11/16 Filesystem

→ name to data (storage devices)
  → read/write, create/delete, specify & enforce permission for data

→ how to materialize these abstractions/features?
  1) Where do we place data on disk?
  2) How to track information about data?
  3) How to name & organize data?

Data Placement (Q1)

→ need to map data blocks to disk
→ require us to know usage info of all disk blocks
→ hundreds of millions of blocks to track
  ↳ tracks in unit of multiple blocks
  ↳ use bitmap to track usage (1 bit for each block)
    0 = free, 1 = used

→ where to store bitmaps?
  ↳ disk, bitmap lives in disk blocks & track those blocks as used.
→ how do we find the bitmap?
  ↳ known location

Manage Data (Q2)

→ What information do we need to track?

↳ size, owner, permission, location of data blocks

↳ metadata (also called file header, file record, inode)

→ Where does metadata live?

↳ on disk! inode array (reserved section on disk that stores inodes)

↳ track start of inode array

↓

hmm... how do we know which entries are allocated and which are free?

☆ lots of things to track at known location, can we group them?

→ Superblock (metadata for filesystem)

both needs to be at known loc.

split our bitmap into
① inode bitmap
② data bitmap

↳ keep this at a known loc.

↳ stores info about the bitmaps & inode array & other metadata

Naming & Organize Data (Q3)

→ Option 1: random strings (hash, no organizational info

→ Option 2: user defined names & path (orgnizational info)

→ file: user defined name to data

→ directories: group files

↳ implemented as file (type dir)

↳ data = directory entries, each directy { name, inumber (index into the inode array) }

→ where do we find directory/file on disk?

↳ directory entry of parent dir

↳ what about the top level directory?

☆ special case: root dir, its inode is at a known loc.

Path Traversal for /home/tom/foo.txt.

① open inode for root directory
use inode to find data for root

dir entries

name	inum
home	158

② read root's data

③ inode for home is at index 158 in the inode array, open inode 158 & find data for home

tom	830

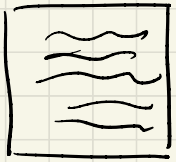
④ read home's data

⑤ open inode 830 & find data for tom

foo.txt	871

⑥ read tom's data

⑦ open inode 871, find data for foo.txt



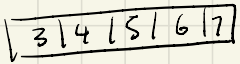
data for foo.txt

Data Layout

→ how metadata tracks location of data depends on the data layout

→ basic approach

→ contiguous allocation



- allocate consecutive blocks
- track start blk & # of blocks

→ linked allocation

- allocate blocks, each block has data & a ptr to the next block
- just track the starting block

→ indexed allocation