CSC 139 Operating System Principles
Homework 2

Fall 2021

Posted on Nov. 23, due on Dec. 5 (11:59 pm). Write your own answers. Late submission will be penalized (turn in whatever you have).

Exercise 1. (OSC 6.22) (15%) Consider the code example for allocating and releasing processes shown below:

```
#define MAX_PROCESSES 255

int number_of_processes = 0;

/* the implementation of fork() calls this function */

int allocate_process() {

    int new_pid;

    if (number_of_processes == MAX_PROCESSES)
    return -1;
    else {
        /* allocate necessary process resources */
        ++number_of_processes;

        return new_pid;
    }
}

/* the implementation of exit() calls this function */ void
release_process() {
    /* release process resources */
    --number_of_processes;
}
```

---

1. Identify the race condition(s).

- Race condition happen when they increment and decrement the shared counter. As there is a chance that when you increment the counter you will never hit the == and it being true.

2. Assume you have a mutex lock named mutex with the operations acquire() and release(). Indicate where the locking needs to be placed to prevent the race condition(s).

- A lock put before and after the increments and decrements would prevent this race condition.

Exercise 2. (OSC 8.28) (20%) Consider the following snapshot of a system:
Answer the following questions using the banker's algorithm:

	Allocation				**Max**			
	A	B	C	D	A	B	C	D
T_0	3	1	4	1	6	4	7	3
T_1	2	1	0	2	4	2	3	2
T_2	2	4	1	3	2	5	3	3
T_3	4	1	1	0	6	3	3	2
T_4	2	2	2	1	5	6	7	5

Available

A	B	C	D
2	2	2	4

Need	A	B	C	D
T0	3	3	3	2
T1	2	1	3	0
T2	0	1	2	0
T3	2	2	2	2
T4	3	4	5	4

1. Illustrate that the system is in a safe state by demonstrating an order in which the threads may complete.

- T3,T0,T1,T2,T4

2. If a request from thread T_4 arrives for (2,2,2,4), can the request be granted immediately?

    - No it can't as it exactly fits the amount of available resources plus there is no safe sequence with this request.

3. If a request from thread T_2 arrives for (0,1,1,0), can the request be granted immediately?

    - Yes it can as it exactly fits the amount of available resources plus it is less than its need. T1 ,T3,T4,T0,T1 is the safe sequence.

4. If a request from thread T_3 arrives for (2,2,1,2), can the request be granted immediately?

    - No, as there is no safe sequence once the request is done.

Exercise 3. (OSC 8.22) (5%) Consider a system consisting of four resources of the same type that are shared by three processes, each of which needs at most two resources. Is this system deadlock-free? Why or why not?

- It is as there is a sequence in which all three processes will at one point be able to have both resources if need be. As long as they setup a lock to block multiple processes acquiring resources at the same time without waiting for the other processes to let go of it.

Exercise 4. (5%) Can a system be in a state that is neither deadlocked nor safe? If yes, give an example system.

- Yes it can. As in an unsafe state as long as they do not try to allocate their maximum amount of resources without it being available, causing deadlock, then they can continue running although not safe. Which due to the fact that they are never waiting in deadlock since they do have the proper amount albeit under their max they will be in a state that is unsafe but won't cause deadlock.

Exercise 5. (5%) Two processes, *A* and *B*, each need three records, 1, 2, and 3, in a database. If *A* asks for them in the order 1, 2, 3, and *B* asks for them in the same order, deadlock is not possible. However, if *B* asks for them in the order 3, 2, 1, then deadlock is possible. With three resources, there are 3! or six possible combinations in which each process can request them. What fraction of all the combinations is guaranteed to be deadlock-free?

All combinations are with A = 1,2,3

1,2,3	No deadlock
1,3,2	No deadlock
2,1,3	Potential deadlock (A could be waiting on 2 as B waits on 1)
2,3,1	Potential deadlock (A could be waiting on 2 while B waits on 1)
3,1,2	Potential deadlock (A could be waiting on 3 while B waits on 1)
3,2,1	Potential deadlock(A could be waiting on 3 while B waits on 1)

- So 2/6 are deadlock free or 1/3 are deadlock free

Exercise 6. (5%) A machine has 48-bit virtual addresses, 32-bit physical addresses, and the page size is 8 KB. How many entries are needed for one process's page table?

- Offset: 8*1024 = 8196 = 2^13
- Number of pages: 2^48/2^13 = 2^35

Exercise 7. (5%) A computer with a 32-bit address space uses a two-level page table. Virtual addresses are split into a 9-bit top-level page table field (the directory), an 11-bit second-level page table field, and an offset. How large are the pages and how many are there in the address space?

- Offset: 32-9-11 = 12
- Number of Pages: 2^32/2^12 = 2^20

Exercise 8. (OSC 10.1) (10%) Under what circumstances do page faults occur? Describe the actions taken by the operating system when a page fault occurs.

- When the system trys to access a page not currently in memory.
- Operating system looks at another table to decide if it is a valid reference or not. Then looks for a free frame. Swaps the page into frame via scheduled disk operation. Resets table to indicate to indicate page is now in memory. Restarts instruction that caused page fault.

Exercise 9. (OSC 9.6) (15%) Given six memory partitions of 300 KB, 600 KB, 350 KB, 200 KB, 750 KB, and 125 KB (in order), how would the first-fit, best-fit, and worst-fit algorithms place processes of size 115 KB, 500 KB, 358 KB, 200 KB, and 375 KB (in order)? Rank the algorithms in terms of how efficiently they use memory.

First Fit:

300KB	600KB	350KB	200KB	750KB	125KB
-115 (185 remain)	-500 (100 remain)	-200 (150 remain)		-358 (392 remain)	
				-375 (17 remain)	

Best Fit:

300KB	600KB	350KB	200KB	750KB	125KB
	-500 (100 remain)		-200 (0 remain)	-358 (392 remain)	-115 (10 remain)
				-375 (17 remain)	

Worst Fit:

300KB	600KB	350KB	200KB	750KB	125KB
	-358 (242 remain)	-200 (150 remain)		-115 (635 remain)	
				-500 (135 remain)	

- 375 can not fit into this system

Ranking:

- In terms of ranking it would go Best Fit, First fit, and Worst fit in decreasing order.
    - Worst fit is last due to it not even being to fit all of the processes in memory
    - First fit is second as even though it is able to fit in all processes like Best fit it does so with smaller partitions
    - Best fit is the best as it fits all and has the biggest partitions remaining

Exercise 10. (OSC 10.9) (15%) Consider the following page reference string: 7, 2, 3, 1, 2, 5, 3, 4, 6, 7, 7, 1, 0, 5, 4, 6, 2, 3, 0, 1. Assuming demand paging with three frames, how many page faults would occur for the following replacement algorithms? (Show the steps and highlight the page faults.)

- LRU replacement

	7	2	3	1	2	5	3	4	6	7
Index 0	7	7	7	1	1	1	3	3	3	7
Index 1		2	2	2	2	2	2	4	4	4
Index 2			3	3	3	5	5	5	6	6

	7	1	0	5	4	6	2	3	0	1
Index 0	7	7	7	5	5	5	2	2	2	1
Index 1	4	1	1	1	4	4	4	3	3	3
Index 2	6	6	0	0	0	6	6	6	0	0

- 18 page faults

- FIFO replacement

	7	2	3	1	2	5	3	4	6	7
Index 0	7	7	7	1	1	1	1	1	6	6
Index 1		2	2	2	2	5	5	5	5	7

Index 2			3	3	3	3	3	4	4	4

	7	1	0	5	4	6	2	3	0	1
Index 0	6	6	0	0	0	6	6	6	0	0
Index 1	7	7	7	5	5	5	2	2	2	2
Index 2	4	1	1	1	4	4	4	3	3	1

- 17 page faults

- Optimal replacement

	7	2	3	1	2	5	3	4	6	7
Index 0	7	7	7	1	1	1	1	1	1	1
Index 1		2	2	2	2	5	5	5	5	5
Index 2			3	3	3	3	3	4	6	7

	7	1	0	5	4	6	2	3	0	1
Index 0	1	1	1	1	1	1	1	1	1	1
Index 1	5	5	5	5	4	6	2	3	3	3
Index 2	7	7	0	0	0	0	0	0	0	0

- 13 page faults

Please complete the following survey questions:

1. How much time did you spend on this homework?

   - 4 hours

2. Rate the overall difficulty of this homework on a scale of 1 to 5 with 5 being the most difficult.

   - 3

3. Provide your comments on this homework (e.g., amount of work, difficulty, relevance to the lectures, form of questions, etc.)