CompSci 3SH3 - Sample Exam

Multiple Choice Questions

1. What one of the following is not true?

a) Kernel is the program that constitutes the central core of the operating system
b) Kernel is the first part of operating system to load into memory during booting
c) Kernel is made of various modules which cannot be loaded in running operating system
d) Kernel remains in the memory during the entire computer session

2. Which of the following is not a system call?

a) fork()
b) wait()
c) pthread_create()
d) pthread_exit()
e) Two of the above
f) None of the above
g) All of the above

3. The OS structure used by the original UNIX is monolithic. The reason this architecture is used is:

a) Easy to implement and extend
b) Speed and efficiency
c) It is more flexible
d) It consumes less power to execute an instruction

4. Which storage is the fastest?

a) Solid state drive
b) Hard drive
c) L1 cache
d) Registers
e) RAM
f) ROM
g) None of the above

5. Which of the following cannot cause deadlock?

a) Not releasing the mutex lock, which causes the thread to have exclusive access, there by preventing a deadlock
b) If two or more mutex locks are used, the order in which the locks are acquired are different for each thread, which prevents deadlock because each thread can acquire a different lock
c) If there is a circular wait where P1 is waiting on P2, and P2 is waiting on P3, and P3 is waiting on P1.
d) All of the above
e) None of the above

6. Which one of the following is a volatile storage?

a) Hard drive
b) Floppy disk
c) CDs/DVDs
d) Random Access Memory
e) Cache
f) Registers
g) All of the above
h) Only (f) and (e)
i) Only (d) and (e) and (f)
j) None of the above

7. Which one of the caches are managed by the operating system (OS)?

a) L1 Cache
b) L2 Cache
c) L3 Cache
d) All of the above
e) None of the above
f) Only L3 cache is managed by the OS, because it is furthest from the CPU

8. Which of the following components of are shared across threads in a multithreaded process?

a) Register values
b) Heap memory
c) Global variables
d) Stack memory
e) All of the above
f) Only (a) and (c)
g) Only (b) and (c)
h) Only (a) and (b) and (c)
i) None of the above

True & False Questions

9. Interrupt response time is usually shorter in polling system than in vectored interrupt system. (T / F)

10. Primary storage is a volatile type of memory. (T / F)

11. SSD is primary storage. (T / F)

12. In this course (3SH3), a binary semaphore is is semantically equivalent to a lock. (T / F)

13. Transposing a matrix in parallel can be solved by data parallelism. (T / F)

14. It is possible to have a deadlock involving only one single-threaded process. (T / F)

15. It is possible to have a deadlock with a single mutex lock. (T / F)

16. A system of four resources of the same type that are shared by three threads, each of which needs at most two resources, is in a deadlock state. (T / F)

Short Answer Questions

17. We know that the order in which mutex locks are acquired is important to prevent deadlock. For example, if the locks are acquired in a different order across multiple threads, then deadlock can occur. However, does it matter in which order the locks are released? Why or why not? Show your work.

18. Is it possible to have concurrency but not parallelism? Explain

19. Should terminating a process also terminate all of its child processes? Give an example where this is a good idea and another example where it is not a good idea.

20. For the following code, draw the corresponding process tree diagram:

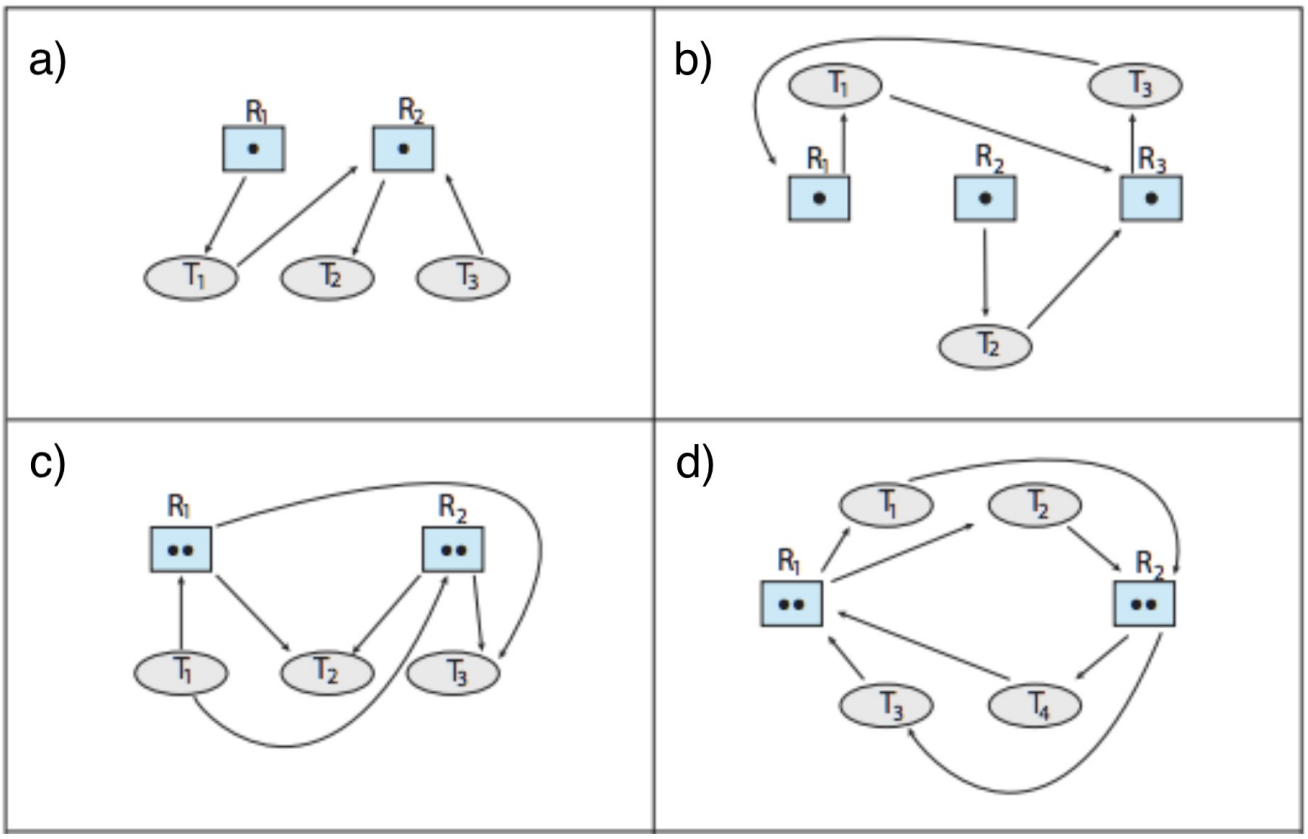
```
main() {

    pid_t pid;
    pid = fork();

    if (pid == 0)
        fork();

    fork();
}
```

21. For the following code, what value is printed at lines "A", "B", "C", and "D". Assume that the PID of the parent process is 500, and the PID of the child process is 501.

```
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>

main() {

    int pid_t pid, pid1;

    pid = fork();

    if (pid < 0) {
        fprintf(stderr, "Fork Failed");
        return 1;
    } else if (pid == 0) {
        pid1 = getpid();
        printf("child: pid = %d",pid);      // A //
        printf("child: pid1 = %d",pid1);    // B //
    } else {
        pid1 = getpid();
        printf("parent: pid = %d",pid);     // C //
        printf("parent: pid1 = %d",pid1);   // D //
        wait(NULL);
    }

    return 0;
}
```

22. Refer to the resource allocation graph (RAG), below, and answer the following questions:

i) For each RAG, state whether the system is in deadlock or not.
ii) If the system is in deadlock, provide the cycle of threads/resources that causes the deadlock
iii) If the system is not in deadlock, provide the order of execution that will allow the system to be in a safe state.



	ALLOCATION	**MAX**
	A B C D	A B C D
T0	3 0 1 4	5 1 1 7
T1	2 2 1 0	3 2 1 1
T2	3 1 2 1	3 3 2 1
T3	0 5 1 0	4 6 1 2
T4	4 2 1 2	6 3 2 5

23. Above is a snapshot of a system. Using Banker's algorithm, determine whether or not each of the following states is unsafe. If the state is unsafe, illustrate the order in which the threads may complete. Otherwise, explain why the system is in an unsafe state.

a) Available = (1 0 0 2)

b) Available = (0 3 0 1)

24.

```
#include <stdio.h>
#include <unistd.h>

int main(int argc, char * argv[]) {
    fork();
    fork();
    printf("hello\n");
    fork();
    printf("hello\n");
    printf("hello again\n");
    return 0;
}
```

For the code above, answer the following questions:

a) How many times is "hello" printed?
b) How many times is "hello again" printed?
c) How many processes are created in total?