McGill

December 2018
Final Examination

OPERATING SYSTEMS

ECSE 427 / COMP 310 & SECTION 01

DECEMBER 11, 2018 & 2:00 PM

EXAMINER: Muthucumaru Maheswaran ASSOC. EXAMINER: Clark Verbrugge

STUDENT NAME:	Answers	**McGILL ID:**									

INSTRUCTIONS:

EXAM:	CLOSED BOOK ☒ OPEN BOOK ☐
	SINGLE-SIDED ☐ PRINTED ON BOTH SIDES OF THE PAGE ☒
	MULTIPLE CHOICE ☐ NOTE: The Examination Security Monitor Program detects pairs of students with unusually similar answer patterns on multiple-choice exams. Data generated by this program can be used as admissible evidence, either to initiate or corroborate an investigation or a charge of cheating under Section 16 of the Code of Student Conduct and Disciplinary Procedures. ANSWER IN BOOKLET ☐ EXTRA BOOKLETS PERMITTED: YES ☐ NO ☐ ANSWER ON EXAM ☒
	SHOULD THE EXAM BE: RETURNED ☒ KEPT BY STUDENT ☐
CRIB SHEETS:	NOT PERMITTED ☒ PERMITTED ☐ e.g. one 8 1/2X11 handwritten double-sided sheet Specifications:
DICTIONARIES:	TRANSLATION ONLY ☒ REGULAR ☒ NONE ☒
CALCULATORS:	NOT PERMITTED ☐ PERMITTED (Non-Programmable) ☒
ANY SPECIAL INSTRUCTIONS: e.g. molecular models	

Part I – Short Answers. Provide the answers in the allotted space. **Answer all questions**. **Each question is worth 3 points**.

VERY IMPORTANT: Provide precise and up to the point answers. Long and rambling answers will receive 0 marks.

1. Chrome web browser uses processes for each tab within a browser. Why is it using processes instead of kernel-level threads? A word-processing software is using kernel-level threads for the different activities inside the system. That is, the word processor creates different threads for rendering, printing, spell checking, etc. Why is the word processor preferring to employ threads over processes?

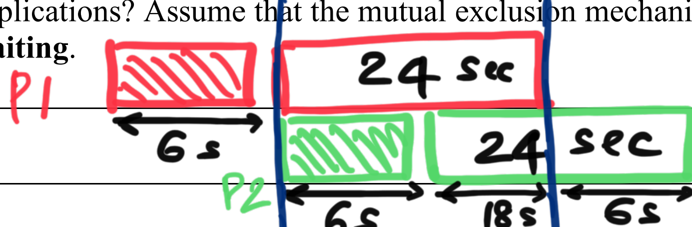
CHROME: Needs strong isolation and has no data sharing between tabs.

WORD PROC: Needs data sharing between threads. Not much in isolation. Need to engage available cores and run threads pre-emptively.

2. State Amdahl's law in words. **Note**: Amdahl's law is not the speed up equation.

Performance improvement we can expect from an enhancement is limited by the fraction of the time the enhancement can be applied to the problem.

3. Consider an application that takes 30 seconds to run. It has a critical section at the beginning of the program where it is manipulating variables in a shared memory that is accessible for all programs of that type. That is, all instances of that application share that memory region. The critical region is 20% of the program execution time. Suppose we start two instances of the application and the second one is launched 6 seconds after the first instance. The computer is not running anything else besides these applications. It has a **single core** and a **fair scheduler with zero scheduling overhead**. What is the completion time of both applications? Assume that the mutual exclusion mechanism is implemented using **busy waiting**.



$6+6+2\times 24 = 60\,s$

Since P2 arrives 6s later no waiting.

4. Suppose you modify the program to use a mutual exclusion mechanism based on sleep-wakeup. What will be completion time of both applications? Assume sleep wakeup has an overhead of 0.5 seconds.

Both answers accepted

$30+30+0.5+0.5 = 61$ sec ⇒ Taking into account the LOCK-CHECK overhead.

$30+30 = 60$ sec

↳ If 0.5s was considered only if SLEEP/WAKE was ever called.

10

5. Consider the following file **/users/diamond/research/clouds/models.pdf** in a UNIX file system. Suppose the root directory is already loaded into the memory. Determine the number of disk accesses that are necessary to access the first block in the above file. Assume all directories in the path are small and can fit into a single disk block. For example, the **/users** directory can fit into a single data block.

/users	/users/diamond	/users/diamond/res	/users/dia/res/clou	file.
1 - inode	1 - inode	1 - inode	1 - inode	1 - inode
1 - dir	1 - dir	1 - dir	1 - dir	1 - data

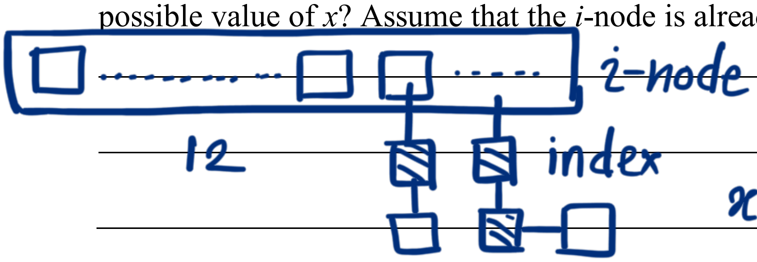
6. A file system using FAT table for file allocation is storing the following two files. FileXYZ in disk blocks 14, 36, 33, 97 and FileABC in disk blocks 55, 10, 70. What is a possible FAT table configuration? Note that there are many possible answers. You need to show just a possible one.

FileXYZ	10
FileABC	30

11	14
12	36
13	33
-1	97

31	55
32	10
-1	70

7. Consider a UNIX file system with 4096-byte blocks. The disk pointer needs 4 bytes. Suppose you are opening a file and seeking to location *x* from the beginning of the file and trying to access 2 bytes in the file. You end up reading 3 disk blocks to get the data bytes. What is a possible value of *x*? Assume that the *i*-node is already in memory.

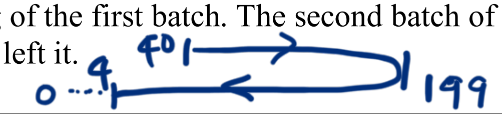


$y = 12 \times 4096 + \left(\frac{4096}{4}\right) \times 4096$

$y < x < y + \left(\frac{4096}{4}\right)^2 \times 4096$

$x = 4243456\ (+1) = 4243457$

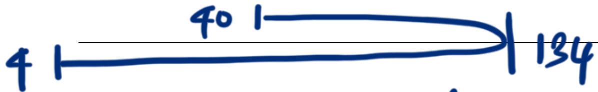
8. Consider the SCAN disk scheduling algorithm. Suppose the disk scheduler receives 15, 29, 38, 27, 4, 134, 81, 42 as data requests and the disk read/write head is at track 40 and moving upwards (towards higher numbered tracks) and the disk has tracks numbered 0 to 199. What is the total number tracks crossed by the SCAN algorithm? Suppose the requests arrived in two batches: {15, 29, 38, 27} and 4, 134, 81, 42}. What is the total number of tracks crossed by the SCAN algorithm? For the second part, assume that the read/write head is at 60 and moving upwards at the beginning of processing of the first batch. The second batch of requests find the head where the first batch has left it.



160 + 195 = 355 tracks

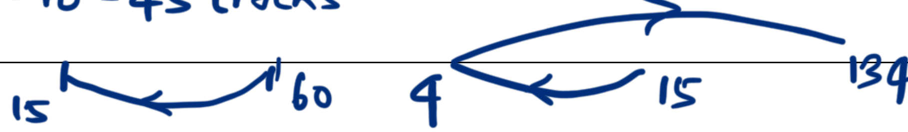
140 + 184 = 324 tracks   16 + 135 = 151 tracks

9. Repeat the above problem for the LOOK disk scheduling algorithm.



$94 + 130 = 224$ tracks

$61 - 16 = 45$ tracks



10. Repeat the above problem for the SSTF disk scheduling algorithm.

40 - 42 - 38 - 29 - 27 - 15 - 4 - 81 - 134 $\quad 36 + 130 = 166$ tracks

60 - 38 - 29 - 27 - 15 ~ 45 tracks $\quad$ 15 - 4 - 42 - 81 - 134

$11 + 130 = 141$ tracks

11. Consider a system with 4 processes: P_1, P_2, P_3, and P_4. There are 3 types of resources: X, Y, and Z in the system. The following are the matrices denoting maximum resource requirement and resources held by the different processes, respectively. Also, the currently available resources are given by the availability vector. Determine whether the system is safe or not. If it is safe, is it very safe?

$$Max = \begin{array}{c} P_1 \\ P_2 \\ P_3 \\ P_4 \\ \end{array}\begin{array}{c} \begin{bmatrix} 9 & 6 & 5 \\ 9 & 6 & 4 \\ 9 & 7 & 5 \\ 7 & 6 & 6 \end{bmatrix} \\ \begin{matrix} X & Y & Z \end{matrix} \end{array} \quad Hold = \begin{array}{c} P_1 \\ P_2 \\ P_3 \\ P_4 \\ \end{array}\begin{array}{c} \begin{bmatrix} 6 & 4 & 5 \\ 7 & 5 & 4 \\ 5 & 4 & 3 \\ 4 & 3 & 4 \end{bmatrix} \\ \begin{matrix} X & Y & Z \end{matrix} \end{array} \quad Avail = [\,2 \quad 2 \quad 2]$$

$$Need = \begin{bmatrix} 3 & 2 & 0 \\ 2 & 1 & 0 \\ 4 & 3 & 2 \\ 3 & 3 & 2 \end{bmatrix}\begin{array}{c} P1 \\ P2 \\ P3 \\ P4 \end{array}$$

It is safe, but not very safe.
P2 runs first.
Then any of rest may run.

12. If the above state is safe, what is the largest request that could be launched by P_1 that would be accepted by the Banker's algorithm? The size of the request is the total number of units allocated as part of the request. For example, a request [1, 2, 1] has a size of 1 + 2 + 1 = 4.

To preserve safe state P2 should still be able to run to completion upon accepting P1's request.

$P1_{MAX} = [AVAIL] - P2_{NEED} = [2\ 2\ 2] - [2\ 1\ 0] = [0\ 1\ 2]$

*But P1 already has MAX of resource Z required.

So biggest request = [0 1 0] = 1

Two ways: incoming job pre-empts any low priority execution right away OR let it run the current quantum.

13. Write the pseudo code for multi-level feedback scheduling algorithm we discussed in the class. Use C like notation. Keep your pseudo-code in about 10 lines, but capture all the core details of the algorithm.

Three events to deal with: Job-Arrival, Quantum-Expire, Job-sleep (Before quantum expiry)

```
Job-Arrival(j){
  enqueue(j);
  if (runJob >0) {
    preempt_save(runJob);
    runJob = j;
  }
}

Job-Sleep(j) {
  if (j.QLevel >0)
    push_up(j);
}

QuantumExpiry (j) {
  if(j.QLevel < max)
    push_down(j);
}
```

14. Consider the ***Shortest Remaining Time Next*** algorithm. Assume a single processor core. This algorithm is used to schedule the following tasks <10, 0>, <20, 5>, <5, 10>, <15, 10>, <10, 15>, <10, 20>. In the above list, <*x*, *y*> means *x* is the burst time and *y* is the arrival time in arbitrary time units. At what time would all of the above tasks complete their run? What is the average turnaround time?

Answer Provided in the last page.

15. If we use shortest job first with the above tasks, what would be the completion time of the tasks? What would be the average turnaround time?

16. Linux containers provide a virtual machine like functionality. Name two important feature in the Linux kernel that are key building blocks for containers? What is a key advantage of containers over virtual machines?

* Linux Feature: Namespaces & Cgroups.
* Containers are lightweight.

17. Suppose you have a program that crashes the whole operating system (Linux) when it is run for long periods of time. After some examination of the problem, you figure out that kernel memory exhaustion is the reason for the problem. The program still runs for long periods of time and you are interested in running it to obtain the results (for one your project reports), but are concerned about the system crash that ensues. Would you be able to run it in a container and isolate the host from crashing afterwards? If the container can help explain how it helps. If not, explain the reason as well.

$\boxed{\text{Yes}}$ The memory cgroup has the memory.kmem_limit settings which can be set such that the process is killed before causing kernel memory exhaustion.

18. Suppose you are given a program that is crashing the machine on which runs because the program is managing to trigger an obscure kernel bug by the sequence of system calls, signals, etc. You are still able to run it for long periods of time before the bug is triggered. You have two choices to run the program: a virtual machine or a container. Which one would you select? Briefly explain your answer.

We cannot do anything to isolate the host in this case when the container is used. So we need to use VM if host functioning is necessary.

19. Let w = 0,4,1,4,1,5,1,6,2,6,3,6,2,6,4,5 be a page reference string. Given a main memory that can hold up to 3 page frames and initially empty, how many page faults will the given reference string cause using the FIFO strategy?

$\boxed{\text{9 Faults}}$

Frame 0	0	0	0	0	0	5	5	5	5	5	3	3	3	3	3	3
Frame 1		4	4	4	4	4	4	6	6	6	6	6	6	6	4	4
Frame 2		.	1	1	1	1	1	1	2	2	2	2	2	2	2	5

20. Let w = 0,4,1,4,1,5,1,6,2,6,3,6,2,6,4,5 be a page reference string. Given a main memory that can hold up to 3 page frames and initially empty, how many page faults will the given reference string cause using the LRU strategy?

$\boxed{\text{9 Faults}}$

Frame 0	0	0	0	0	0	5	5	5	2	2	2	2	2	2	2	5
Frame 1		4	4	4	4	4	4	6	6	6	6	6	6	6	6	6
Frame 2			1	1	1	1	1	1	1	1	3	3	3	3	4	4

Part II – Long Answers. Your answers should **still** be up to the point. Provide the answers in the allotted space. **Answer all questions**. **Each question is worth 10 points**.

1. **[14 points]** Consider a virtual memory system with logical (virtual) addresses that are 48 bits long. The size of a page table entry is 8 bytes. If the page size is 2048 bytes, what is the number of levels in the page tables? Assume that the physical addresses have 32 bits. **Hint:** Not all information is necessary to solve the problem.

   Show the allocation of address bits for the different components of the paged logical (virtual) address for the maximum and minimum sized page configurations (if they exist).

   Sketch the organization of the page tables and show how the different portions of the logical (virtual) address are used to index the different tables and eventually find the physical address. Mark all important components including how the CPU points to the page tables.

2. **[13 points]** Develop a semaphore-based solution for the Dining Philosophers problem. Your solution should have no deadlock by allowing only up to two philosophers to eat concurrently. State all assumptions.

Q14

3. **[13 points]** Develop a monitor-based solution for the Dining Philosophers problem. Your solution should have no deadlock by allowing only up to two philosophers to eat concurrently. State all assumptions.

		Arrival		Start Time	End time	Turn Ar^nd Time	<x, y>
1	P_0	0		0	10	10	<10, 0>
6	P_1	5	[20]	50	70	65	<20, 5>
2	P_2	10	[5]	10	15	5	<5, 10>
5	P_3	10	[15]	35	50	40	<15, 10>
3	P_4	15	[10]	15	25	10	<10, 15>
4	P_5	20	[10]	25	35	15	<10, 20>

Completion time = 70

$$\text{Average Turnaround time} = \frac{10+65+5+40+10+15}{6}$$

$$= 145/6 \sim \boxed{24.16}$$

Q15

* The above scheduling does not pre-empt any process. So the answer for Q15 is also the same as above.