5. Suppose a variation of FFS includes in each inode 12 direct, 1 indirect, 1 double indirect, 2 triple indirect, and 1 quadruple indirect pointers. Assuming 6 KB blocks and 6-byte pointers

   a. What is the largest file that can be accessed via direct pointers only?

   b. To within 1%, what is the maximum file size this index structure can support?

6. On a Unix or Linux system, use the ls -l command to examine various directories. After the first ten characters that define each file's access permissions, there is a field that indicates the number of hard links to the file. For example, here we have two files, bar with two links and foo with just one.

```
drwxr-sr-x 2 dahlin prof 4096 2012-02-03 08:37 bar/
-rw-r--r-- 1 dahlin prof    0 2012-02-03 08:36 foo
```

   For directories, what is the smallest number of links you can observe? Why?

   For directories, even though regular users cannot make hard links to directories, you may observe some directories with high link counts. Why?

5. For a computer architecture with multi-level paging, a page size of 4 KB, and 64-bit physical and virtual addresses:

   a. List the required and optional fields of its page table entry, along with the number of bits per field.
   b. Assuming a compact encoding, what is the smallest possible size for a page table entry in bytes, rounded up to an even number.
   c. Assuming a requirement that each page table fits into a single page, and given your answer above, how many levels of page tables would be required to completely map the 64-bit virtual address space?

7. Of the following items, which are stored in the thread control block, which are stored in the process control block, and which in neither?

   a. Page table pointer
   b. Page table
   c. Stack pointer
   d. Segment table
   e. Ready list
   f. CPU registers
   g. Program counter

10. For a computer architecture with multi-level paging, a page size of 4 KB, and 64-bit physical and virtual addresses:

    a. What is the smallest possible size for a page table entry, rounded up to a power of two?
    b. Using your result above, and assuming a requirement that each page table fits into a single page, how many levels of page tables would be required to completely map the 64-bit virtual address space?

11. Suppose you are designing a system with paged segmentation, and you anticipate the memory segment size will be uniformly distributed between 0 and 4 GB. The overhead of the design is the sum of the internal fragmentation and the space taken up by the page tables. If each page table entry uses four bytes per page, what page size minimizes overhead?

13. Suppose a machine with 32-bit virtual addresses and 40-bit physical addresses is designed with a two-level page table, subdividing the virtual address into three pieces as follows:

| 10 bit page table number | 10 bit page number | 12 bit offset |

The first 10 bits are the index into the top-level page table, the second 10 bits are the index into the second-level page table, and the last 12 bits are the offset into the page. There are 4 protection bits per page, so each page table entry takes 4 bytes.

a. What is the page size in this system?
b. How much memory is consumed by the first and second level page tables and wasted by internal fragmentation for a process that has 64K of memory starting at address 0?

c. How much memory is consumed by the first and second level page tables and wasted by internal fragmentation for a process that has a code segment of 48K starting at address 0x1000000, a data segment of 600K starting at address 0x80000000 and a stack segment of 64K starting at address 0xf0000000 and growing upward (towards higher addresses)?

4. 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:		

13. Three tasks, A, B, and C are run concurrently on a computer system.

- Task A arrives first at time 0, and uses the CPU for 100 ms before finishing.
- Task B arrives shortly after A, still at time 0. Task B loops ten times; for each iteration of the loop, B uses the CPU for 2 ms and then it does I/O for 8 ms.
- Task C is identical to B, but arrives shortly after B, still at time 0.

Assuming there is no overhead to doing a context switch, identify when A, B and C will finish for each of the following CPU scheduling disciplines:

a. FIFO
b. Round robin with a 1 ms time slice
c. Round robin with a 100 ms time slice
d. Multilevel feedback with four levels, and a time slice for the highest priority level is 1 ms.
e. Shortest job first

5. Now suppose the computer system needs to support fault isolation. What hardware and/or operating support do you think would be needed to do the following?

a. Protect an application's data structures in memory from being corrupted by other applications.
b. Protecting one user's disk files from being accessed or corrupted by another user.
c. Protecting the network from a virus trying to use your computer to send spam.

6. How should an operating system support communication between applications? Explain your reasoning.
   a. Through the file system?
   b. Through messages passed between applications?
   c. Through regions of memory shared between the applications?
   d. All of the above?
   e. None of the above?

5. Define three types of user-mode to kernel-mode transfers.

6. Define four types of kernel-mode to user-mode transfers.

7. Most hardware architectures provide an instruction to return from an interrupt, such as iret. This instruction switches the mode of operation from kernel-mode to user-mode.

   a. Explain where in the operating system this instruction would be used.
   b. Explain what happens if an application program executes this instruction.

8. How many processes are created if the following program is run?

```
main(int argc, char ** argv) {
    forkthem(5)
}
void forkthem(int n) {
    if (n > 0) {
        fork();
        forkthem(n-1);
    }
}
```

9. Consider the following program:

```
main (int argc, char ** argv) {
    int child = fork();
    int x = 5;

    if (child == 0) {
        x += 5;
    } else {
        child = fork();
        x += 10;
        if(child) {
            x += 5;
        }
    }
}
```

How many different copies of the variable x are there? What are their values when their process finishes?

10. What is the output of the following programs? (Please try to solve the problem without compiling and running the programs.)

```
// Program 1
main() {
    int val = 5;
    int pid;

    if (pid = fork())
        wait(pid);
    val++;
    printf("%d\n", val);
    return val;
}

// Program 2:
main() {
    int val = 5;
    int pid;
    if (pid = fork())
        wait(pid);
    else
        exit(val);
    val++;
    printf("%d\n", val);
    return val;
}
```

Questions

1. Compute the response time and turnaround time when running three jobs of length 200 with the SJF and FIFO schedulers.
2. Now do the same but with jobs of different lengths: 100, 200, and 300.
3. Now do the same, but also with the RR scheduler and a time-slice of 1.
4. For what types of workloads does SJF deliver the same turnaround times as FIFO?
5. For what types of workloads and quantum lengths does SJF deliver the same response times as RR?
6. What happens to response time with SJF as job lengths increase? Can you use the simulator to demonstrate the trend?
7. What happens to response time with RR as quantum lengths increase? Can you write an equation that gives the worst-case response time, given N jobs?