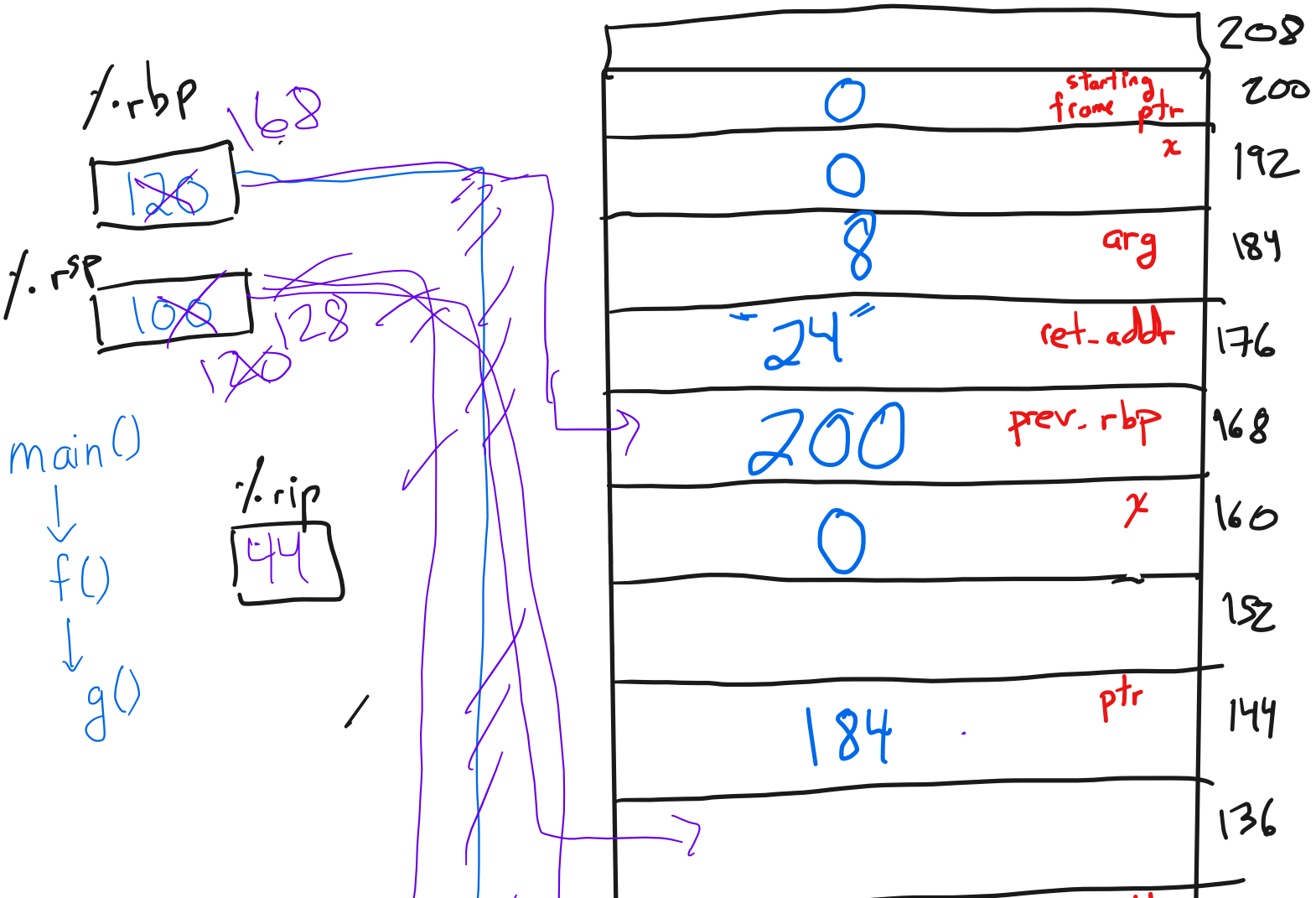
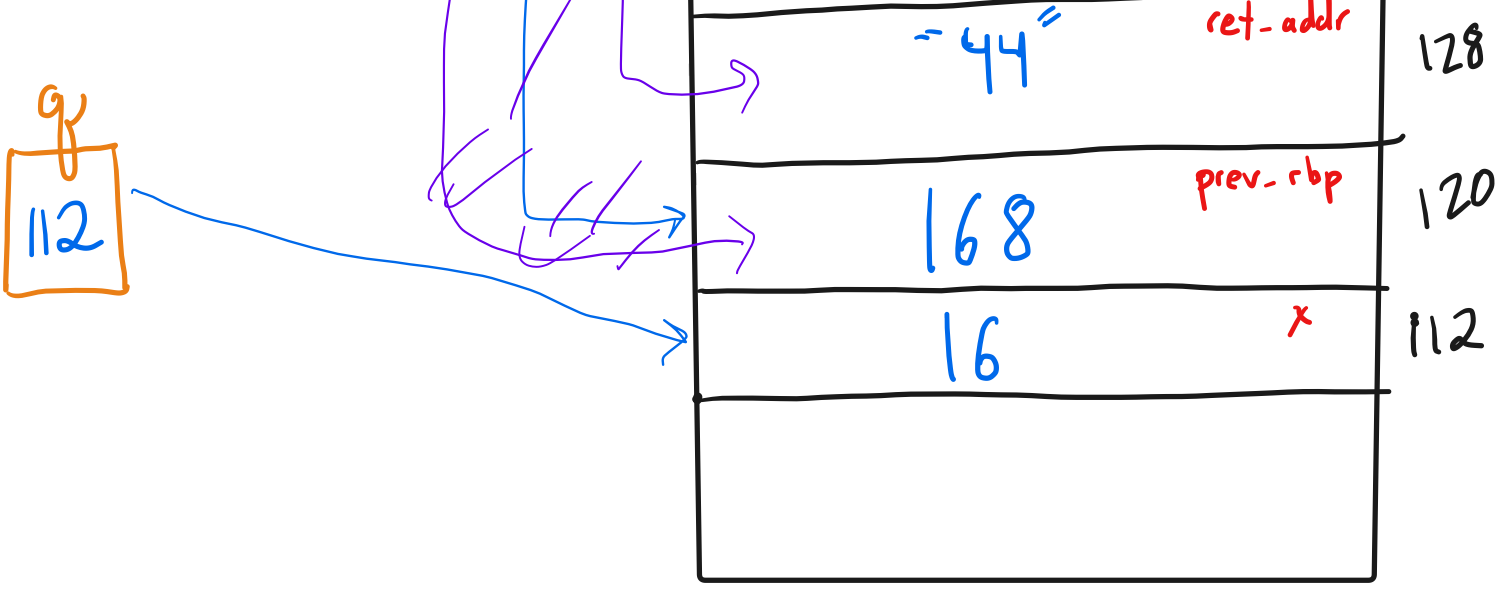
- [ ] 1. Last time
- [ ] 2. Stack frames, continued
- [ ] 3. System calls
- [ ] 4. Process/OS control transfers
- [ ] 5. Git/lab setup
- [ ] 6. Process birth
- [ ] 7. Shell, part I
- [ ] 8. File descriptors
- [ ] 9. Shell, part II

} next time





---

3. System calls

Examples:

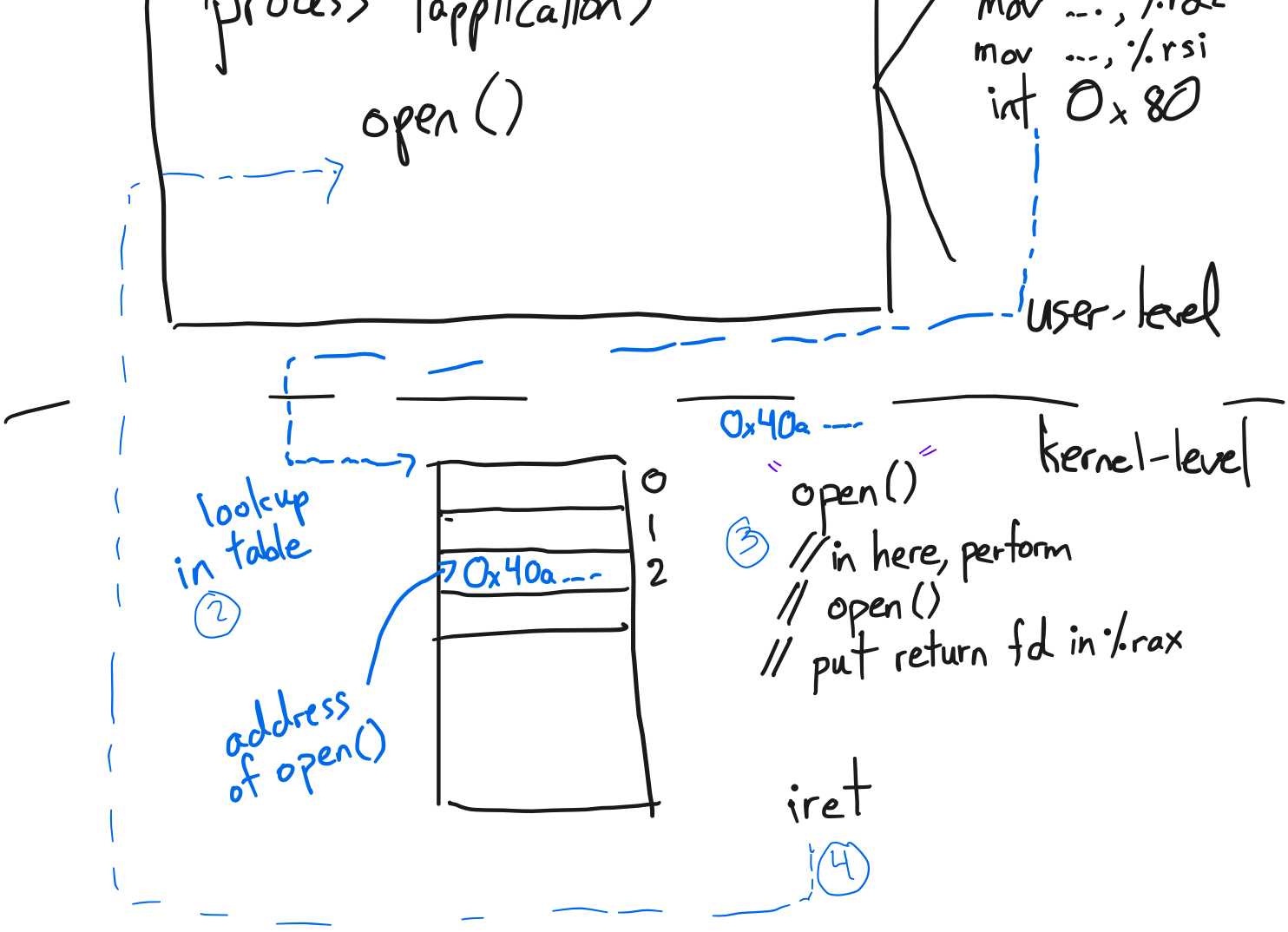
```
int fd = open(const char* path, int flags);
int rc = write(int fd, const void*, size_t s);
int rc = read(int fd, void*, size_t s);
```

```
int fd;
fd = open("/tmp/foo", O_RDWR | O_CREATE);
write(fd, "abc... z", 26);
```

$ man 2 open
$ man 2 readdir

---

4. Process/OS control transfers



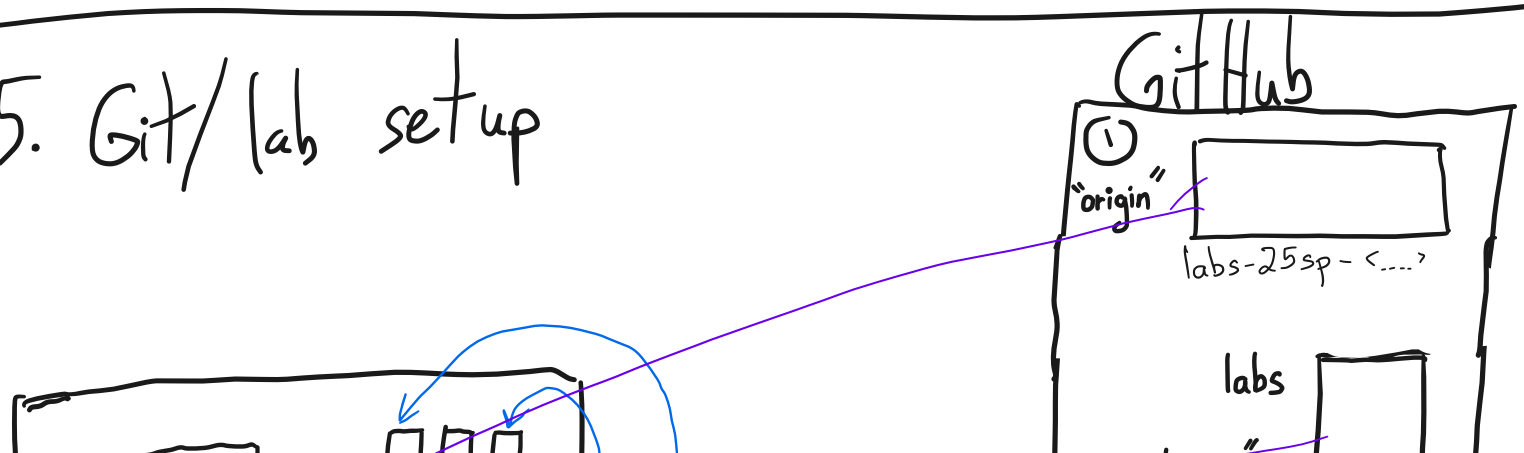
A. System calls

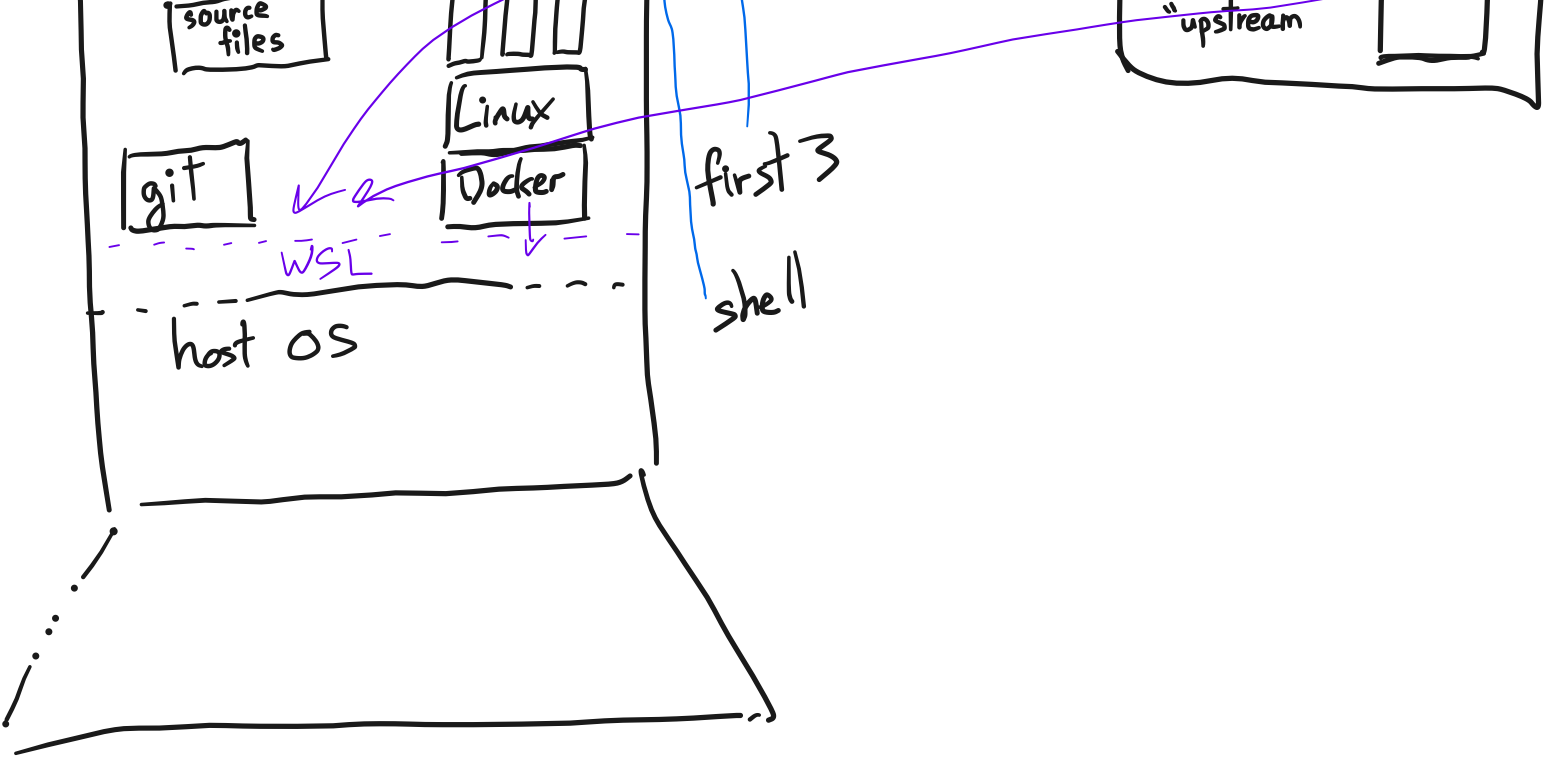
B. Interrupts

C. Exceptions

---

5. Git/lab setup





---

⑥ Process birth

```
fork();

switch ( fork() ):

    case 0:

        —
        —
        —

    default:
```

default

```
wait();
```

```
for (i=0; i < 10; i++ ) {
    fork();
}
```

```
while (1) { }
```

---

(7) The shell, part I

- program that creates processes
- the human's interface to the computer
- GUIs in OSes are another kind of shell

core loop in a text shell:

```
while (1) {
    write(1, "$ ", 2);
```

```
    readcommand (command, args);  // parse input
    if ((pid = fork()) == 0)      // child?
        execve (command, args, 0);
    else if (pid > 0)  // parent?
        wait (0);  // wait for child
    else
        perror ("failed to fork()");
}
```