CS111 Week 8 Discussion Notes:

The goal is to produce the same output. Even if things are wrong.
Make sure that there is nothing wrong in the logging.
The code needs to be finished within reasonable amount of time.

Test cases can be found online.
https://web.cs.ucla.edu/classes/cs111/Samples/

The trivial.img file is different from the version online.
Also the EXT2_img is another different test case.
Follow the definition and print all the corresponding output.

No make check in the makefile.

Part 1 TA said it is useless. HEHE
You can use Beaglebone for sudo linux (HEHEHEHHEE)

File system divides image into blocks, following some conventions.

File types: they are all files.

File structure:

Superblock:

Group Summary:
Help to find information about current group

Bitmap(shortcut for checking if the specific data block is in use.
Bitmap Botmap

Inode data block
Provide basic information about the file
Inode points to one or more data blocks

char* p = malloc(128); //heap
If you don't deallocate this memory, the memory will always be there

Char array[128]; //stack
If you go outside the scope of this program, the os will automatically deallocate this memory.

pread read and write from a file descriptor at a given offset

Int fd = open(“trivial.img”, *read_only*);
char* buffer = malloc(1024);
pread(fd, buf, 1024, 1024);

Type cast the buffer to output the

ext2_superblock * sp = (ext2_superblock) buf;
printf(“%d”, sp);

The main purpose of this project is to read the block (the smallest unit).

Total number of blocks: 500
blocks per group: 1000
There are two groups in total, what is the size of the group?

Blocks per group is truly except for the last group, which always has the remaining blocks.

Free block bitmap == block bitmap

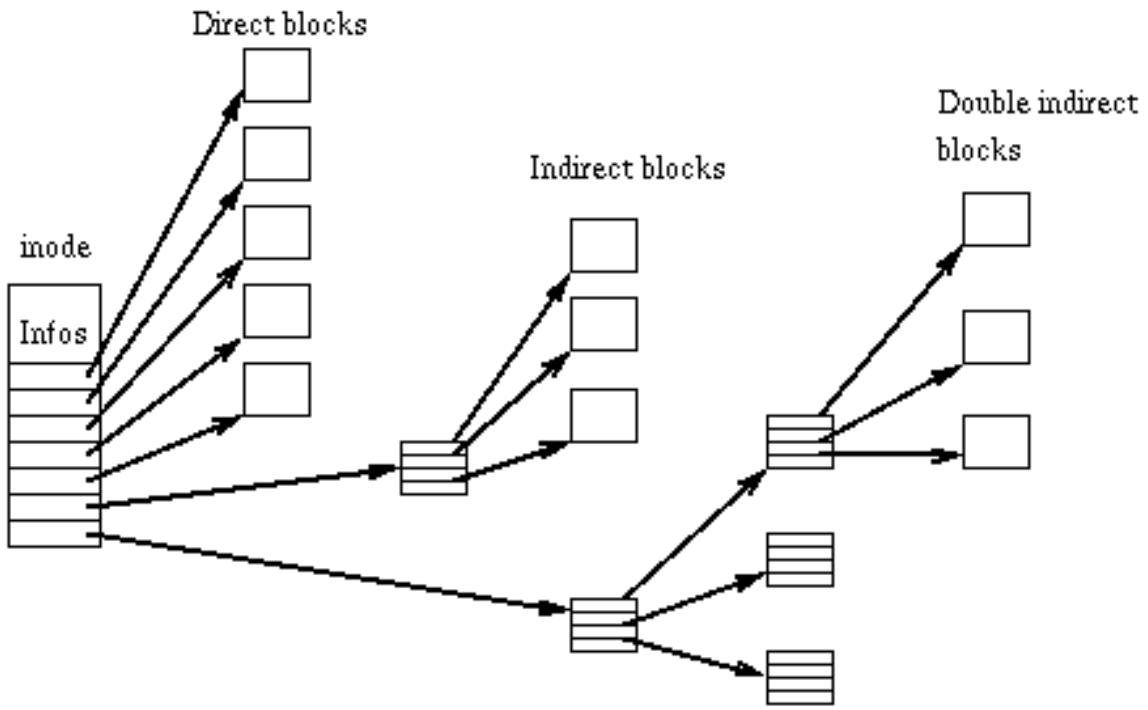
There will only be a single group.

In C, when you increment the pointer by 1, it automatically increments by one data structure in memory.

In order to read a bit, first read a byte and then do bit manipulation.
Need to figure out where the bitmap is. (It is not in block 3)

This is correct:
Symbolic links are a little more complicated. If the file length is less than the size of the block pointers (60 bytes) the file will contain zero data blocks, and the name is stored in the space normally occupied by the block pointers. If this is the case, the fifteen block pointers need not be printed.

Fifteen pointers:



- Twelve pointers that directly point to blocks of the file's data (direct pointers)
- One singly indirect pointer (a pointer that points to a block of pointers that then point to blocks of the file's data)
- One doubly indirect pointer (a pointer that points to a block of pointers that point to other blocks of pointers that then point to blocks of the file's data)
- One triply indirect pointer (a pointer that points to a block of pointers that point to other blocks of pointers that point to other blocks of pointers that then point to blocks of the file's data)

How to find all the directory entries?
Simply scan the through the inode. Scan every possible data blocks. Stop after scanning through all the blocks.
Adding all the offsets to get the next entry.

In C, if we have an array, when printing a string like this:
c[2] = {‘a’, ‘b’};
printf(“%s”, a);

C does not have boundary checking. It will print until it reaches a null byte.
Output:
ab…….(bunch of garbage information) until the null terminating byte.