Assignment 5
Due date: Nov. 13th, 23:59

Each question is worth 1 point.

1. Some DMA controllers support direct virtual memory access, where the targets of I/O operations are specified as virtual addresses and a translation from virtual to physical address is performed during the DMA. How does this design complicate the design of the DMA controller? What are the advantages of providing such functionality?

2. One way to use contiguous allocation of the disk and not suffer from holes is to compact the disk every time a file is removed. Since all files are contiguous, copying a file requires one seek and rotation to read the file, followed by the transfer at full speed. Writing the file back requires the same work. Assuming a seek time of 5 ms, 7200 RPM disk with a transfer rate of 80 MB/sec, and an average file size of 8 KB.

   a. How long does it take to read a file into main memory and then write it back to the disk at a new location?
   b. Using these numbers, how long would it take to compact half of a 16-GB disk?
   c. Does compacting the disk ever make any sense?

3. A disk may have multiple surfaces, arms, and heads, but when you issue a read or write, only one head is active at a time. It seems like one could greatly increase disk bandwidth for large requests by reading or writing with all of the heads at the same time. Given the physical characteristics of disks, can you figure out why no one does this?

4. Suppose that a disk drive has 5000 cylinders, numbered 0 to 4999. The drive is currently serving a request at cylinder 143, and the previous was at cylinder 125. The queue of pending requests, in FIFO order, is:

   86, 1470, 913, 1774, 948, 1509, 1022, 1750, 130

Starting from the current head position, what is the total distance (in cylinders) that the disk arm moves to satisfy all the pending requests for FCFS, SSTF, SCAN, C-SCAN.

5. How many disk operations are needed to fetch the i-node for a file with the path name /usr/zzk/courses/os/xyz.txt? Assume that the i-node for the root directory is in memory, but nothing else along the path is in memory. Also assume that all directories fit in one disk block.

6. Consider a UNIX-style i-node with 10 direct pointers, one single-indirect pointer, one double-indirect pointer, and one triple-indirect pointer. Assume that the block size is 4K bytes, and that the size of a pointer is 4 bytes.

   a. How large a file can be indexed using such an i-node? Explain your answer in detail.
   b. Assume that the OS has already read the i-node for a file into main memory (i.e., the file buffer cache). How many disk reads are required to read data block number 800 into memory? Explain your answer in detail.

7. Consider a system that supports 5000 users. Suppose that you want to allow 4990 of these users to be able to read and write one file.

a. How would you specify this protection scheme in Linux?
b. What if you want to allow 10 out of the 4990 to be able to execute that file? How would you specify this protection scheme then?

8. Some operating systems provide a system call rename to give a file a new name. Is there any difference at all between using this call to rename a file and just copying the file to a new file with the new name, followed by deleting the old one?

9. Consider a disk that has 10 data blocks starting from block 14 through 23. Let there be 2 files on the disk: f1 and f2. The directory structure lists that the first data blocks of f1 and f2 are respectively 22 and 16. Given the FAT table entries as below, what are the data blocks allotted to f1 and f2?

(14,18); (15,17); (16,23); (17,21); (18,20); (19,15); (20, −1); (21, −1); (22,19); (23,14)

In the above notation, (x, y) indicates that the value stored in table entry x points to data block y.

10. The beginning of a free-space bitmap looks like this after the disk partition is first formatted: 1000 0000 0000 0000 (the first block is used by the root directory). The system always searches for free blocks starting at the lowest-numbered block, so after writing file A, which uses six blocks, the bitmap looks like this: 1111 1110 0000 0000. Show the bitmap after each of the following additional actions:

a. File B is written, using five blocks.
b. File A is deleted.
c. File C is written, using eight blocks.
d. File B is deleted.