3 + 2^{16}$
 - B. $2^8 + 2^{14}$
 - C. $2^9 + 2^2$
 - D. $2^{10} * 3 + 2^{16}$
 - E. $2^{10} + 2^{16}$

Answer:

7. [2 marks] Consider a virtual memory system with the page table stored in memory (memory resident). If a memory access takes 200 nanoseconds, what is the longest time that a **paged memory reference** takes?
- A. 600 ns
 - B. 600 ns + time to service a page fault
 - C. 400 ns + time to service a page fault
 - D. 200 ns + time to service a page fault
 - E. More than 600 ns + time to service a page fault.

*time to service a page fault includes the time to bring the page from SWAP to memory.

Answer:

8. [2 marks] The page table base hardware register points to the page table of the process currently running on the CPU. This register needs to be updated when _____ takes place.

Answer:

9. [2 marks] Given 5 physical frames for a process, and LRU (least recently used) page replacement algorithm. After the following sequence of page accesses (memory reference string), what are the pages in RAM sorted from first frame to last frame?

Sequence: 3, 2, 4, 5, 1, 5, 7, 4, 7, 6, 3, 5

- A. 3, 4, 5, 6, 7
- B. 7, 6, 4, 3, 5
- C. 5, 4, 6, 7, 3
- D. 7, 6, 3, 5, 4
- E. 7, 6, 4, 5, 3

Answer:

10. [2 marks] Given 5 physical frames for a process, and FIFO (first in first out) page replacement algorithm. After the following sequence of page accesses (memory reference string), what are the pages in RAM sorted from first frame to last frame?

Sequence: 3, 2, 4, 5, 1, 5, 7, 4, 7, 6, 3, 5

- A. 3, 7, 6, 5, 1
- B. 1, 3, 5, 6, 7
- C. 7, 6, 3, 5, 1
- D. 5, 1, 7, 6, 3
- E. 7, 6, 4, 5, 3

Answer:

11. [2 marks] Given 5 physical frames, which of the page replacement algorithms give the same number of page faults as the optimal page replacement algorithm (OPT) for the following memory reference string (sequence)?

Sequence: 3, 2, 4, 5, 1, 5, 7, 4, 7, 6, 3, 5

- i. LRU
 - ii. FIFO
 - iii. CLOCK
- A. iii.
 - B. i. and ii.
 - C. ii. and iii.
 - D. i. and iii.
 - E. None of the above.
 - F. All of the above.

Answer:

12. [2 marks] A program maps a file to memory using `mmap`. Consider the following memory regions:
- i. Frames in RAM.
 - ii. SWAP.
 - iii. Disk blocks.

Choose where the content of file might be located during the execution of the program:

- A. i., iii.
- B. iii.
- C. ii, iii.
- D. i, ii.
- E. All of the above.

Answer:

13. [2 marks] The code handling a `fork()` system call is stored in the following memory region:
- A. OS memory region.
 - B. Virtual memory space of the program that calls `fork()`.
 - C. Frames of the process that calls `fork()`.

Answer:

14. [2 marks] A process is generating a page fault while accessing a memory region for which it does not have access rights. What happens in this case?
- A. The operating system will handle the page fault and the process can continue the execution with the next instructions.
 - B. A signal `SIGSEGV` is sent to the process. If the operating system does not have a signal handler set for `SIGSEGV`, the process is terminated.
 - C. A signal `SIGSEGV` is sent to the process. The process might terminate.
 - D. A signal `SIGSEGV` is sent to the operating system. If the operating system does not have a signal handler set for `SIGSEGV`, the process is terminated.
 - E. The operating system will handle the page fault, and the process retries the instruction that caused the page fault.

Answer:

15. [2 marks] Throughout their execution, parent and child processes share ALL entries in the system-wide open file table, but different file descriptor tables.
- A. True
 - B. False

Answer:

16. [2 marks] When opening a file using `open`, the existing entry in the system-wide open file table is returned. If no such entry exists, a new entry is returned.
- A. True
 - B. False

Answer:

17. [2 marks] A process is started using an executable file. Immediately after starting, the physical memory occupied by this process is 8 KiB, even though the executable program's size is 630KiB (1kiB is 2^{10} B). This is observed because the operating system is using _____

Answer:

18. [2 marks] A symbolic link SL to a file `/home/e1234567/B/A` (given with full path) is created in the current path. SL is moved to the parent folder.
- A. SL becomes invalid.
 - B. SL is valid, but it does not point to the correct file A.
 - C. SL cannot be moved.
 - D. When SL is moved, file A moves too.
 - E. SL is valid and A can be accessed using SL.

Answer:

19. [2 marks] A process creates multiple threads (using `pthread` library); next, each thread opens the same file and reads 32 bytes from the file (using `read` system call). What will each thread read?
- A. Same information (same bytes).
 - B. Different parts of the file, depending on the order the threads get to read.
 - C. There is a race condition because the threads read from the same file.
 - D. If the file is open for reading by another process, it is impossible to say.

Answer:

20. [2 marks] A file `A.txt` uses 14 data blocks on a EXT2 file system. Assume you have already read the directory entry for `A.txt` and only the I-node for `A.txt` is cached in RAM. What is the number of accesses to disk/RAM do you need to make to read the whole file?
- A. 14 accesses to disk and 14 accesses to RAM.
 - B. 14 accesses to disk and 13 access to RAM.
 - C. 16 accesses to disk and 15 accesses to RAM.
 - D. 16 accesses to disk and 14 accesses to RAM.
 - E. 17 accesses to disk and 14 accesses to RAM.

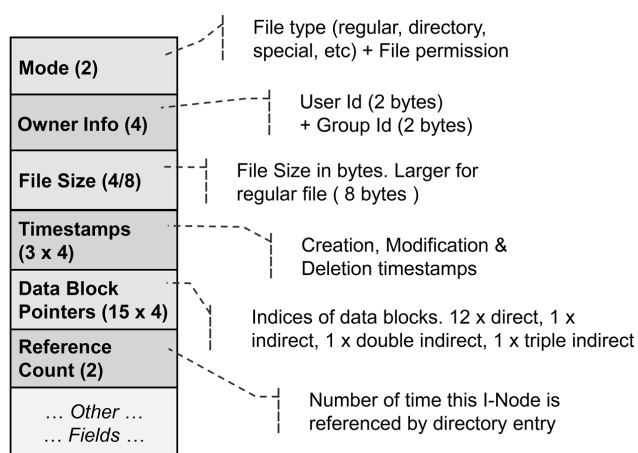
Answer:

21. [2 marks] A process opens twice the same file. Assume the process is not opening or closing any files during its execution. The **total number of entries** added by the process to the system-wide open file table is:
- 1
 - 2
 - 5
 - Impossible to give a number.

Answer:

22. [2 marks] According to the lecture, i-node in ext2 file system contains the following information:

Ext2: I-Node Structure (128 Bytes)



Consider system call `ftruncate`, with the following description:

`int ftruncate (int fd, off_t length)`

Description: **ftruncate()** cause the regular file referenced by `fd` to be truncated to a size of precisely `length` bytes. If the size of the file previously exceeded `length`, the extra data shall no longer be available to reads on the file. If the file previously was smaller than this size, `ftruncate()` shall increase the size of the file. If the file size is increased, the extended area shall appear as if it were zero-filled. The value of the seek pointer shall not be modified by a call to `ftruncate()`.

Consider the following statements:

- ftruncate** call will modify exactly one i-node field.
- ftruncate** call might modify the data blocks pointers in the i-node.
- ftruncate** call will modify the entry in the per-process file descriptors table for the process that calls it.
- ftruncate** call will modify the entry in the system-wide open files table.

Which statements are TRUE?

- i. and ii.
- i., iii. and iv.
- ii.
- ii. and iv.
- None of the above.

Answer:

23. [2 marks] In a program, the following call is used:

```
fd = open("/home/e1234567/myfile", O_RDONLY);
```

The call is successful and myfile is a hard link. How many I-nodes have been read by this call?

- A. 1
- B. 3
- C. 4
- D. 5
- E. Impossible to say.

Answer:

24. [2 marks] Consider the following code:

Line#	Code snippets
1	int t = 0;
2	int thread()
3	{
4	int *s;
5	s = &t;
6	(*s)++;
7	}
8	int main()
9	{
10	pthread_t t1, t2;
11	pthread_create(&t1, NULL, thread, NULL);
12	pthread_create(&t2, NULL, thread, NULL);
13	t++;
14	printf("%d", t);
15	}

Assume the code is part of a program that compiles and runs successfully. The code will:

- A. Have a race condition and we cannot predict what it will print
- B. Always print 3, because the threads share the data section of the memory space.
- C. Always print 2, because the threads have independent stack space.
- D. Always print 1, because the global variables are not affected by threads' execution.
- E. None of the options.

Answer:

Section 3. Short Questions

Q25-27: Answer questions 25-27 based on the following description:

A process has 3 threads, T1, T2, and T3, and its code does not contain any `exec*()` commands. T1 opens 3 files and T2 creates a pipe. After these actions, T1 executes a `fork()` command and T2 executes a `fork()` whose child immediately calls `execvp()` to execute a single-threaded program. T1 closes the three files and then terminates. T3 terminates without opening any file. Note that all threads are POSIX threads.

25. [2 marks] How many different processes execute? Why?

Assume no other processes have been created without being mentioned in the problem description.

a. Answer: _____ processes

b. Explanation: _____

26. [1 mark] How many different programs are being executed? Why?

Assume no other programs have been called without being mentioned in the problem description.

a. Answer: _____ programs

27. [3 marks] How many file descriptors have been created (opened) by this program? Why?

Assume no other files have been opened without being mentioned in the problem description. Each process has `STDIN`, `STDOUT`, `STDERR`.

a. Answer: _____ file descriptors (in the file descriptors table)

b. Explanation: _____

Section 3. Synchronization

Q28-38: Answer the questions 28-38 based on the following problem description and pseudo-code (questions are independent)

Barbershop Problem

A barbershop consists of a waiting room with n chairs, and the barber room containing the barber chair. If there are no customers to be served, the barber goes to sleep. If a customer enters the barbershop and all chairs are occupied, then the customer leaves the shop. If the barber is busy, but chairs are available, then the customer sits in one of the free chairs. If the barber is asleep, the customer wakes up the barber. Customers and barber are modeled using threads.

There are multiple Customer threads that invoke a function named `getHairCut()`. If a customer thread arrives when the shop is full, it can invoke `exit()`.

There is one barber thread serving Customers. This thread invokes `cutHair()`. When the barber invokes `cutHair()` there should be exactly one thread invoking `getHairCut` concurrently.

A program in pseudo-code modeling this problem follows:

Line#	Initialization	
1	<code>customers = 0</code>	<code>#count the customers</code>
2	<code>mutex = Semaphore (1)</code>	<code>#protect access to counter</code>
3	<code>customer = Semaphore (0)</code>	<code>#wait for customer</code>
4	<code>barber = Semaphore (0)</code>	<code>#wait for barber</code>
5	<code>customerDone = Semaphore (0)</code>	<code>#customer is done</code>
6	<code>barberDone = Semaphore (0)</code>	<code>#barber is done</code>

Line#	Customer Pseudo-code	Line#	Barber Pseudo-code
1	<code>wait(mutex);</code>	21	
2	<code>if (customers == n) {</code>	22	
3	<code> signal(mutex);</code>	23	
4	<code> exit();</code>	24	
5	<code>}</code>	25	
6	<code>customers += 1;</code>	26	
7	<code>signal(customer);</code>	27	
8		28	<code>while (TRUE) {</code>
9	<code>wait(barber);</code>	29	<code> wait(customer);</code>
10	<code>signal(customerDone);</code>	30	<code> signal(barber);</code>
11		31	
12	<code>getHairCut ();</code>	32	<code> cutHair();</code>
13		33	
14	<code>signal(customerDone);</code>	34	<code> wait(customerDone);</code>
15	<code>wait (barberDone);</code>	35	<code> signal(barberDone);</code>
16		36	<code>}</code>
17	<code>wait(mutex);</code>	37	
18	<code>customers -= 1;</code>	38	
19	<code>signal(mutex);</code>	39	

28. [1 mark] The code works according to the requirements and there is no synchronization problem.

- A. TRUE
- B. FALSE

Answer:

29. [1 mark] Assume the code is currently synchronizing correctly (either because the given code is correct, or you can make it correct with minor changes). Line 1 can be moved before line 6 without affecting the correctness of the code.

- A. TRUE
- B. FALSE

Answer:

30. [1 mark] Assume the code is currently synchronizing correctly (either because the given code is correct, or you can make it correct with minor changes). Line 3 can be removed without affecting the correctness of the code.

- A. TRUE
- B. FALSE

Answer:

31. [1 mark] The code will deadlock.

- A. TRUE
- B. FALSE

Answer:

32. [1 mark] Swapping lines 9 and 10 will make the code deadlock-free (the rest of the code stays the same).

- A. TRUE
- B. FALSE

Answer:

33. [1 mark] Swapping lines 29 and 30 will make the code deadlock-free (the rest of the code stays the same).

- A. TRUE
- B. FALSE

Answer:

34. [1 mark] Swapping lines 14 and 15 will make the code deadlock-free (the rest of the code stays the same).

- A. TRUE
- B. FALSE

Answer:

35. [1 mark] Swapping lines 34 and 35 will make the code deadlock-free (the rest of the code stays the same).

- A. TRUE
- B. FALSE

Answer:

36. [1 mark] Assume the code is synchronizing correctly (either because the given code is correct, or you can make it correct with minor changes). Assume that each thread is given equal opportunities to run (i.e. scheduling is fair). The code suffers of starvation.

- A. TRUE
- B. FALSE

Answer:

37. [1 mark] Assume the code is synchronizing correctly (either because the given code is correct, or you can make it correct with minor changes). The barber is busy waiting.

- A. TRUE
- B. FALSE

Answer:

38. [1 mark] Assume the code is synchronizing correctly (either because the given code is correct, or you can make it correct with minor changes). Choose the statement that is TRUE:

- A. The customers are served in the order they arrive because scheduling is fair.
- B. The customers are served in the order they arrive because the semaphore is using a FIFO queue.
- C. To serve the customers in the order they arrive, a queue of customers can be used.
- D. To serve the customers in the order they arrive, a queue of semaphores can be used.
- E. One more semaphore can be used to ensure that customers are served in the order they arrive.

Answer:

Section 4. Essay

39. [4 marks] You are working on implementing a special purpose dynamic allocator. Suppose all allocations and deallocations of objects follow a particular pattern characterized by two properties:

- Allocation and deallocation happen in a strict LIFO fashion, i.e., any more recently allocated object will be freed before, or simultaneously with, a less recently allocated object.
- Objects will be freed in groups of one or more objects without any intervening computation. For an example, consider the following memory allocation pattern, where [compute] denotes sections of computation, A_i denotes allocation of object i , and $F\{j, k\}$ denotes deallocation of objects j and k :

A_1 , [compute], A_2 , [compute], A_3 , [compute], A_4 , [compute], $F\{3,4\}$, [compute], A_5 , [compute], A_6 , [compute], $F\{6,5,2,1\}$

Describe how would you implement an optimized allocator that exploits this application's particular allocation pattern. In your description, touch on the following points:

- How would you maintain the information about the allocated objects (i.e. what type of data structure)? (2 marks)
- How should the allocation and deallocation functions be implemented?
There is no need to use pseudo-code, a simple explanation would do. (2 marks)
- Any other assumptions you are making about the allocated space.

Note that the number and size of allocated objects are not known beforehand, only the allocation pattern is.

Answer:

40. [3 marks] What would be the fastest way to **transfer** a large amount of data between two processes running on the same operating system? Explain your answer.

Note: The information does not need to be copied during the transfer.

Answer:

41. [2 marks] In general `exec()` syscall takes longer time to execute than `fork()` syscall. Briefly explain why.

Answer:

42. [6 marks] A programmer observes that many file blocks pertaining to different files have identical data content in an EXT2 file system. The programmer wants to implement a mechanism that would reduce the space occupied by such blocks on the disk. You are tasked to help this programmer to define a new system call `fork_file()` that replicates the file without allocating new blocks on disk.
- a. Define the parameters, return value and give a description for `fork_file()`. The description must explain how to use the new system call and what it returns. (2 marks)
 - b. What type of mechanism would you implement to allow for the files to be written with different information? (i.e. not all blocks will be different after write). Avoid duplication of blocks that are still the same. (2 marks)
 - c. Would the current I-node structure and EXT2 filesystem allow you to implement your write mechanism? If not, explain the changes that are needed in EXT2 to support the solution proposed at point b. (2 marks)

Answer:

—End of paper—