

Last Name: _____ First Name: _____

Instructions:

- This assignment is an individual assignment. You must work on this homework independently.
- There is no online submission for this homework. Homework **MUST** be submitted in class at the beginning of class on **Monday, March 9th**. Absolutely no late submissions will be accepted.
- This four-page homework **MUST** be printed, single-sided, and stapled in the top-left corner. We are not responsible for unstapled solutions and will not grade solution-only submissions (we require the solutions to be written or typed on the on the printed out homework).
- All answers must fit in the spaces provided on this hardcopy. No addendums or additional sheets please. Answers that are not clearly legible will be marked as incorrect.
- You must read and sign the statement at the end of the homework. Thank you.

Grader Use Only:

Question	Q1 (5)	Q2 (5)	Q3(2)	Q4 (3)	Q5 (10)	Q6 (5)	Q7 (5)	Q8 (5)	Total/40
Score:									
Grader:									

Question 1 (5 x 1point): Which of the following is correct use of pointers? (Mark correct or incorrect in the space provided)

- a) `int *x, q=0;`
`*x = q;` (a): _____
- b) `int b[2];`
`*b = 20;` (b): _____
- c) `char greet[80];`
`strcpy ("Hello", greet);` (c): _____
- d) `int *p, q[2];`
`p = malloc (sizeof (int));`
`*p = 3;`
`q[2]=*p;` (d): _____
- e) `int *x, y;`
`x = &y;`
`*x = 10;` (e): _____

Question 2 (5 x 1point): In (a) through (e) below, assume that all tasks **arrive together at time 0**. Consider the scheduling policies: (i) SJF, (ii) FIFO, (iii) Round-Robin with quantum set equal to 1/10 of the computation time of the shortest task.

- a) Which scheduling policy minimizes average task waiting time? (a): _____
- b) Which scheduling policy minimizes average task response time? (b): _____
- c) Which scheduling policy suffers the convoy effect? (c): _____
- d) Which scheduling policy has the largest context switch overhead? (d): _____
- e) Which scheduling policy may suggest a starvation effect? (e): _____

Question 3 (2 x 1point): Briefly, in TWO sentences or less, what is the difference between the following two cases?

a) Function g() calls function f().

b) Function g() creates a POSIX thread that executes function f().

Question 4 (3 x 1point): Connect each of three command names or calls on the left with the most appropriate definition on the right. If more than one definition applies, choose the best and most accurate one.

- | | |
|------|---|
| | Runs a new program in the current address space |
| | Returns from a function call |
| kill | Delivers a SIGTERM to a process |
| exec | Creates a child process |
| exit | Sends a signal to a process |
| | Terminates a UNIX process |
| | Creates a new POSIX thread |

Question 5 (10 x 1point): The following cases show pseudo-code executed by two different threads, thread 1 and thread 2 around a critical section. (If code of thread 2 is not mentioned, it is the same as thread 1.) Fill-in the table below specifying, for each case, (i) whether mutual exclusion is guaranteed and (ii) whether progress is guaranteed. In all cases, assume that all variables are initialized to zero in the beginning. In answering the question you should consider all possible scenarios including scenarios where a thread executes more than once (sequentially) and scenarios where only one of the two threads remains, while the other has terminated. However, you should assume that once a thread has entered into the code shown below, it will finish the block of code given if possible (eg: it won't exit/abort in the middle of our given code).

	Case (a)	Case (b)	Case (c)	Case (d)	Case (e)
Mutual exclusion (Yes/No)					
Progress (Yes/No)					

Thread 1

Thread 2

- a)

<pre>while (x > 0) { }; x ++; /* You may assume this line is atomic */ execute critical section; x --; /* You may assume this line is atomic */</pre>	
--	--

- b)

<pre>x2 = 1; while (x1 != 0) { }; execute critical section; x2 = 0;</pre>	<pre>x1 = 1; while (x2 != 0) { }; execute critical section; x1 = 0;</pre>
---	---

- c)

<pre>while (testandset (x)) { }; /* testandset returns current value then sets it to 1 */ execute critical section; x = 0;</pre>	
--	--

- d)

<pre>if (x is odd) { execute critical section; x=2; }</pre>	<pre>if (x is even) { execute critical section; x=1; }</pre>
---	--

- e)

<pre>while (y == 1) { }; y = 1; execute critical section;</pre>	<pre>while (y == 0) { }; y = 0; execute critical section;</pre>
---	---

Question 6 (5 x 1point): Mark True or False.

- a) You cannot join a detached thread (a): _____
- b) In non-preemptive scheduling a process cannot go from ready to blocked state (b): _____
- c) Unlocking a mutex can cause some process to go from blocked to ready state (c): _____
- d) When a thread calls **pthread_exit**, the entire process terminates (d): _____
- f) alarm(X) always terminates the process in X seconds (e): _____

Question 7 (5 points): You want to change the default behavior of **SIGUSR1** to print a message on the screen. You also want to block **SIGINT** while SIGUSR1 is handled. Your process should continue to execute in the absence of signals. Which of the following calls you are likely to use? (Circle all that apply.) Each wrong choice knocks off one point.

- sleep alarm sigwait
- sigprocmask sigaction sigsuspend

Question 8 (5 x 1point): Please complete the definition of each of the performance criteria below by filling-in the spaces. Only ONE word per space, please.

- a) The turnaround time minus the _____ time is equal to the waiting time.
- b) The turnaround time is the time from the process creation time to its _____ time.
- c) The total time a process spends in queues is called _____ time.
- d,e) Response time is the interval from _____ time to _____ time.

I have neither received nor given help on this assignment.

Your Signature: _____