EC440, Fall 2025

Homework #2

Yigong Hu, yigongh@bu.edu

Problem 1: [Ryan]

Explain the tradeoffs between using multiple processes and using multiple threads

Problem 2: [Ryan]

Does a multi-threading solution always improve performance? Please ex- plain your answer and give reasons.

Problem 3: [Ryan]

Explain the tradeoffs between preemptive scheduling and nonpreemptive scheduling

Problem 4: [Silberschatz]

Which of the following components of program state are shared across threads in a multi-threaded process?

1. Register values
2. Heap memory
3. Global variables
4. Stack memory

Problem 5: [Ryan]

What are two differences between user-level threads and kernel-level threads? Under what circumstances is one type better than the other?

Problem 6: [Ryan]

What would be a possible problem if you executed the following program (and intend for it to run forever)? How can you solve it? .

```
#include <signal.h>
#include <sys/wait.h>
int main()
{
    for (;;) {
        if (! fork()) { exit(0);}
        sleep(1);
    }
    return 0;
}
```

Problem 7: [Ryan]

Consider the following program: .

```
#include <stdlib.h>
int main()
{
    fork();
    if (fork()) {
        fork();
    }
    fork();
    return 0;
}
```

Draw a tree diagram showing the hierarchy of processes created when the program executes. How many total processes are created (including the first process running the program)? **Hint:** You can always add debugging code, compile it, and run the program to experiment with what happens

Problem 8: [Ryan]

Show how a lock, and acquire() and release() functions can be implemented using atomic SWAP instruction. The following is the definition of swap instruction: .

```
void swap (char* x, char * y) // All done atomically
{
    char temp = *x;
    *x = *y;
    *y = temp
}

struct lock {
}

void acquire (struct lock *) {
}

void release (struct lock *) {
}
```

Problem 9: [Ryan]

Suppose you have an operating system that has only binary semaphores. You wish to use counting semaphores. Show how you can implement counting semaphores using binary semaphores.
Hints: You will need two binary semaphores to implement one counting semaphore. There is no need to use a queue – the queuing on the binary semaphores is all you'll need. You should not use busy waiting. The wait() operation for the counting semaphore will first wait on one of the two binary semaphores, and then on the other. The wait on the first semaphore imple- ments the queueing on the counting semaphore and the wait on the second semaphore is for mutual exclusion.

Problem 10: [Anderson]

Given the following mix of tasks, task lengths, and arrival times, compute the completion and response time for each task, along with the average response time for the FIFO, RR, and SJF algorithms. Assume a time slice of 10 milliseconds and that all times are in milliseconds.

Task	Length	Arrival Time	Completion Time	Response Time
0	85	0		
1	30	10		
2	35	15		
3	20	80		
4	50	85		
			Average:	

Problem 11: [Anderson]

Are there non-trivial workloads for which Multi-level Feedback Queue is an optimal policy? Why or why not? (A trivial workload is one with only one or a few tasks or tasks that last a single instruction.)