CS 460 Final Exam Review Questions

♦ Review your labs, quizzes, previous exams, previous review sheets, and sacrifice a goat diagram.
READ THE BOOK

0. Vocabulary!

1. Why would you put /tmp on its own partition?

2. To turn off Address Space Layout Randomization, we ran the command:

```
echo 0 | sudo tee /proc/sys/kernel/randomize_va_space
```

How does writing data to this file change a kernel setting?
The file /proc/sys/kernel/randomize_va_space is a virtual file. What is a virtual file?
What benefits are there to using virtual files to give access to this type of functionality?

3. We discussed the fact that printf() may buffer output and not print data to the screen immediately. What benefits does this provide to a program that uses printf()?

4. How does your CS460_Shell work?
    1. Specifically, why is dup2() called? Is dup2() called before or after fork()? Why?

    2. When is pipe() called, before or after fork()? Why?

5. How much do you appreciate strtok_r() now?

6. Sketch the ELF file for CS460_Shell. Draw the process memory layout for CS460_Shell and refer to your ELF drawing to explain exactly where each piece of process memory originates.

7. Does a mutex protect code or data? Justify your answer.

8. What (bad things) would happen if a shared library used compile time address binding?

9. What does it mean to say a function is *thread-safe*? Explain what may cause a function to not be thread-safe.

10. What parts of Arch are Linux and what parts are GNU?

11. Looking at figure 6.15 in your book (page 244), explain precisely how this solution could deadlock.

12. page 236 and 237 show a possible implementation for wait() and signal(). Why do these implementations require a list? What property of synchronization could be violated without this list?

13. What does it mean when we say "POSIX is a specification, not an implementation?" Describe why it is useful to proscribe an interface for pthreads but not a specific implementation.

14. When we say the first Linux scheduler was O(n), what operation exactly is O(n)? What is n?

15. When we say the previous Linux scheduler was O(1), what operation exactly is O(1)?

16. In a priority based scheduler, why might you need to boost the priority of processes as they age?

17. What portions of the Linux kernel are likely to be written in assembly language?

18. With respect to the L1 cache, why might accessing an array in linear fashion (0 to n in steps of 1) be faster than accessing the same array in a random fashion?

19. Describe what code in Game of Life might cause you to not see a linear speed-up when running with 8 threads on beast.

20. Why did we see malloc() allocate more space than requested (via brk()) in the strace lab?

21. What performance characteristics are most important to a desktop operating system?

22. What is the input, output, and main job of each of the following:
    1. compiler
    2. linker
    3. loader

23. In the strace lab, we saw a number of mmap() commands. What does mmap() do and what data was being mmap()ed?

24. GDB (the debugger) works via the ptrace interface. What types of functionality do you think the ptrace interface must provide to gdb? (hint: man ptrace)

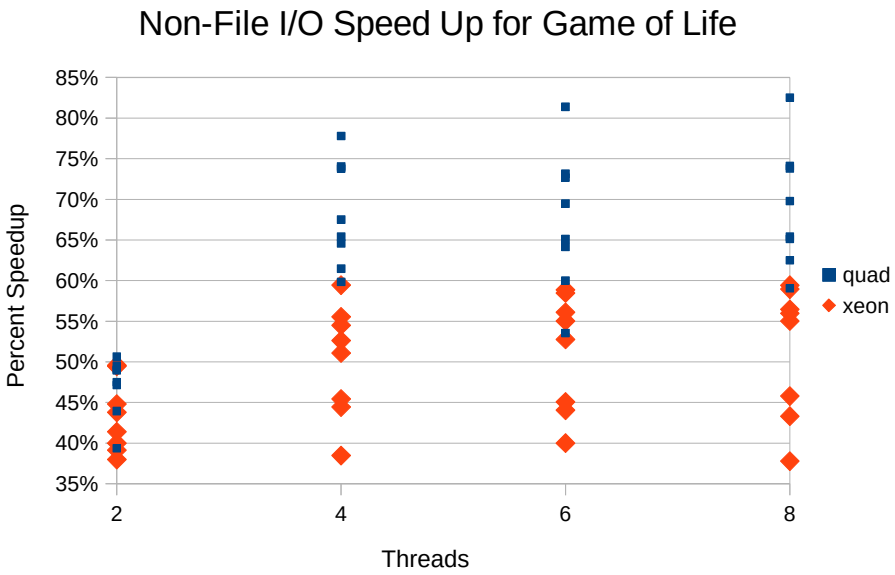
25. Read about the --trace-children command line option to valgrind in the man page. Explain why the statement in the man page "it would be difficult not to, since fork makes an identical copy of a process" is true. Explain how valgrind works and how this impacts how fork() works. Further, explain why extra work does need to occur to continue running Valgrind through an exec().

26. Describe a situation when a spin lock is advantageous.

27. Write your own question.

28. The operating system relies heavily on hardware support provided by the CPU. What role do each of these hardware pieces play in the OS and how would the OS need to be different (how would the functionality change, how would the implementation be more difficult, etc) if each piece did not exist?
    1. Dual mode bit
    2. TLB
    3. MMU
    4. timer interrupt

29. Whose job is it to keep the file descriptor tables synchronized?



30. The chart shown displays the non-File I/O speed up for the game of life for a previous iteration of this course. Higher is better (more speedup) on this chart. Each data point represents one student's reported result. Xeon is a dual CPU computer where each CPU is dual core. Quad is a quad core single CPU machine. Each machine is running an identical installation of OpenSUSE Linux.

    Xeon saw a speed up competitive with quad for 2 threads but there is a pretty clear separation happening with 4 to 8 threads. Discuss possible hardware and software reasons for this behavior.

    Make a note of any other data (regarding the hardware/operating system or the execution timings) that you feel would help you to better answer this question.

```
int gTotalSum = 0;
const int CHUNK = 1000;

void *runner(void *pParam)
{
  int start = (int*) pParam;
  int counter;

  for(counter = start; counter < start+CHUNK; counter ++)
  {
    gTotalSum += counter;
  }

}

int main()
{
  const int NUM_THREADS = 10;

  int counter = 0;
  pthread_t tid;

  for(counter = 0 ; counter < NUM_THREADS * CHUNK; counter += CHUNK)
  {
    pthread_create(&tid, NULL, runner, &counter);
  }


  printf("The sum: %d\n", gTotalSum);

}
```

31. Look at the really bad code above. This code attempts to sum the values from 0 to NUM_THREADS*CHUNK using NUM_THREADS. Describe each error in the code. Fix each error you find to give a correct solution.

32. Look at the i7 Cache layout linked on the class schedule web page. With your game of life, what do you think would happen, performance-wise, if row x was assigned to core 0 and row x+1 was assigned to core 6 on beast (we know core 0 and core 6 are on different CPUs). Be sure to explain how the data and the caches interact and what data is in which cache.