File System Guide

Given a file system with:

- 4KB block size
- 6 direct block pointers
- 2 single indirect pointers
- 1 double indirect pointer

How do you calculate the maximum file size?

1. Calculate the amount of data accessible by each of the direct and indirect pointers.
    a. From direct pointers = **BLOCK SIZE x NUM DIRECT POINTERS**
    b. Calculate number of pointers that fit in a block = **BLOCK SIZE / POINTER SIZE**
    c. From single indirect pointers = **BLOCK SIZE x NUM POINTERS PER BLOCK x NUM SINGLE INDIRECT POINTERS**
    d. From double indirect pointers = **BLOCK SIZE x (NUM POINTERS PER BLOCK)^2 x NUM DOUBLE INDIRECT POINTERS**
    e. From nth indirect pointers = **BLOCK SIZE x (NUM POINTERS PER BLOCK)^n x NUM NTH INDIRECT POINTERS**

Example:
Note that 2^10 = 1024
From Direct Pointers = 4KB * 6 = 24KB
Number of Pointers per Block = 4KB / 4 bytes = 1024
From Single Indirect = 4KB * 1024 * 2 = 8192KB
From Double Indirect = 4KB * 1024^2 * 1 = 2^2 * 2^20 = 2^22KB

2. Sum the values for each type of pointer.

Example:
Maximum File Size = Direct + Single + Double
= 24 + 8192 + 2^22KB
= 8216 + 2^22KB

3. Remember that addressing bytes of a file starts at 0. Therefore, the bytes that can be found through direct pointers are: [0, NUM DIRECT PTRS * BLOCK SIZE - 1].

Opening a File for Read or Write

1. Find the i-number of the file; open its i-node and check permissions, etc.
    a. Start by opening the root i-node to retrieve a pointer to the roots data, which is where the hard links can be found.
    b. Open roots data block, retrieve the i-number of the next element of the path.

    c. Repeat this process until the i-number for the file is found.

Example: Open /usr/bin/lua51 for reading.

Read root i-node.
Read root data; retrieve usr i-number.
Read usr i-node.
Read usr data; retrieve bin i-number.
Read bin i-node.
Read bin data; retrieve lua51 i-number.
Read lua51 i-node.

Creating Files

1. To create a file; find the directory where the file will “live” and add a hardlink to the new file; then create the i-node for the new file
    a. Find the i-number of the directory that will contain the file; open the i-node and read the data block.
    b. Read the i-bitmap; find an unused bit N (bit[N] == 0); set bit to 1; N is the new i-number
    c. Read the i-node corresponding to N; write metadata to it
    d. Write the i-node of the containing directory to indicate it was modified (a hardlink was added!)
2. To write NEW data to a file
    a. If a new direct block is needed; read the data-bitmap and search for free block; then write to that block
    b. If a new single indirect block is needed; and there is no block of pointers yet
        i. Read the data-bitmap and find TWO blocks; one for the block of pointers AND one for the data
        ii. Write to BOTH blocks

Example: Create file /usr/bin/blah.txt

Read root i-node.
Read root data; retrieve usr i-number.
Read usr i-node.
Read usr data; retrieve bin i-number.
Read bin i-node.
Read bin data
Read i-node bitmap; find unused inode
Write i-node bitmap
Write bin data; add hardlink for blah.txt
Read i-node for blah.txt

Write i-node for blah.txt; adding metadata

Write bin i-node; update last edited time

Questions

1. How many i-nodes are Read and Written if we execute “ls /usr/bin/local”, where local is a directory?
2. Does “ls -la x.txt” read any of x.txt’s data?
3. What must be Read and Written to delete the file /usr/bin/foo.txt?