COM S 352: Introduction to Operating Systems
Midterm Practice Exam
Spring 2023

Cover Sheet

Student Name: ____

Format:
- Time: 50 mins
- Points: 100
- Question Types: matching, true/false and short answer

Instructions:
- You may use 1 (one) letter sized sheet of paper (front and back) or 2 one-side sheets, that you have prepared yourself with notes before the exam, as "cheat sheet" during the exam.
- You may not consult classmates, electronic devices or resources other than the cheat sheet during the exam.
- Questions of clarification should be asked directly to an instructor or TA.

Question	**Points**
1	/24
2	/30
3	/10
4	/10
5	/8
6	/8
7	/10
Total	/100

1. (24 pts, 3 pts each) For each description on the left, select the best matching term on the right, each term is used only once but some will not be used.

fork makes a copy of the current process

CPU bound a process that frequently exceeds its time slice

semaphore mechanism that can provide both locking and signaling

TCB created as a result of calling pthread_create()

Kernel mode system calls are implemented using traps because they require this

pipe a queue for inter-process communication

locality reason caching can improve performance

TestAndSet machine instruction that can be used to implement mutex locks

a. CPU bound
b. TestAndSet
c. Heap
d. TCB
e. kernel mode
f. page out
g. pipe
h. semaphore
i. fork()
j. swap
k. MMU
l. locality

2. (30 pts, 2 pts each) Which of the following statements are true? Write T or F for true or false.

F____ In RR scheduling, best performance is when the time quantum is small with respect to context-switch time.

F____ In the Linux Completely Fair Scheduler processes with low nice values always run before those with high nice values.

T____ LRU never experiences Belady's Anomaly.

T____ The purpose of priority boost is to prevent starvation.

F____ A context switch is preformed when changing a process from the blocked state to ready state.

F____ Thrashing is a result of insufficient multiprogramming.

T____ A segment can be shared by multiple processes.

F____ If a page's contents in memory are different than they are on disk, the valid bit must be set to invalid.

F____ An illegal memory access results in a page fault.

F____ A translation lookaside buffer (TLB) is used to search for free space in physical memory.

T____ Paging has the problem of internal fragmentation.

T____ Tick based Operating Systems depend on a hardware interrupt to keep track of time.

T____ The bounded-buffer problem can be solved using only semaphores to control concurrency.

T____ A binary semaphore is equivalent to a mutex lock.

T____ A thread requesting a resource never causes a deadlock as long as that thread does not currently have any other resources assigned to it.

3. (10 pts) Consider the following set of jobs, with arrival times and the length of CPU bursts given in milliseconds.

	Arrival Time	CPU Burst
A	0	10
B	1	12
C	4	10
D	5	2

Create Gantt charts and compute the average response time for each of the following scheduling algorithms. Show your calculations.

a) SJF (without preemption)

Gantt chart:

| A | D | C | B |

0 10 12 22 34

Average response time is _8.5___ (show calculation below)

(0 + (10 – 5) + (12 – 4) + (22 – 1))/4 = 8.5

b) STCF

Gantt chart:

| A | D | A | C | B |
0 5 7 12 22 34

Average response time is __7.25____ (show calculation below)

(0 + (22-1) + (12-4) + (5-5))/4 = 7.25

4. (10 pts) Given the reference string of page accesses: 4 3 2 1 4 3 5 1 and a system with 3 page frames, how many page faults result when using the following page replacement policies? Show your work.

a) FIFO

Number of page faults is __8___ (show work below)

Page #	4	3	2	1	4	3	5	1
H/M?	M	M	M	M	M	M	M	M
Frames		4	4	4	3	2	1	5
			3	3	2	1	4	4
				2	1	4	3	3

b) OPT

Number of page faults is __5___ (show work below)

Page #	4	3	2	1	4	3	5	1
H/M?	M	M	M	M	H	H	M	H
Frames		4	4	4	4	4	4	5
			3	3	3	3	3	3
				2	1	1	1	1

5. (8 pts) Below is an excerpt from xv6 scheduler code encountered in Project 1.

```
void
sched(void)
{
  // ...
  struct proc *p = myproc();
  // ...
  swtch(&p->context, &mycpu()->context);
  // ...
}
```

Why does this function call `swtch()`? What does `swtch()` do? Answer in 2 to 3 sentences.

sched() calls swtch() to perform a context switch from the context of the kernel code that interrupted the currently running user process (I.e., the process returned by myproc()) into the context of the scheduler. The purpose is for the scheduler to possibly context switch to a different process. swtch() performs the
actions of a context switch: saving the current CPU registers to the current user process' context and
loading CPU registers from the saved context of the next user process.

6. (8 pts) Given a virtual address space of 1,024 pages and a 4,096 byte page size, how many bits are required to store a virtual address. Show calculations.

1024 pages => page number takes 10 bits
Page size = 4096 bytes => page offset takes 12 bits
Hence, 10+12=22 bits are required to store a virtual address.

Or, Log (1024 * 4096) = Log(2^{10} * 2^{12}) = 22.

7. (10 pts) Consider two threads, a ping_thread and pong_thread, that output the words "ping" and "pong" respectively. The required output of the program is as follows (it is required that ping always goes first):

ping
pong
ping
pong
…

Provide a solution for ping_thread and pong_thread **using only semaphores** for concurrency control. Declare and initialize all variables. Exact pthread syntax is not required.

```
#include <pthread.h>

      // declare any required variables here
sem_t ping_sem, pong_sem;


void init() { // perform initialization here

sem_init(&ping_sem, 1);
sem_init(&pong_sem, 0);

}


void* ping_thread(void *arg) {
    while (1) { // complete the ping thread
           sem_wait(&ping_sem);
printf("ping\n");
sem_post(&pong_sem);
    }
}

void* pong_thread(void *arg) {
    while (1) { // complete the pong thread
sem_wait(&pong_sem);
printf("pong\n");
sem_post(&ping_sem);
    }
}
```