CSE 330: Operating Systems

Class: 26 **Date:** 11/22

Fall 2016

Note Title

File Systems

- Application — API
  - System Calls

- Open, close, read, write, create, delete ...

Unix/Linux System calls

fd = open (filename, options)

fd ↑ file descriptor

filename → does it exist? permissions? well formed?

filename → just a name (current dir)

→ full path name

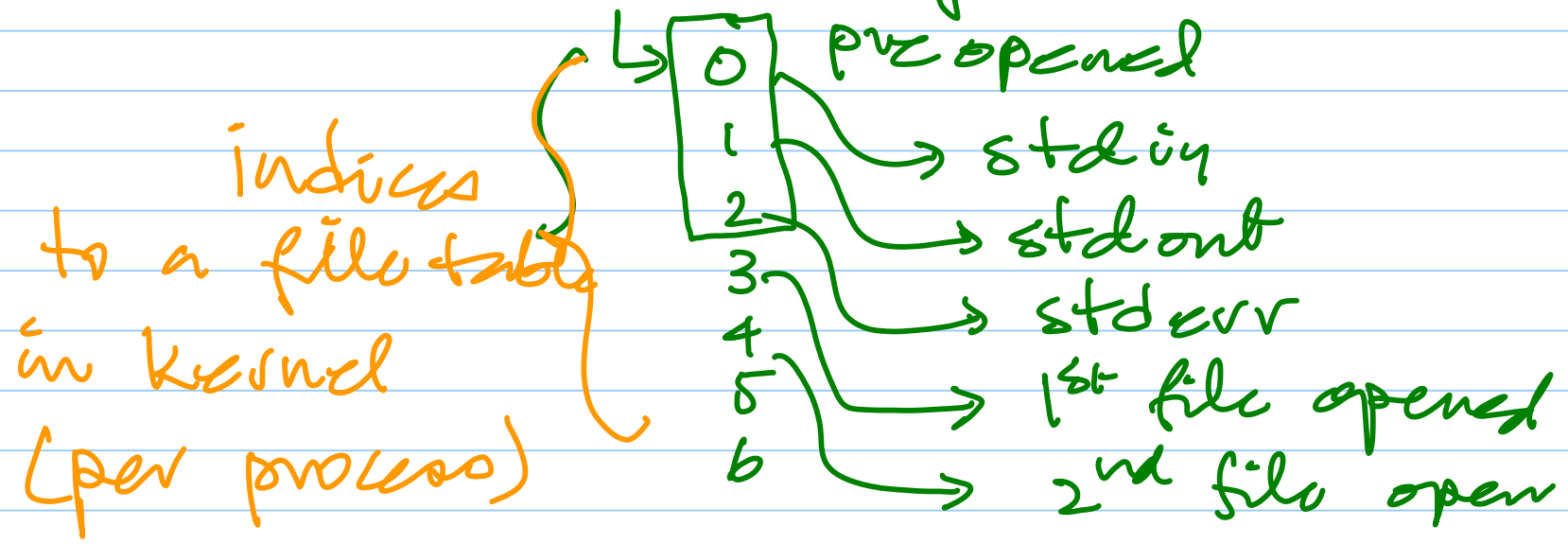
→ partical file path.

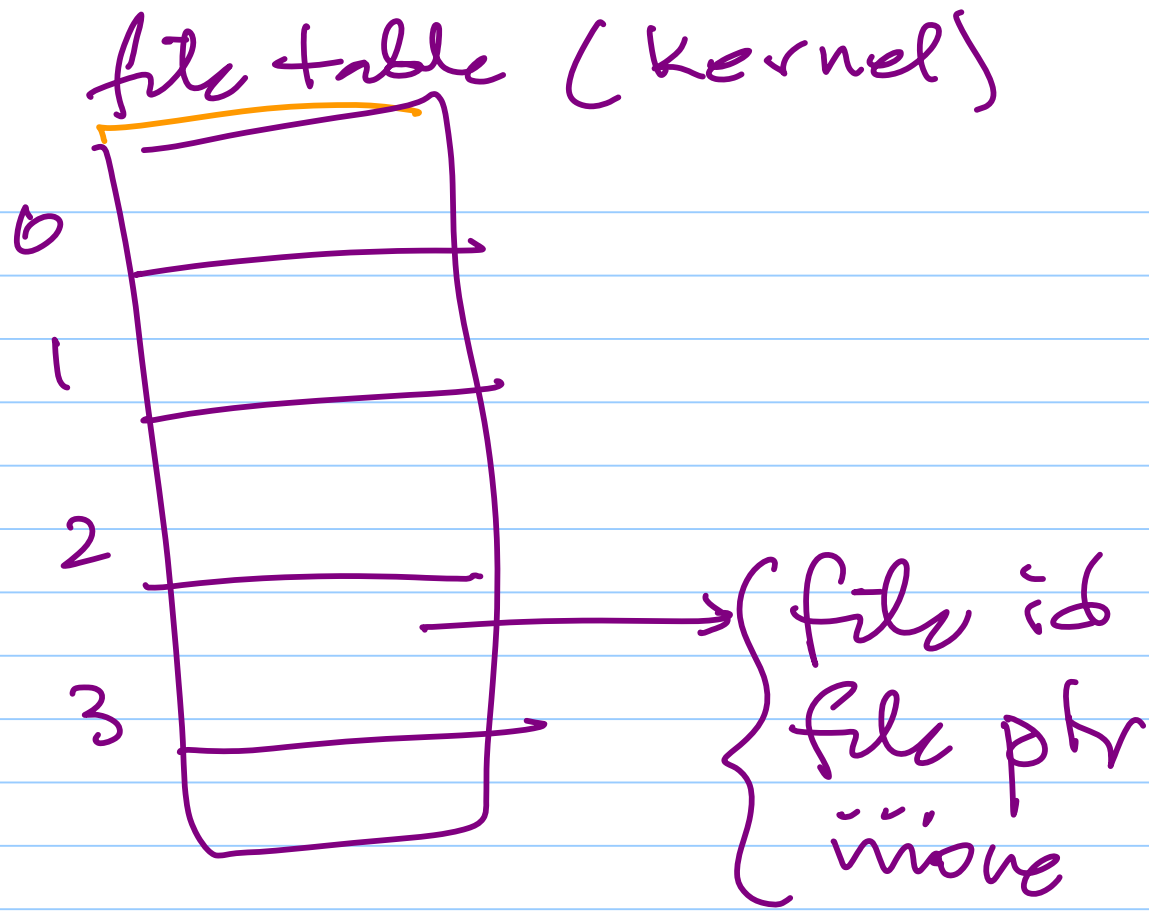
↳ exist → read or write

↳ does not → maybe create

open returns a file descriptor (fd)

fd is a "small integer"

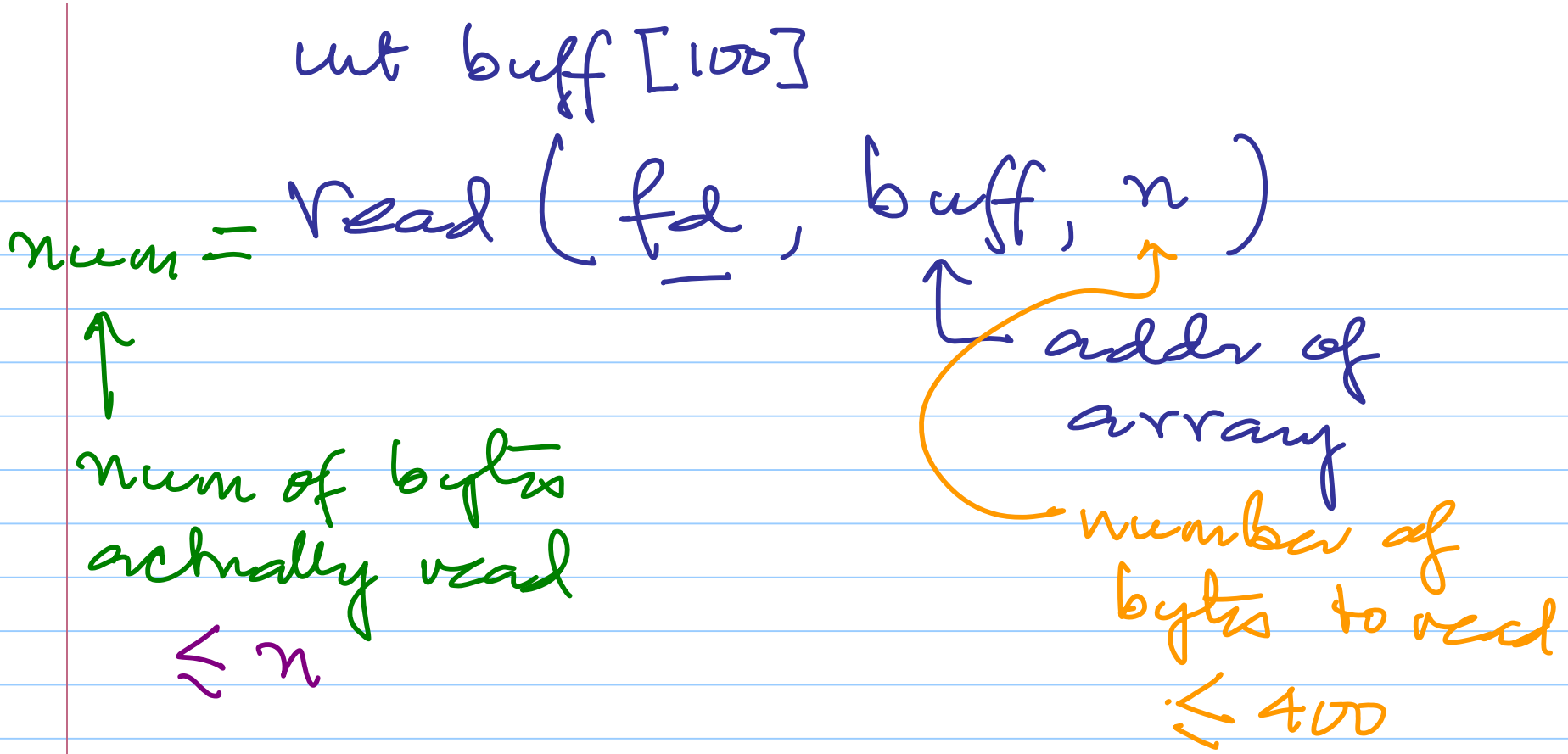



file table (kernel)
0
1
2
3
file id
file ptr
...
inode

Create a file

use open with

→ O_CREAT flag


int buff[100]
num = read (fd , buff, n)
addr of array
number of bytes to read
≤ 400
num of bytes actually read
≤ n
-1 error

num = 0 → all data have been read (@ EOF)

> 0 but < n

⇒ reached EOF

→ " EOL (for stdin)

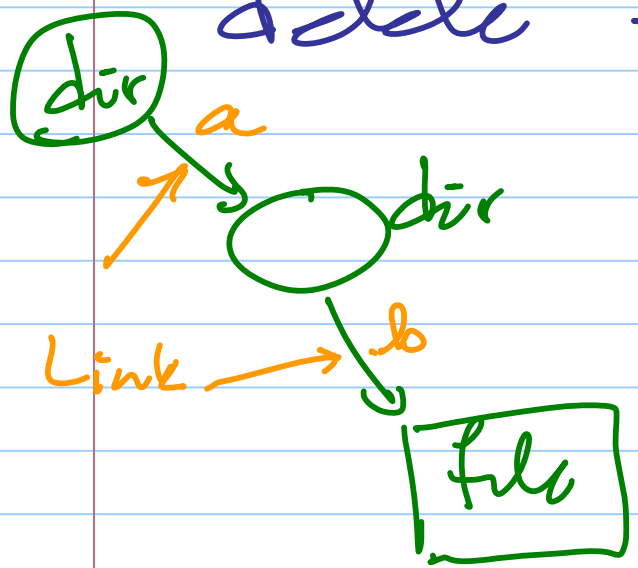
num = write (fd, buff, n)

same as read (almost)

num < n ⇒ something wrong

create → flag in open

delete → unlink (filename)



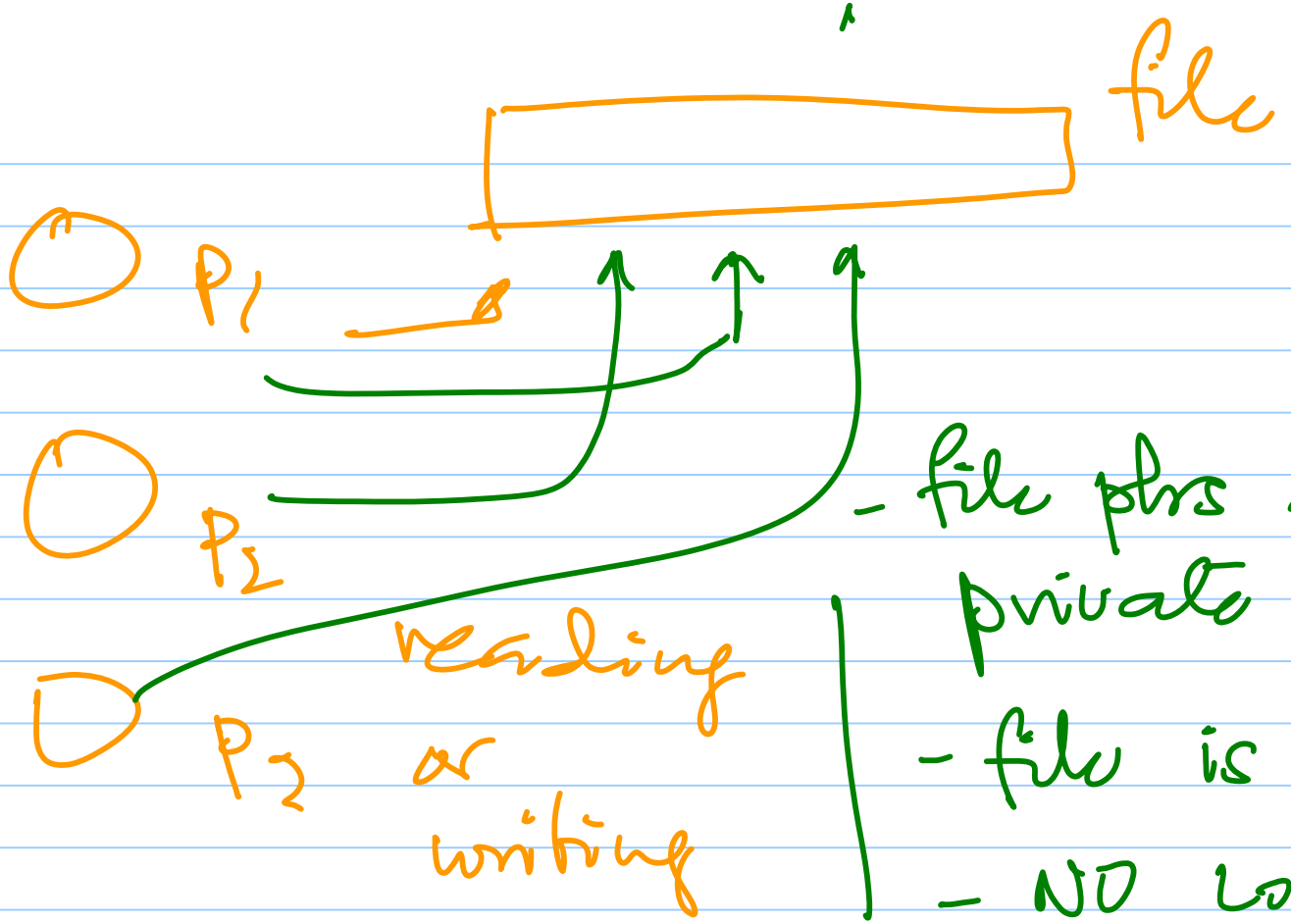
filenames in UNIX are links

Concurrency

↳ multiple processes, multiple files

→ multiple processes, 1 file

→ causes conflicts inside the file system



- file ptrs are private
- file is shared
- NO locking

create / delete same file

by > 1 process

atomic

- files are deleted after it is closed → all blocks added to free list
  - if no other link exists

close a file ... why?

↳ files are closed (auto) when a process terminates

→ there is a limit of # of open files / process.

File System implementation

- disk → huge # of blocks.

1TB disk

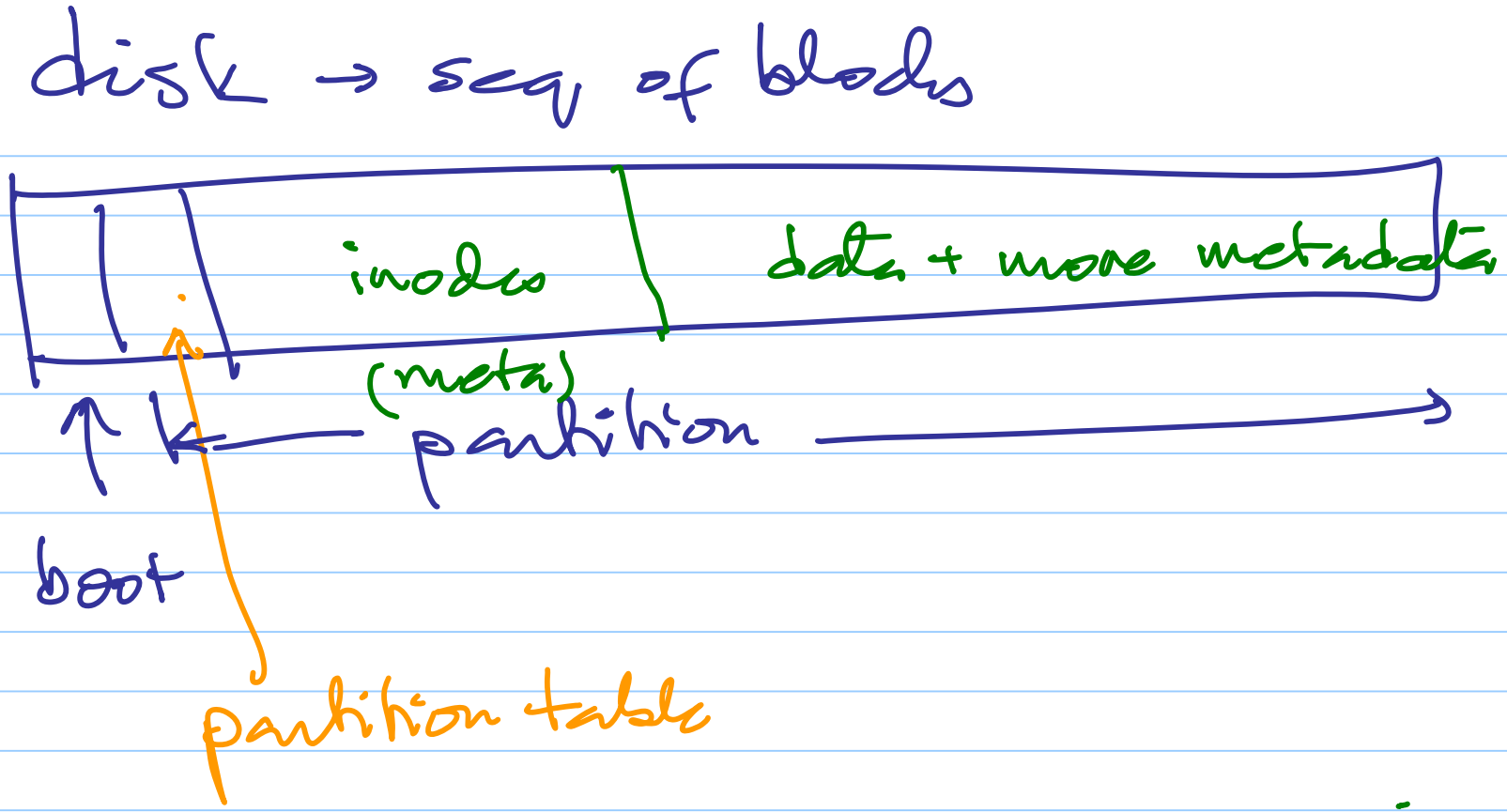
→

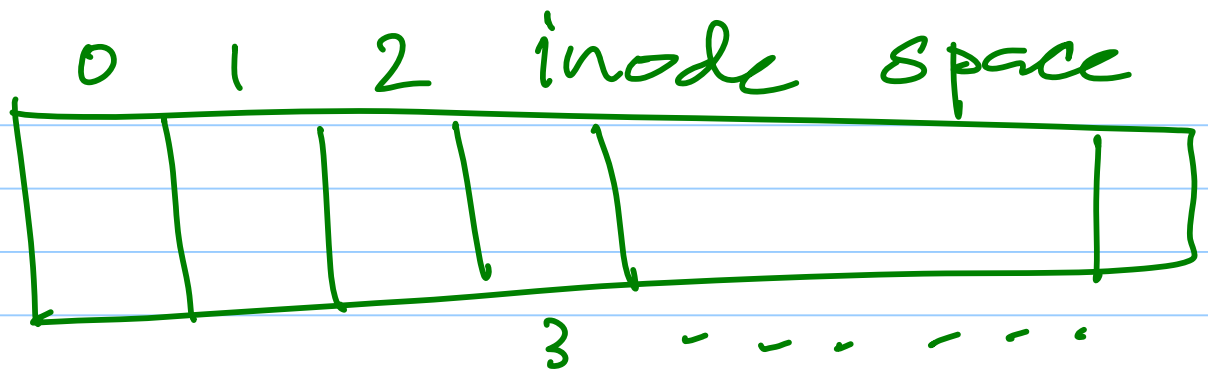
1kB/block

	bytes/block
std	512
adv	4096

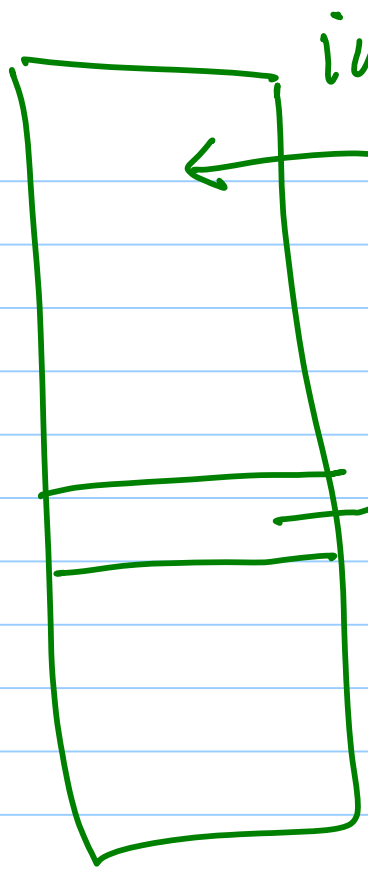
File system manage data on a disk

- data in files (payload) (user data)
- data about the data in files → metadata


disk → seq of blocks
inodes
data + more metadata
(meta)
partition
boot
partition table



inode → . fixed size, one per file
. uniquely ids the file / dir
. contains file properties

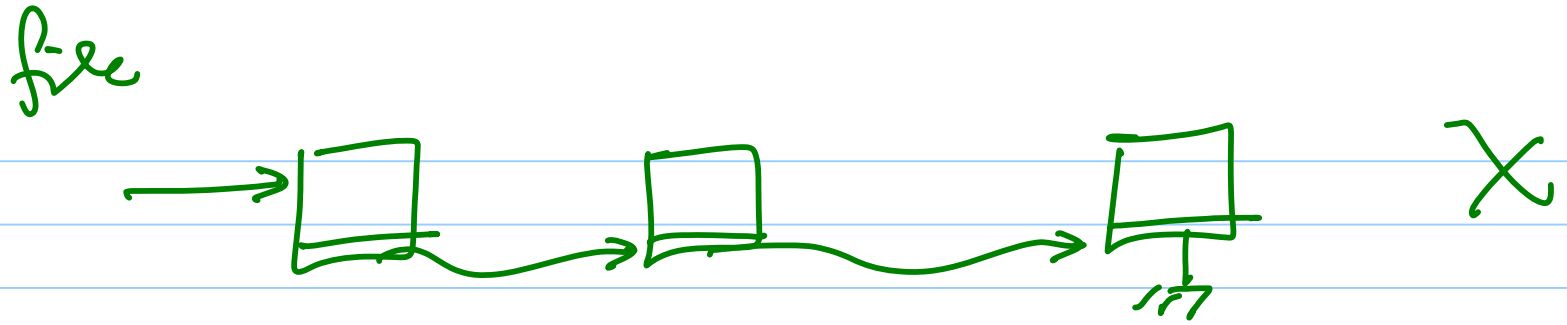


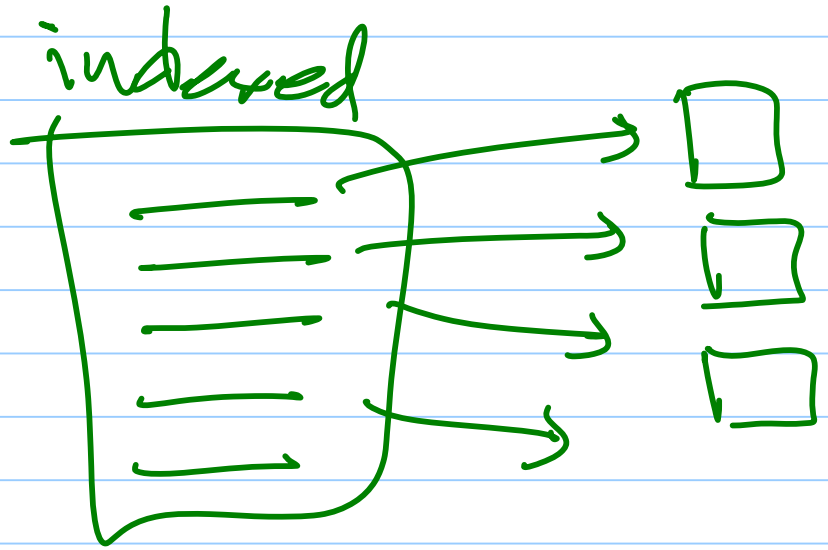
inode

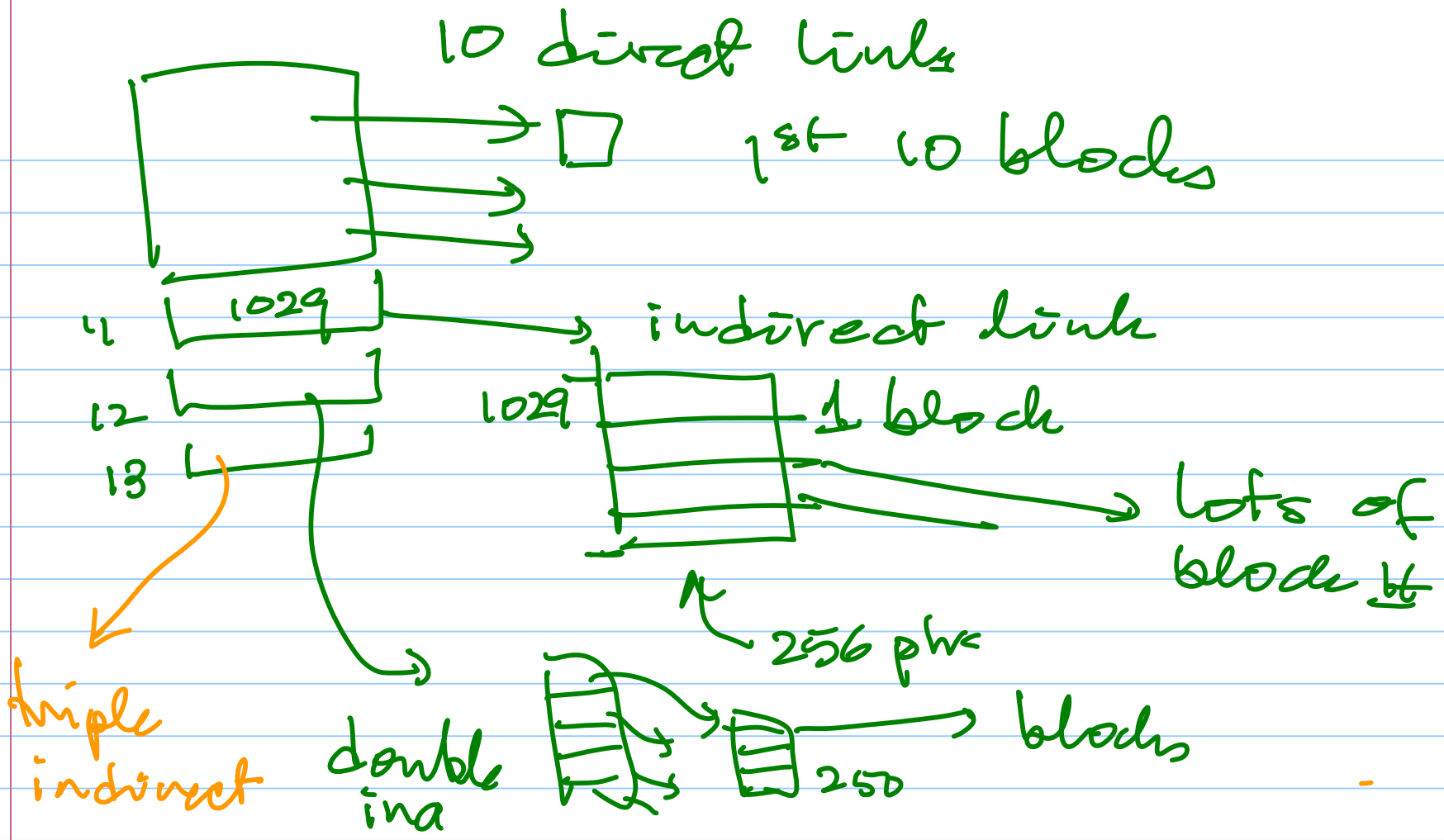
size, timestamps, owner, permissions, no of links, file/dir?,

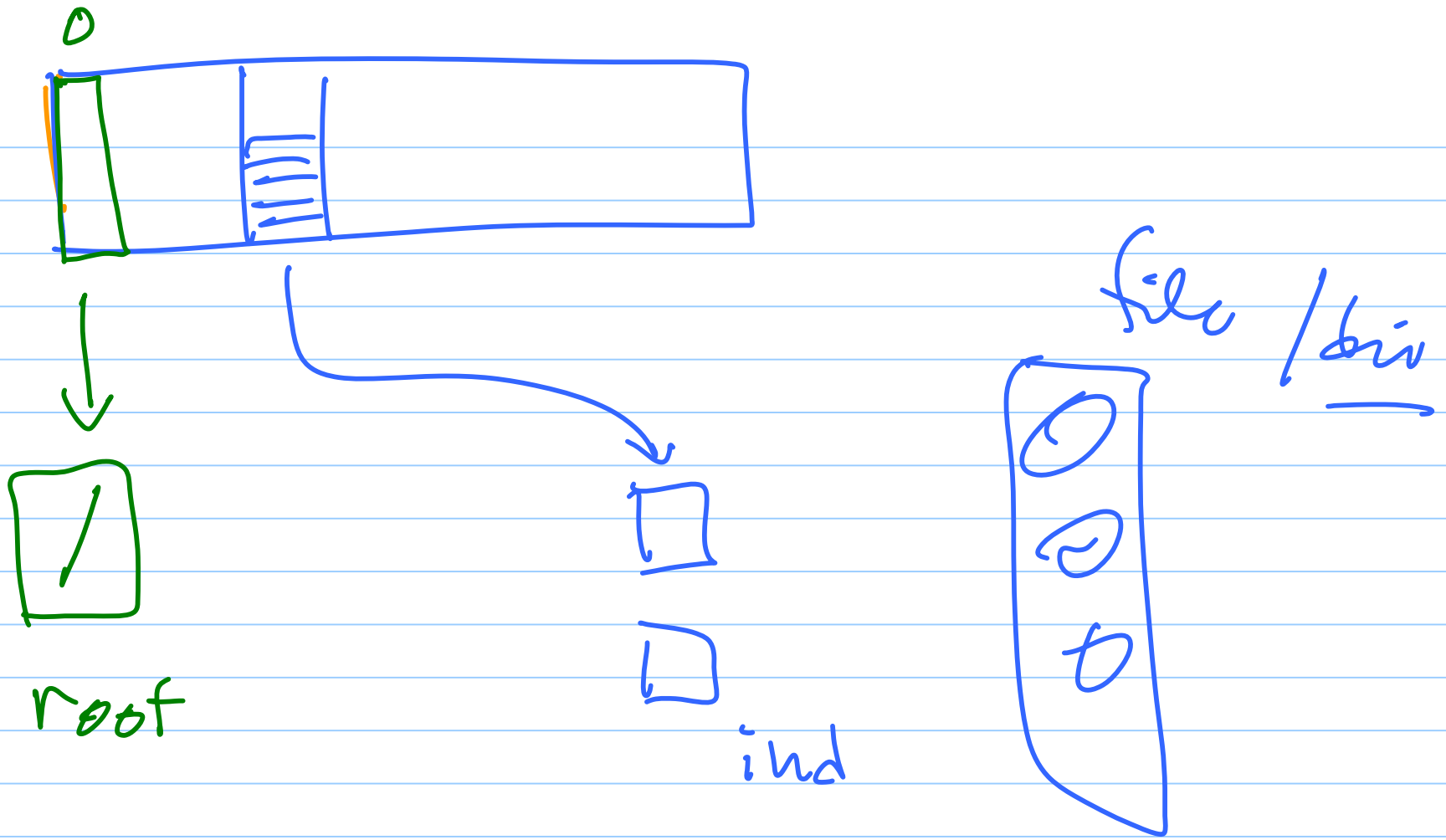
13 disk block address

↳ 1st 10 are block #s of the 1st 10 blocks of the file.


file


indexed


10 direct links
1st 10 blocks
11
1029
12
13
indirect link
1029
1 block
lots of
blocks
256 ptrs
triple
indirect
double
ina
250
blocks


0
/
root
ind
file /bin
0
1
b