CS342 Operating Systems - Spring 2022
Homework #3

Assigned: May 5, 2022.
Due date: May 15, 2022, 23:59. Document version: 1.1

1. Assume we have 2 processes P1 and P2 and one resource type R. There are 15 instances of that resource type. The maximum demand for process P1 is 10, for process P2 is 12. How many allocations are safe state (in the context of deadlock avoidance)? For example, (0,0) is a safe state (number of instances allocated to P1 is 0, and to P2 is 0).

2. Consider the following situation in a system where there are 5 processes and 3 resource types: A, B, C. No deadlock avoidance or prevention is applied. Initially, there exist 10 A, 6 B, and 4 C in the system (without allocations yet). Is there a deadlock at the moment? Prove your answer.

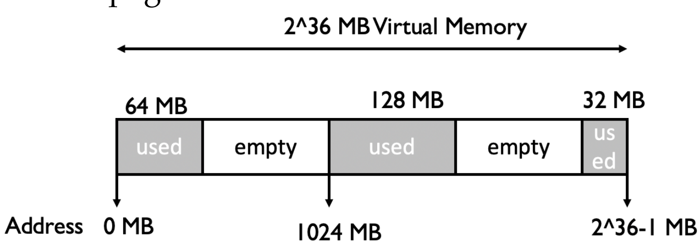
	Alloc	Request
P1	1 0 0	0 0 1
P2	3 0 1	0 4 0
P3	0 2 2	3 1 2
P4	1 0 1	2 3 0
P5	2 1 0	3 3 3

3. Consider the following page reference string for a process. There are 4 frames allocated to the process.

2 5 3 5 6 3 8 5 4 1 2 4 8 5 3 8 4 3 1 5

Show the memory state (which pages are in memory and their reference bits, if applicable for the page replacement algorithm) *after* each page reference, for the following page replacement algorithms: a) Optimal, b) LRU, c) Clock algorithm (assume reference bits are cleared after every 5th reference). *For example, for the clock algorithm, the memory state after the first 5 references will be: *2(1), 5(1), 3(1), 6(1). That means the victim pointer is pointing to page 2. Then all reference bits will be cleared and the memory state will be *2(0), 5(0), 3(0), 6(0), after clearance. After the reference to page 3 (the 6th reference), the memory state will be: *2(0), 5(0), 3(1), 6(0).*

4. Consider a computer system that has 4 GB RAM. Assume the virtual addresses are 36 bits long and page size is 16 KB. Consider a process P that has the following virtual memory layout. Assume a page table entry is 8 bytes long, no matter which page table structure is used.



a) What is the size of the page table of P, if single-level page table scheme is used (i.e., the amount of RAM required to hold the page table information of process P)?

b) What is the amount of RAM (in KB) required to hold the page table information for process P, if the system is using two-level paging with address division [11, 11, 14]?

c) What would the the size of an *inverted* page table in this machine, assuming a page table entry is 8 bytes long and *ignoring* chaining effects?

5. Assume a file system is using *combined indexing* scheme to keep track of the blocks allocated to a file, as in Linux/Unix file system. In the inode of a file, there are 12 direct pointers, one single-indirect pointer, one double-indirect pointer, and one triple-indirect pointer. Assume the size of a disk block address (i.e., disk pointer size) is 8 bytes. Assume block size is 4 KB.

a) What is the maximum file size?

b) How many index blocks are required for files of size 30 KB, 256KB, 15 MB, 512 MB, 32 GB?

c) Assume nothing is cached (no disk caching applied in the system) and we know the disk location of the inode of a file of size 1 GB. How many disk accesses are required to access a byte at offset 2^14 (16K) in the file? How many disk accesses are required to access a byte at offset 2^26 in the file?

6. Consider a hard disk with the following parameters.

RPM: 7200
Average seek time: 5 ms.
Max transfer rate: 80 MB/s.

a) Consider random I/O, where each request if for 4 KB of data (1 block). How many random I/O operations per second are possible? What is the random I/O throughput (random I/O data rate) we can achieve?

b) Consider sequential I/O, where each request is for 10 MB of contiguous data on the disk. What is the sequential I/O throughput (sequential I/O data rate) we can achieve?

7. Consider RAID level 5 system. *Striping unit* size is 4 KB (1 block). Assume we have 9 disks. Hence, in one *stripe*, we have 8 data blocks (each on a different disk) and 1 parity block (again on a different disk). Each disk has identical parameters, where for a disk:

RPM is 15000,
Max transfer rate is 100 MB/s, and
Average seek time is 4 ms.

a) What is the steady-state random read throughput of the RAID5 system (each random read request is small, i.e., for 1 block of data, which is 4 KB)?

b) What is the steady-state sequential read throughput of the RAID5 system (assume a sequential read is reading 100 MB contiguous disk data)?

8. Consider a RAID4 system with 9 disks. Disk are identical and the MTTF of a disk is 50000 hours. Assume MTTR (mean time to recovery) for this RAID4 system is 48 hours. What is the mean time to data loss (MTTDL) for this RAID4 system?

9. Consider an SSD disk where (physical) block size is 256 KB and (physical) page size is 4 KB. Assume the file system installed on this disk is also using blocks (logical blocks) of size 4 KB. Assume the file system would like to write 10 blocks to the SSD and issued the related requests to the SSD (10 write requests issued). But the disk is full: there is no physical block that has writable empty pages. Hence SSD will erase a block and will write the logical blocks into that physical block. The selected physical block has 40 pages as invalid (containing dead blocks). The remaining pages are in active use (containing live blocks). What is the total time needed to write the 10 logical blocks into the selected physical SSD block? Note that before erasing the selected block, the live pages on the block must be read into volatile memory (SRAM) of the SSD. Assume reading a page takes 40 microseconds, writing a page takes 200 microseconds, and erasing a block takes 4 ms.

10. Consider a file F to be shared by N processes. Each process i has ID i (1 <= i <= N). The file can be accessed concurrently by multiple processes, if the sum of the IDs of these processes is less than or equal to M.

a) Write a monitor that will control access to the file. That means the monitor will implement two functions, request() and release(), that will be called by a process that would like to access the file. You also need to define the monitor variables required for the solution. A process will call the request() function before accessing the file and release() function when it has finished accessing the file.

b) This time implement the request() and release() functions using mutex and condition variables (like POSIX Pthreads mutex and condition variables). You need to define some global variables as well to implement these functions.

11. How many different integer sequences (order is important) may be printed out by the following pseudo-code? Write down all of them.

```
Semaphore X = 0; //shared by all processes
Semaphore Y = 0; //shared by all processes
Semaphore Z = 0; //shared by all processes
int n;

main() {
        n = fork();
        if (n==0) {
```

```
        signal(X)
        print (40);
        wait (Y);
        print (30);
        wait(Z);
        print(45);
        exit (0);
    }
    signal (Y);
    print (60);
    wait (X);
    print (10);
    print(80);
    signal(Z);
}
```