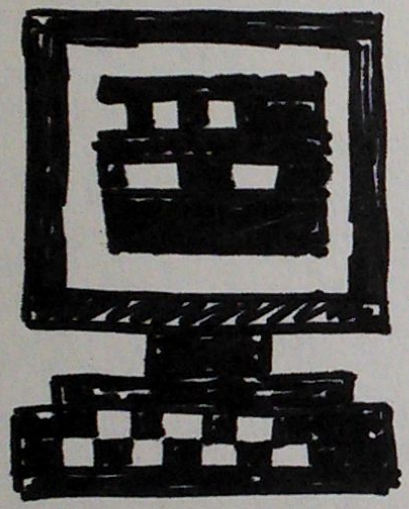
PROGRAMMING EXPERIMENTS

RULES OF PROGRAMMING EXPERIMENTS

1. it doesn't have to work
2. you don't have to finish it
3. you have to learn something

the Recurse Center

New York!
12 weeks!
free!

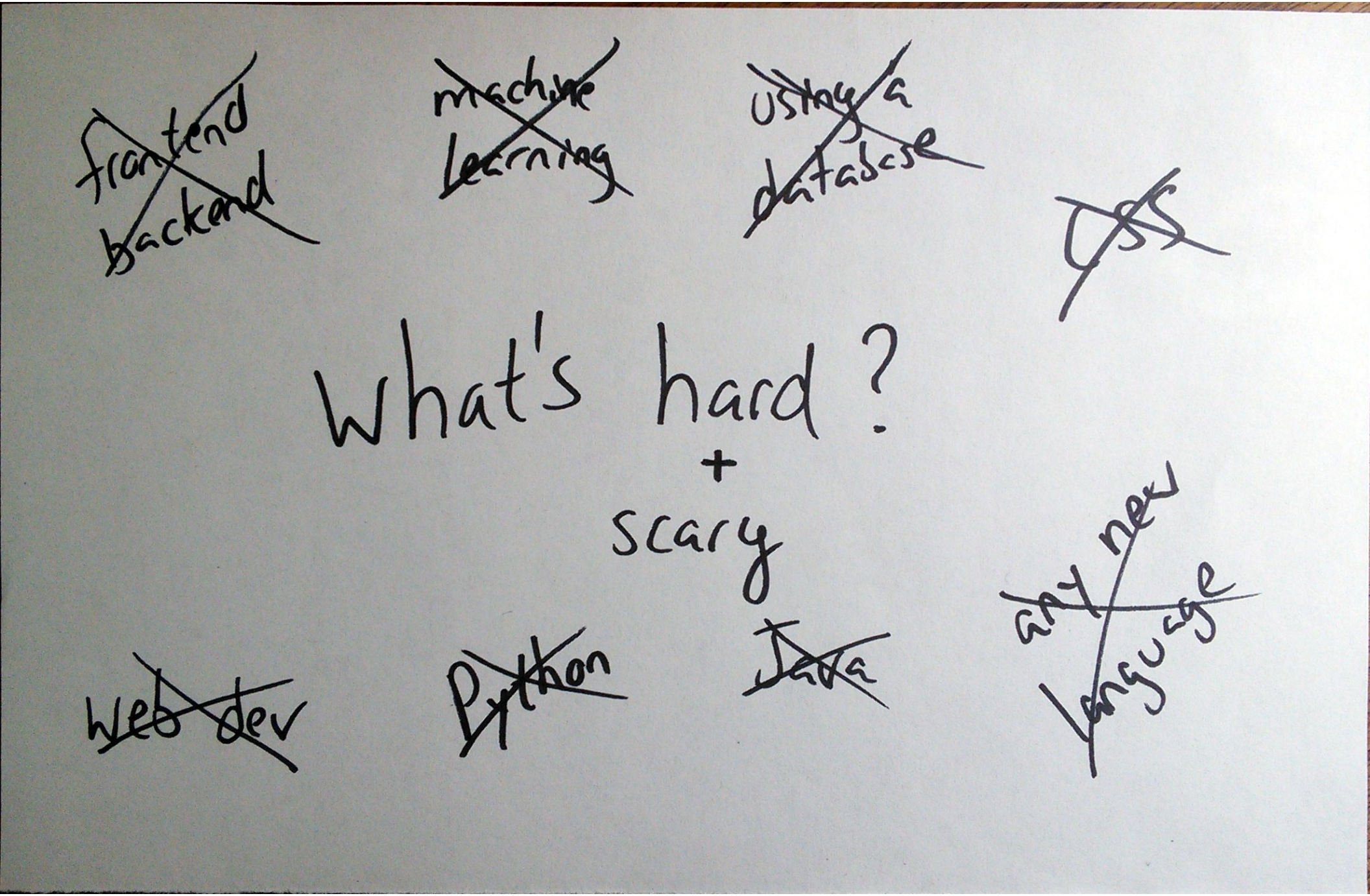


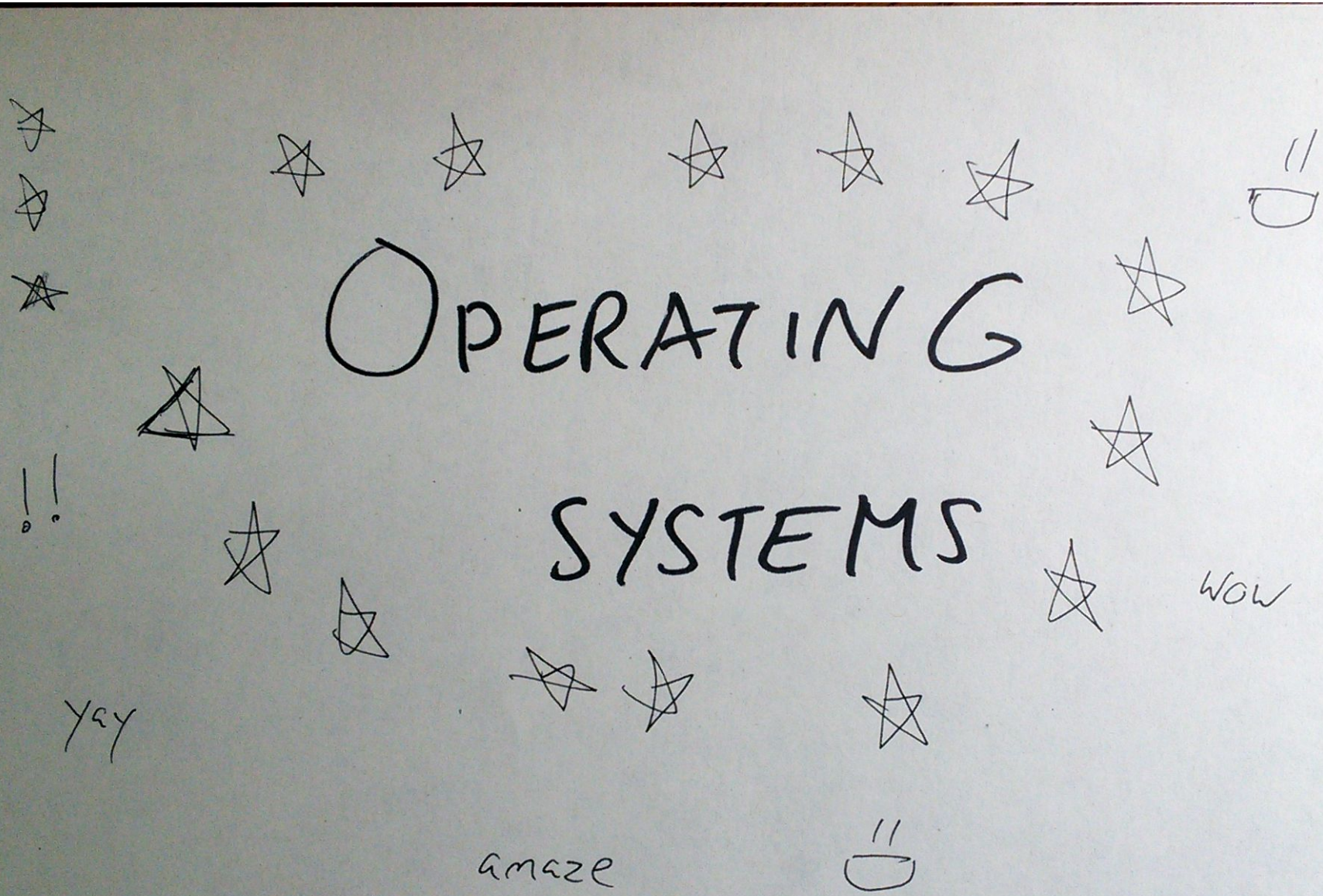
the best
programming
community
in the world

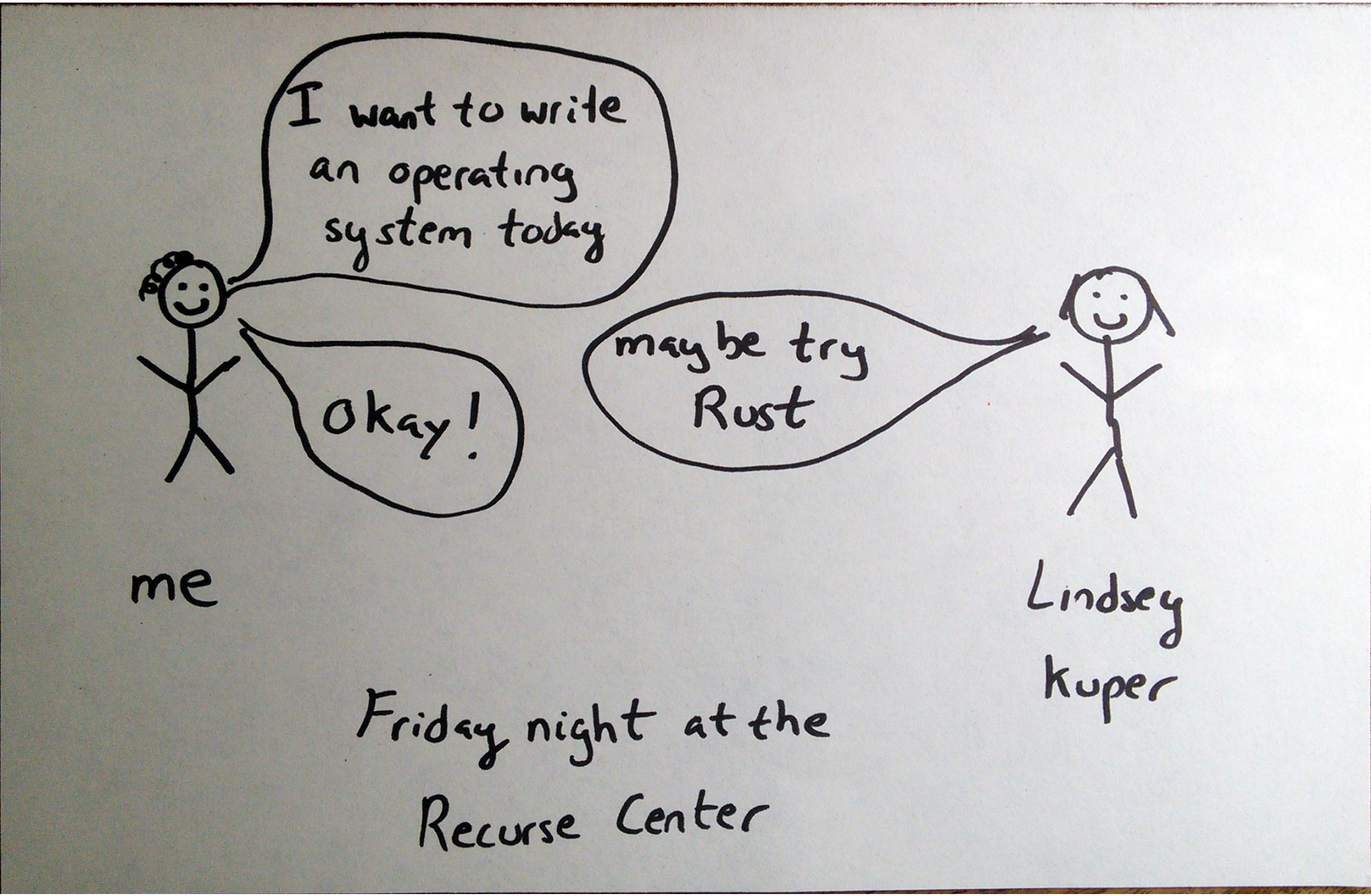
recurse.com

EXPERIMENT 1: WRITE AN OPERATING SYSTEM

remember it doesn't have to work


frontend
backend
machine learning
using a database
CSS
what's hard?
+
scary
web dev
Python
Java
any new language


OPERATING
SYSTEMS
!!
yay
amaze
WOW


I want to write an operating system today
Okay!
me
maybe try Rust
Lindsey Kuper
Friday night at the Recurse Center

Rust lets you write your own

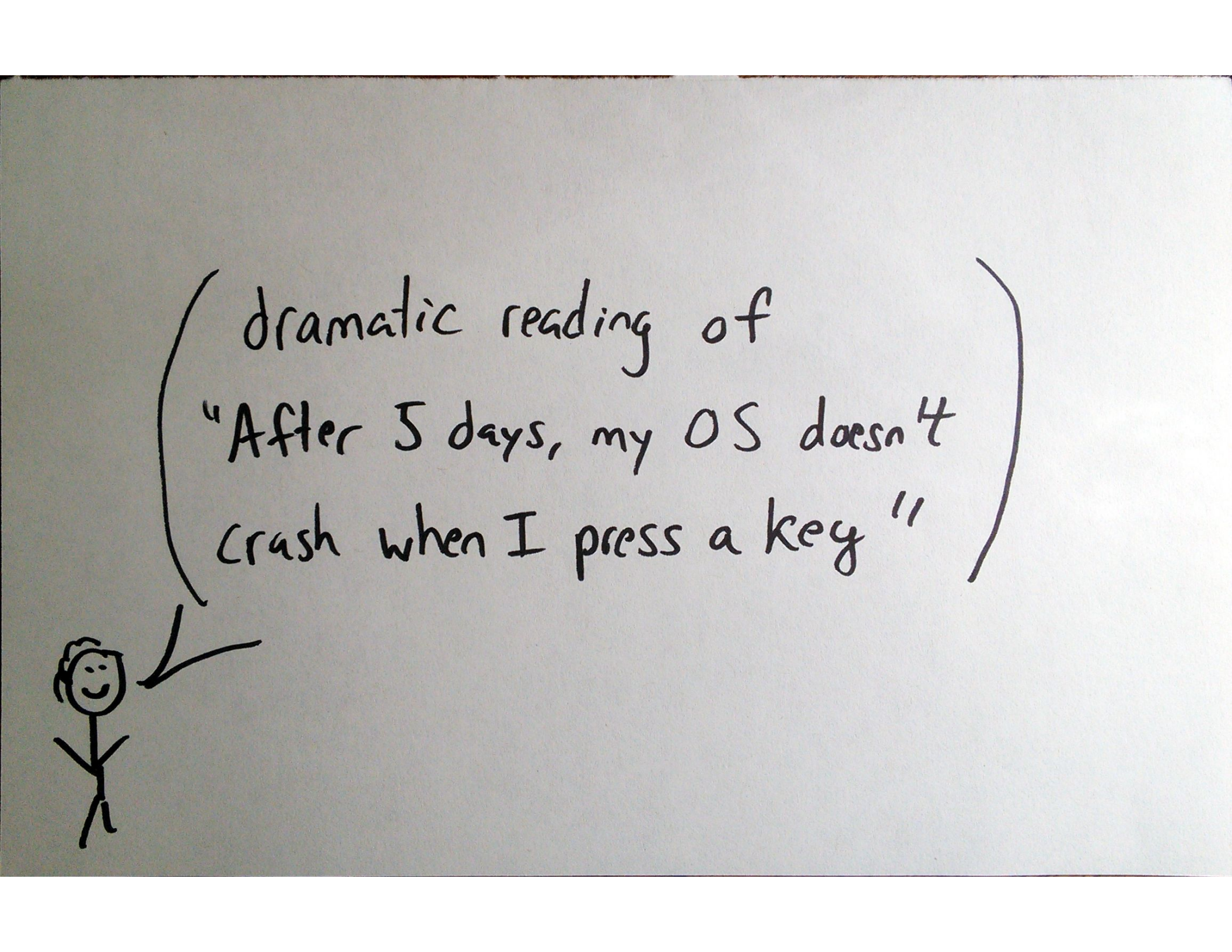
- malloc etc
- threads etc
- I/O

"~~freestanding mode~~"
libcore

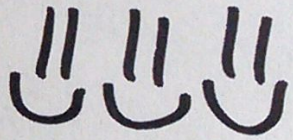
things you do not have when you write a kernel

- a hard drive
- files
- other processes
- a working keyboard
- memory protection
- any I/O
- libraries to use
- ANYTHING !!

"I'm just going to write a keyboard driver"


(dramatic reading of
"After 5 days, my OS doesn't
crash when I press a key")

remember how we don't

have malloc? 

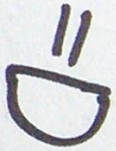
```
fn malloc(size: i32) {
    return 42;
}
```

now we have our weird malloc

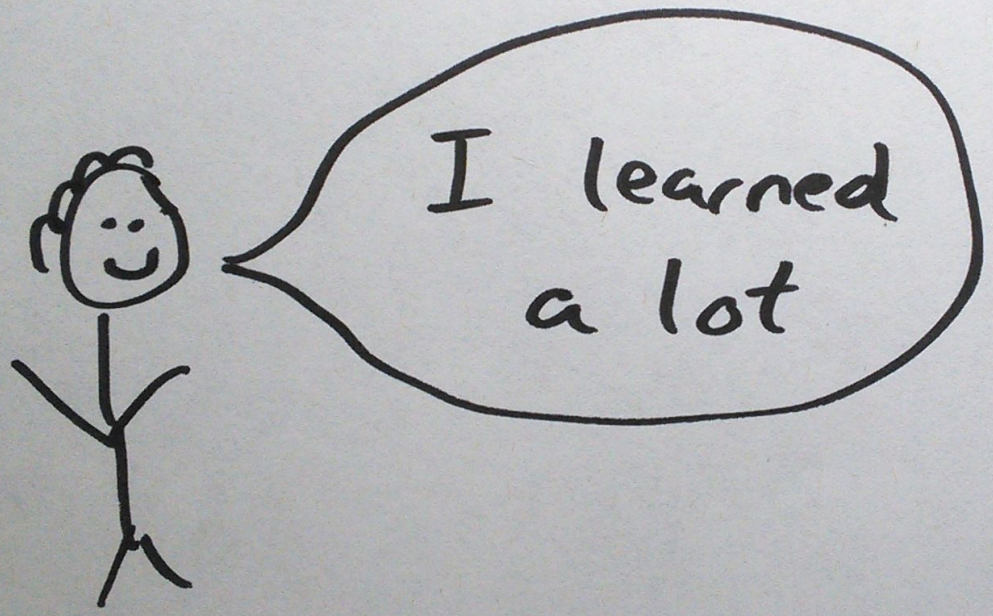
```
let x = Box::new(5)
let y = Box::new(88)
```

x=5

now x = 88



after 3 weeks



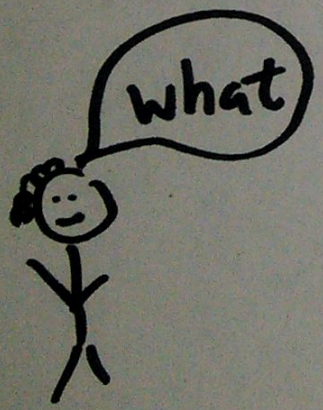

I'm trying to write a kernel and I have no idea what I'm doing also how do Rust strings work. halp.
we can help!
#rust

basically writing
a toy kernel is
REALLY FUN

from: _@apple.com
to: julia@jvns.ca
subject: Opportunity with Apple

Hi,

I manage the Kernel Performance team at Apple...



WHAT I LEARNED

- having more than once process is hard
- filesystems are hard
- operating systems are SO COMPLICATED

EXPERIMENT 2: DATABASE SURGERY

HOW DOES SQLITE WORK?

FUN.SQLITE

```
id | word
1 | greedy
2 | greediness
3 | greener
```

```
$ hexdump fun.sqlite
|.............{.n|
|.a.R.D.4.%......|
|................|
|...y.n._.N.>.,.$|
|................|
|..............F.|
|..EAcevedo.E...D|
|Accra's.D...CAcc|
|ra.C..#BAccentur|
|e's.B...AAccentu|
|re.A..!@Acapulco|
|'s.@...?Acapulco|
```

A DATABASE IS A *TREE*

```
static MemPage *btreePageFromDbPage(DbPage *pDbPage, Pgno pgno, BtSha
    // actual code
  printf("Read a btree page, page number %d\n", pgno); // added by me
   // actual code
}
```

```
sqlite> select * from fun where id = 1;
Read a btree page, page number 1
Read a btree page, page number 5
Read a btree page, page number 828
Read a btree page, page number 10
Read a btree page, page number 2
Read a btree page, page number 76
Read a btree page, page number 6
1|A's
```

```
sqlite> select * from fun where id = 20;
Read a btree page, page number 1
Read a btree page, page number 5
Read a btree page, page number 828
Read a btree page, page number 10
Read a btree page, page number 2
Read a btree page, page number 76
Read a btree page, page number 6
20|Aaliyah
```

```
sqlite> select * from fun where id = 80000;
Read a btree page, page number 1
Read a btree page, page number 5
Read a btree page, page number 1198
Read a btree page, page number 992
Read a btree page, page number 2
Read a btree page, page number 1813
Read a btree page, page number 449
80000|scarfs
```

WHAT I LEARNED

- databases tables are trees
- databases are made of pages
- i can read some of the SQLite source code!

EXPERIMENT 3: WRITE A TCP STACK

EXPERIMENT 3: WRITE A TCP STACK IN PYTHON

David Wolever and 1 other Retweeted

Gary Bernhardt @garybernhardt · 15 Jul 2015

Your **homework** is to implement UDP in Python using raw sockets (SOCK_RAW) and get another machine to talk to you. Hard mode: implement **TCP**.

18 88

```
ip_header = IP(dst=dest_ip, src=src_ip)
syn = TCP(dport=80, sport=59333,
          ack=0, flags="S")
# Send the SYN packet to Google
response = srp(ip_header + syn)
```

WHAT I LEARNED

- how TCP packets are put together!
- you can write a 10% working TCP from scratch in 2 weeks
- python can't keep up

EXPERIMENT 4: CONCURRENCY

```
int counter;

void *AddThings(void *threadid) {
   for (int i = 0; i < 10000; i++)
        counter += 1;
   pthread_exit(NULL);
}
```

WRONG ANSWER

MUTEX

```
pthread_mutex_lock(&mutex);
counter += 1;
```

'ATOM'

```
__sync_add_and_fetch(&counter, 1);
```

WHAT I LEARNED

- atoms are faster than mutexes

"can you discuss the pros and cons of using a lock-free approach for implementing a thread-safe hashmap?"

I BLOG MY EXPERIMENTS

Diving into HDFS

in strace

Yesterday I wanted to start learning about how HDFS (the Hadoop Distributed File System) works internally. I knew that

- It's distributed, so one file may be stored across many different machines
- There's a *namenode*, which keeps track of where all the files are stored
- There are *data nodes*, which contain the actual file data

But I wasn't quite sure how to get started! I knew how to navigate the filesystem from the command line (`hadoop fs -ls /`, and friends), but not how to figure out how it works internally.

DO ENOUGH EXPERIMENTS

END UP WITH ACTUAL KNOWLEDGE