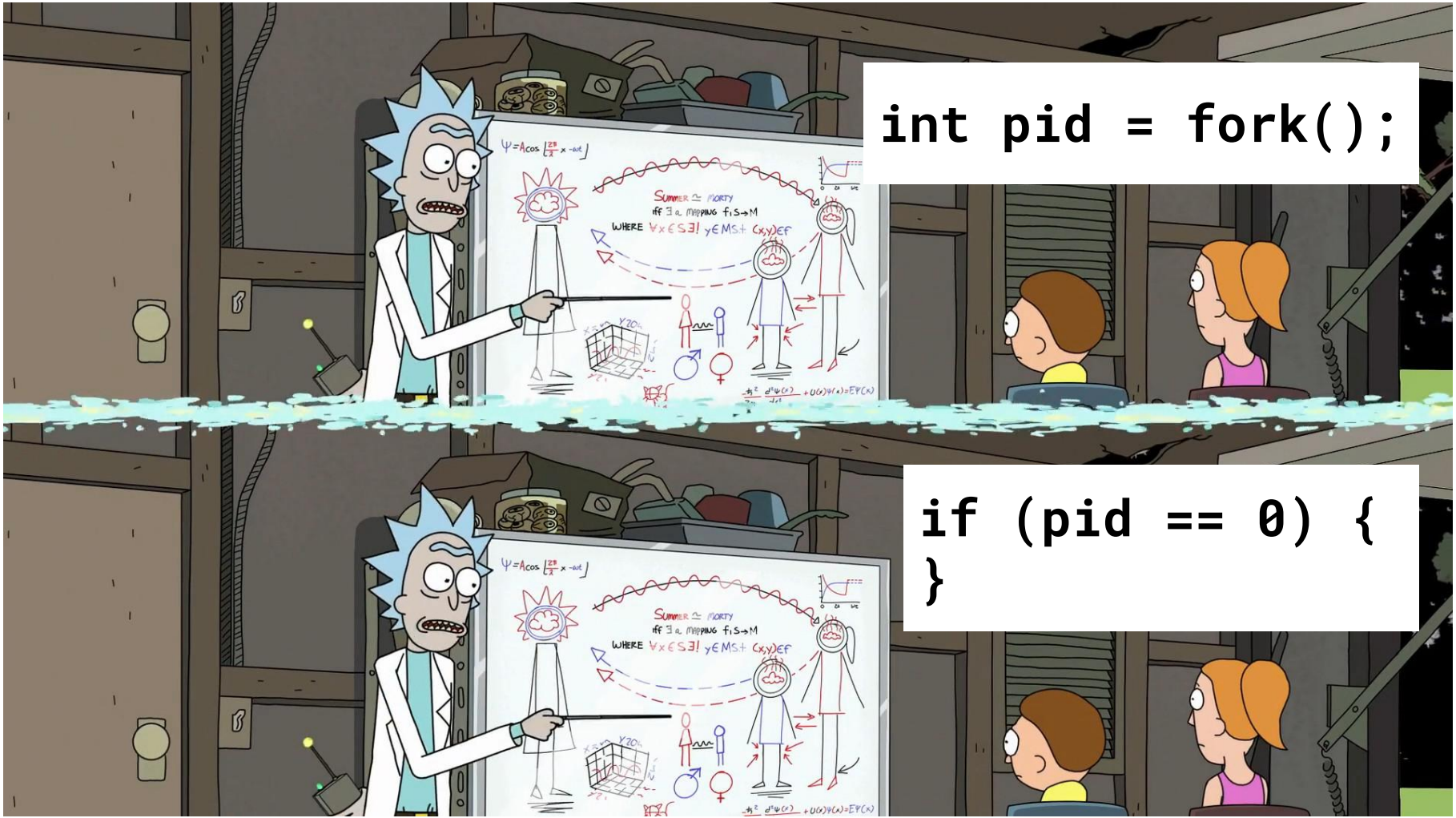
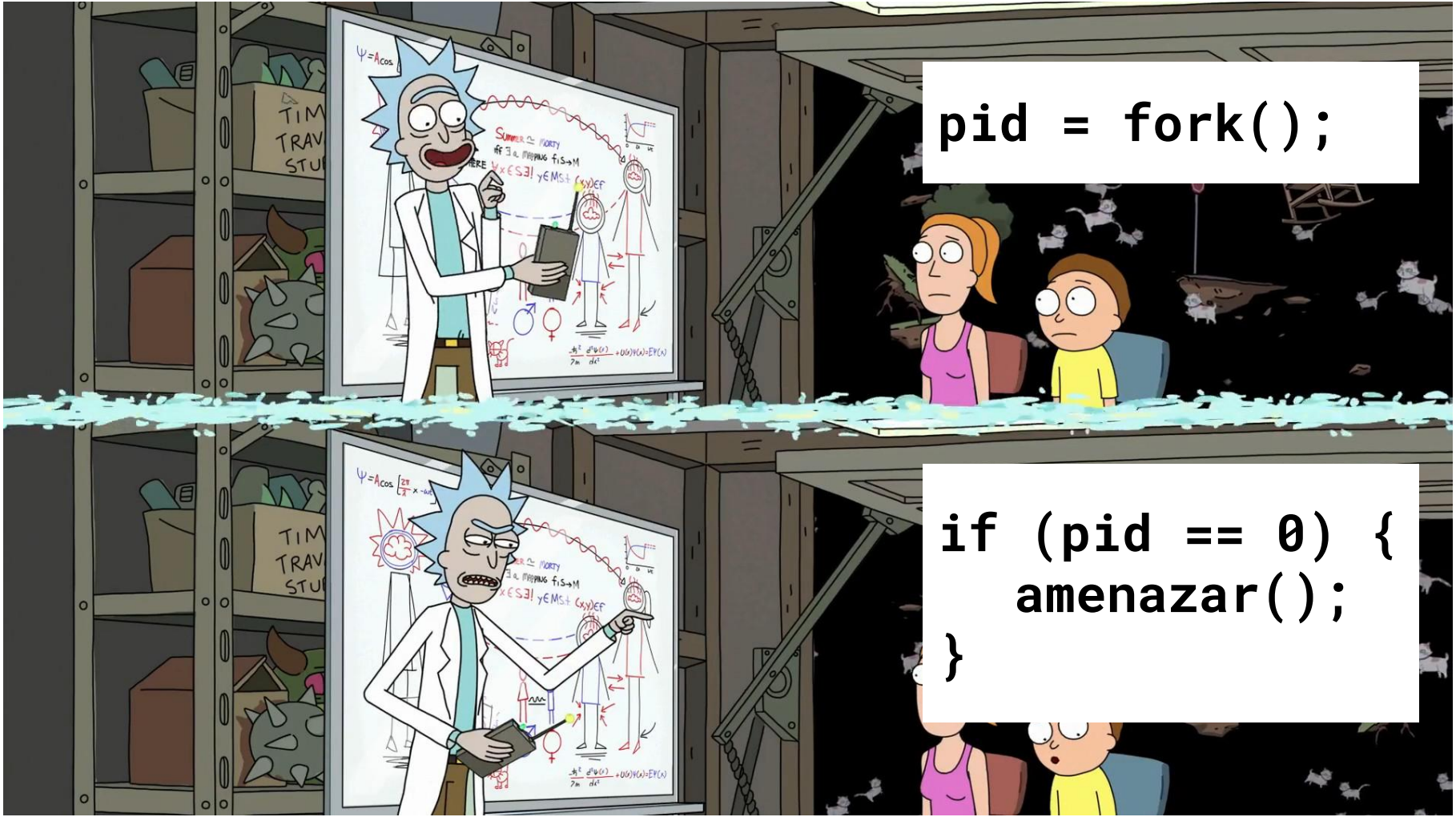
System calls

fork()

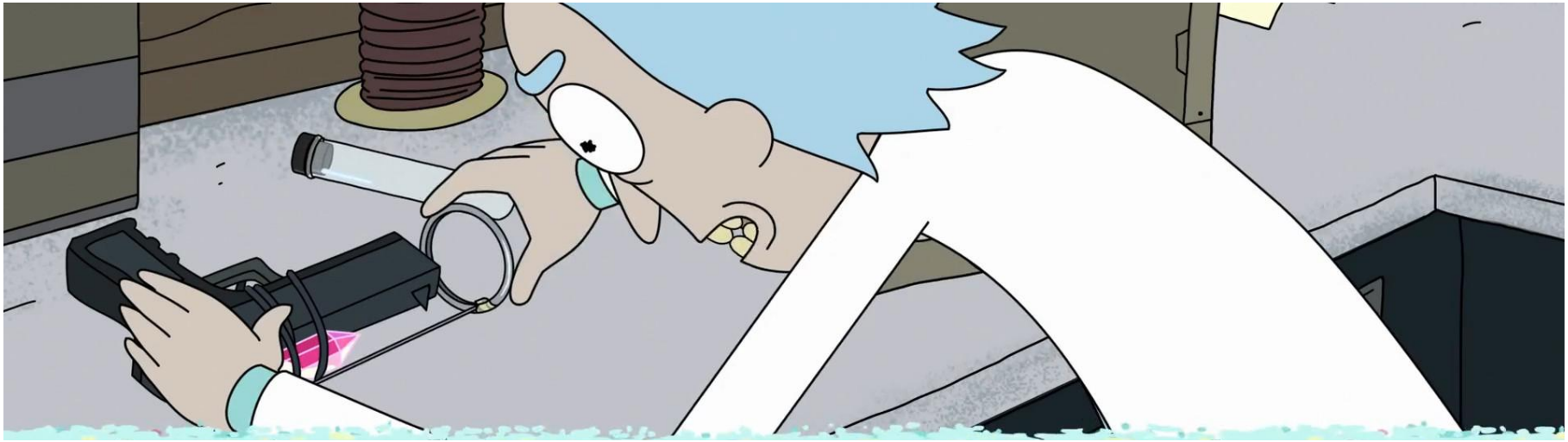
```
pid_t fork(void);
```


int pid = fork();
if (pid == 0) {
}

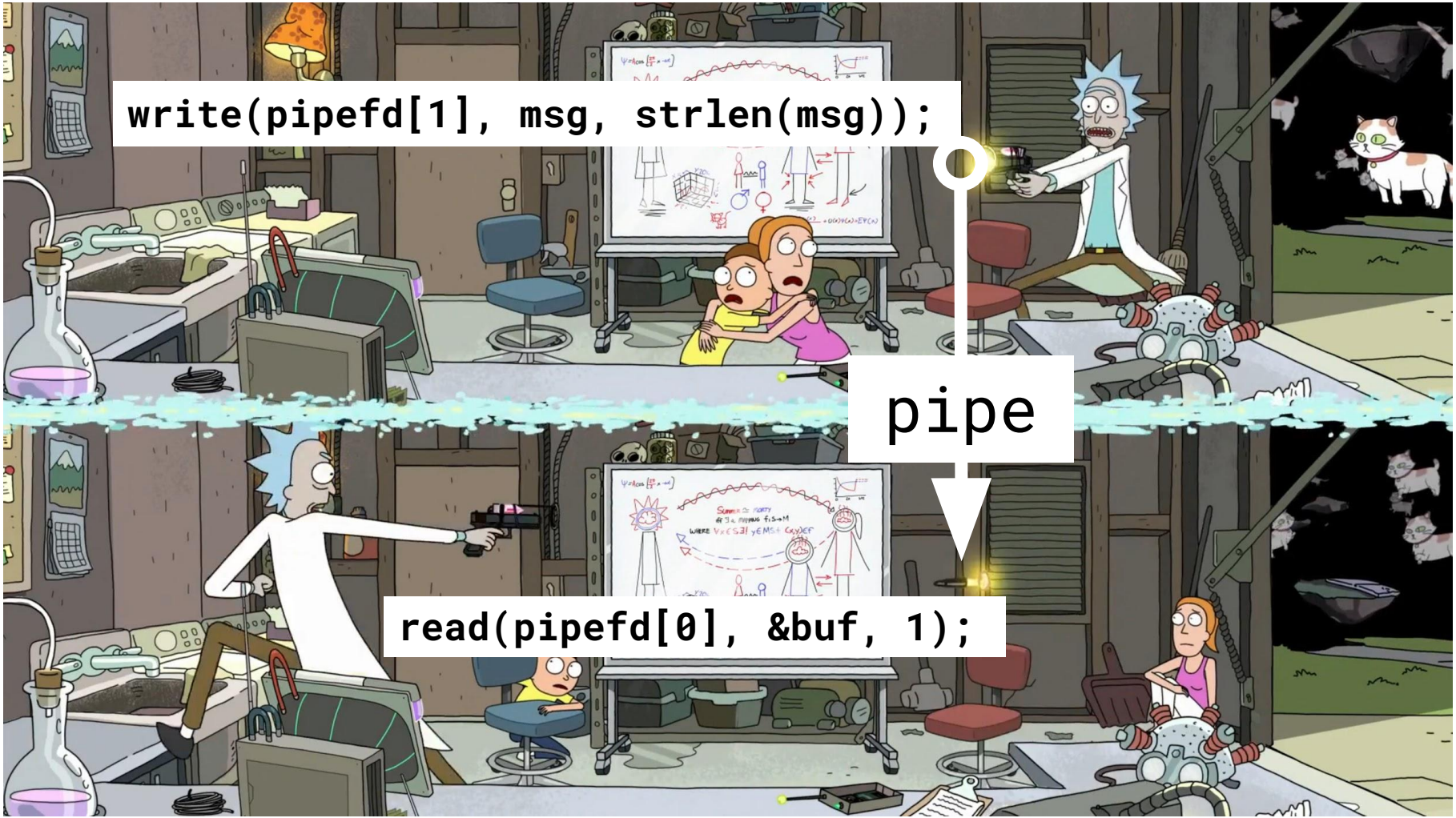

pid = fork();
if (pid == 0) {
amenazar();
}

pipe()

```
int pipe(int pipefd[2]);
```



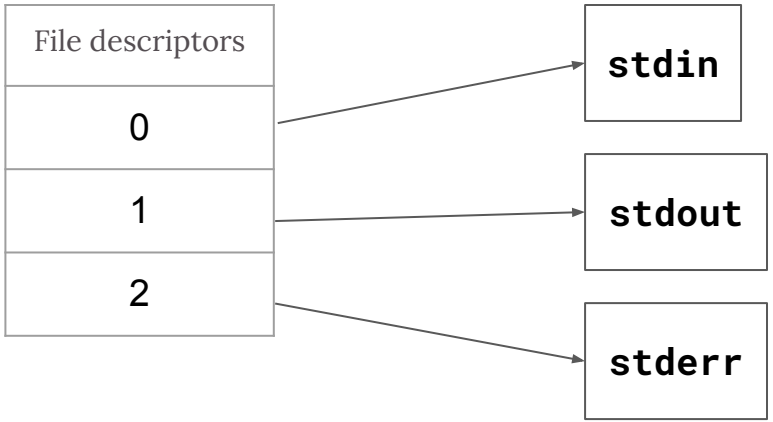
```
int pipefd[2];
if (pipe(pipefd) == -1) {
   exit(EXIT_FAILURE);
}
```


write(pipefd[1], msg, strlen(msg));
pipe
read(pipefd[0], &buf, 1);

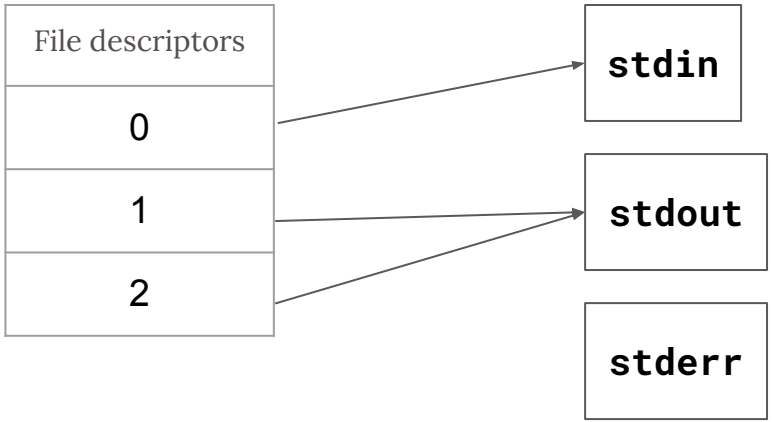
dup()

```
int dup2(int oldfd, int newfd);
```

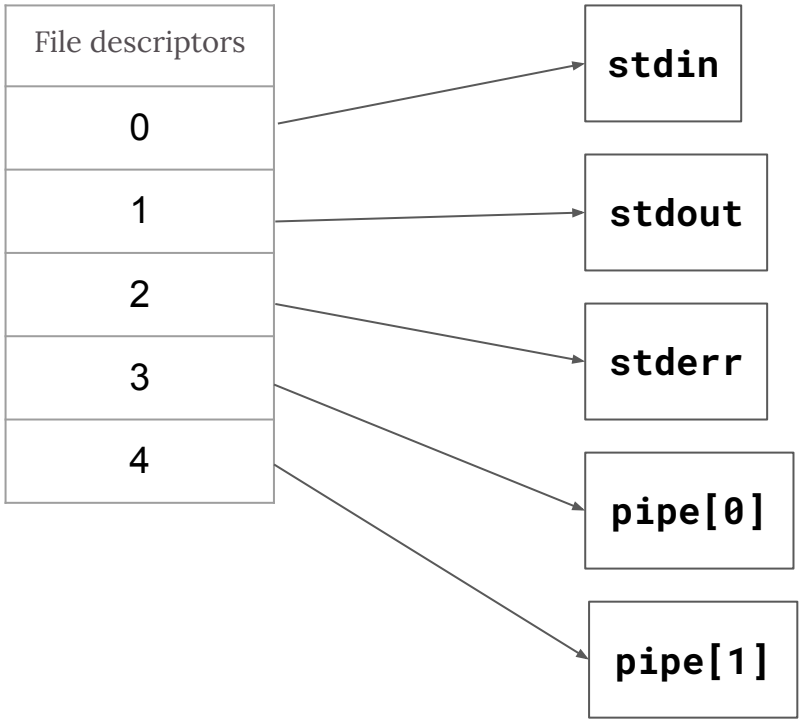
Default



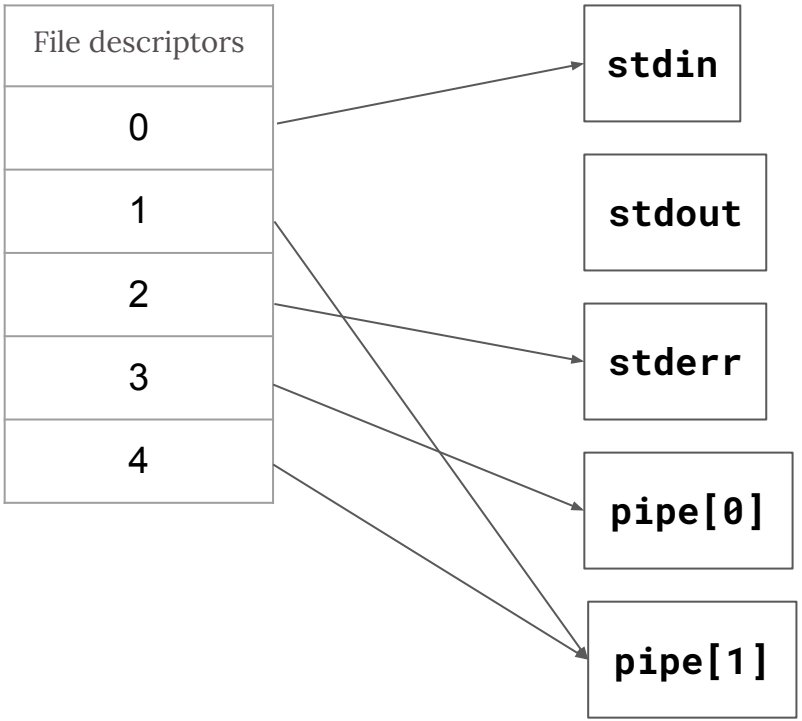
dup2(1,2)



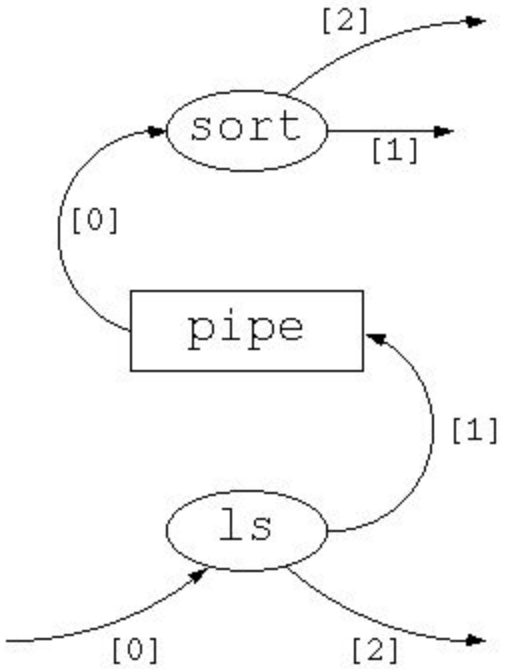
Child


File descriptors
0
1
2
3
4
stdin
stdout
stderr
pipe[0]
pipe[1]

dup(4,1);


File descriptors
0
1
2
3
4
stdin
stdout
stderr
pipe[0]
pipe[1]

$ ls | sort



sort

file descriptor table

[0]	pipe *read*
[1]	*standard output*
[2]	*standard error*

ls

file descriptor table

[0]	*standard input*
[1]	pipe *write*
[2]	*standard error*

execv()

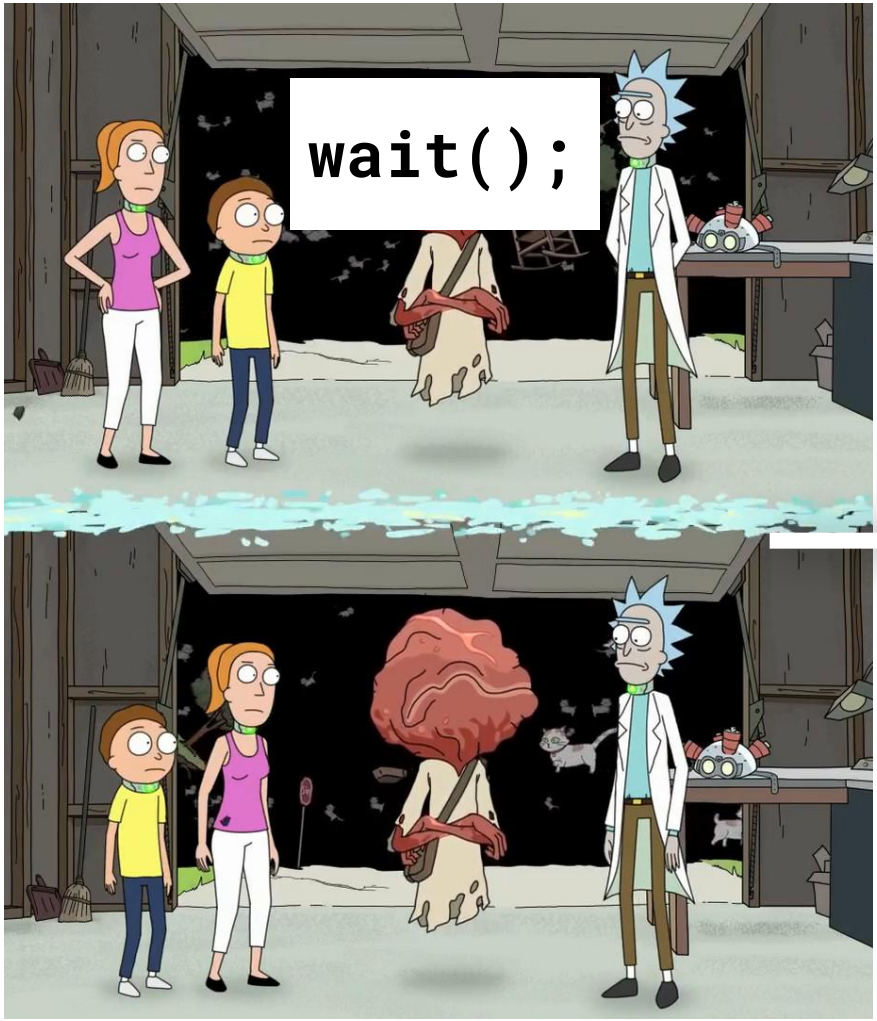
```
int execvp(const char *file, char *const argv[]);
```


parent
execv(“chernobyl”, &args);

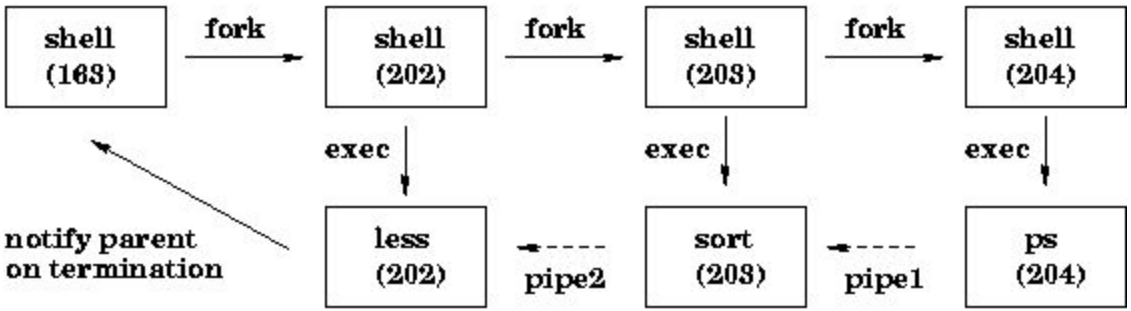

parent
MOSCOW
APRIL 26, 1988

wait()

```
pid_t wait(int *wstatus);
```


wait();

ps | sort | less


shell
(163)
fork
shell
(202)
fork
shell
(203)
fork
shell
(204)
exec
exec
exec
notify parent
on termination
less
(202)
pipe2
sort
(203)
pipe1
ps
(204)

Recomendación:
¡lean las man pages!