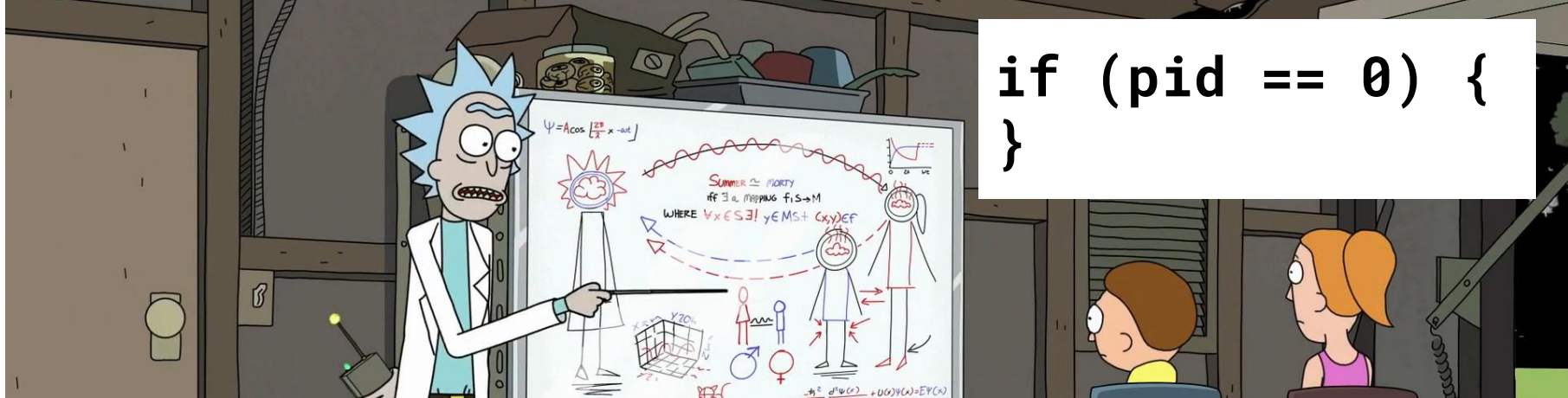
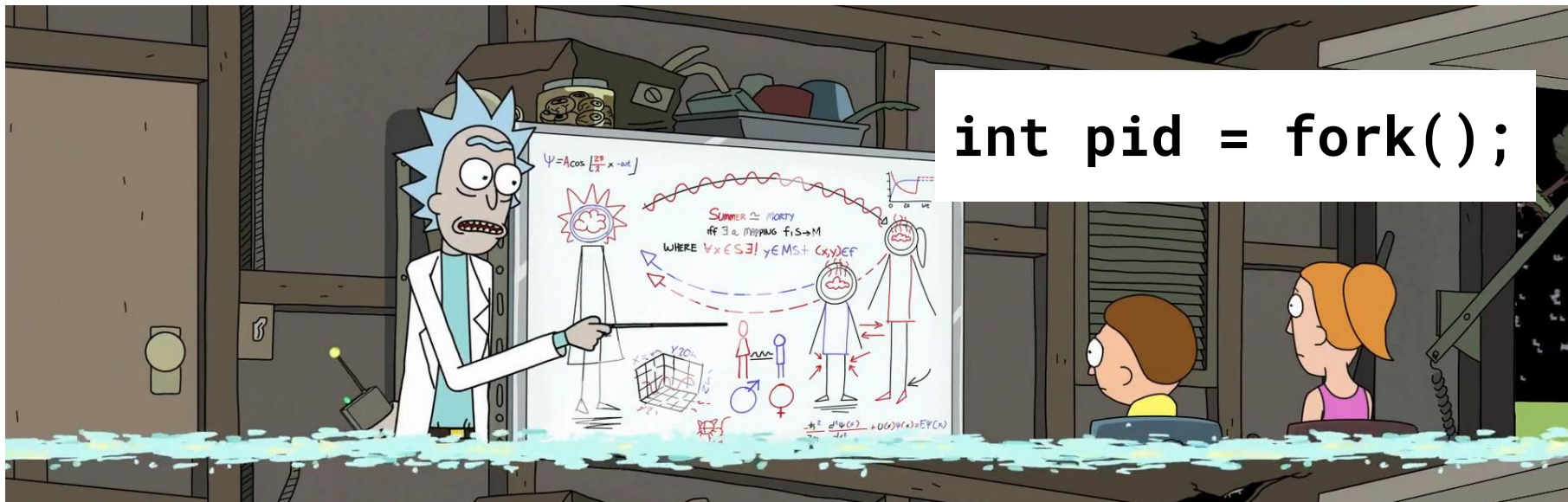


System calls

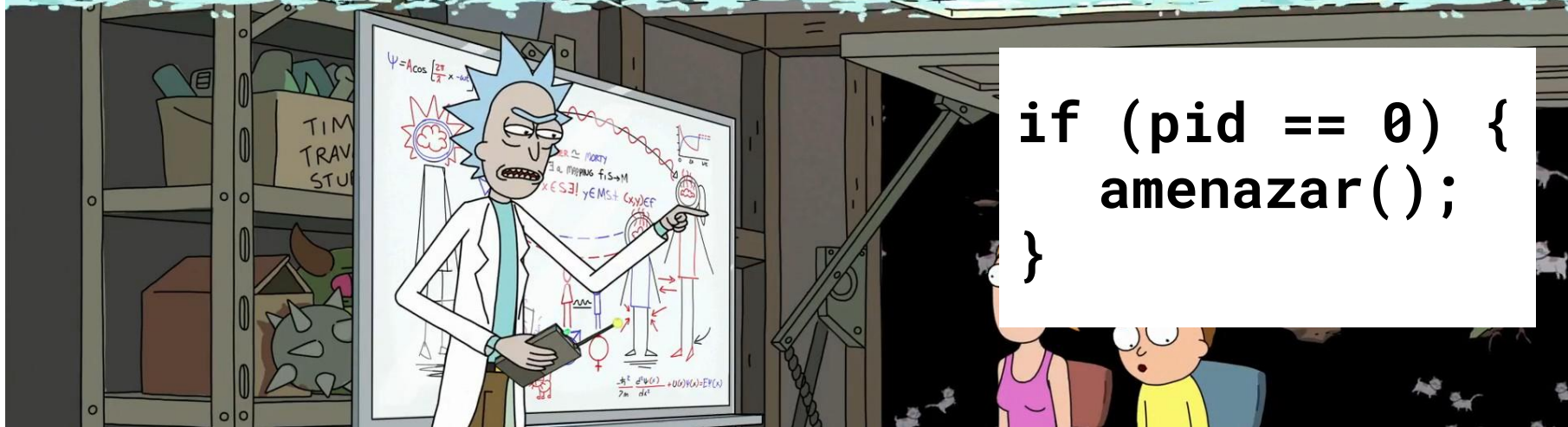
fork()

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





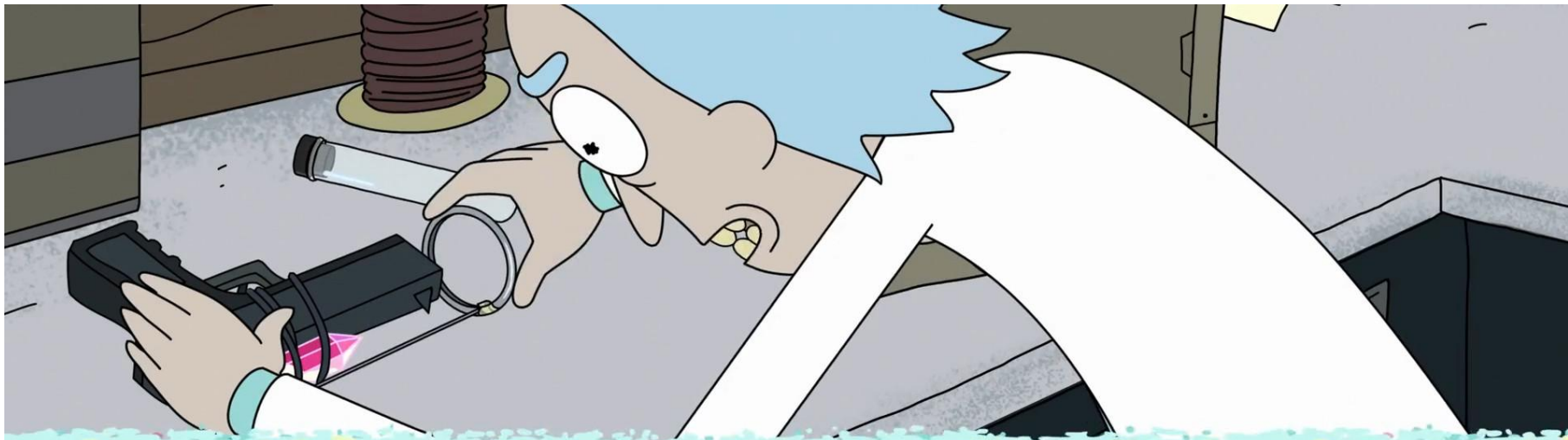
```
pid = fork();
```



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

pipe()

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



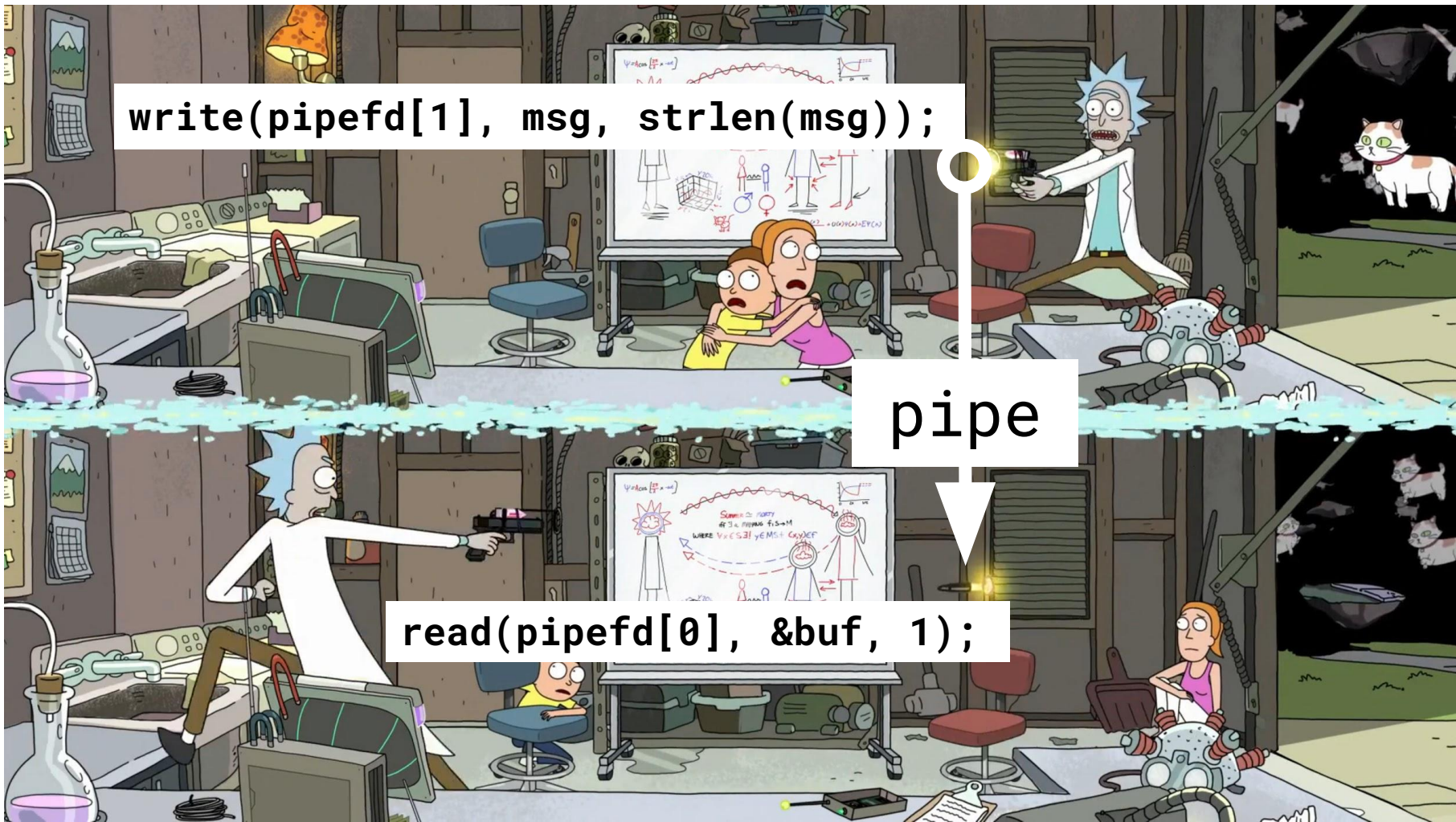
```
int pipefd[2];  
if (pipe(pipefd) == -1) {  
    exit(EXIT_FAILURE);  
}
```



```
write(pipefd[1], msg, strlen(msg));
```

pipe

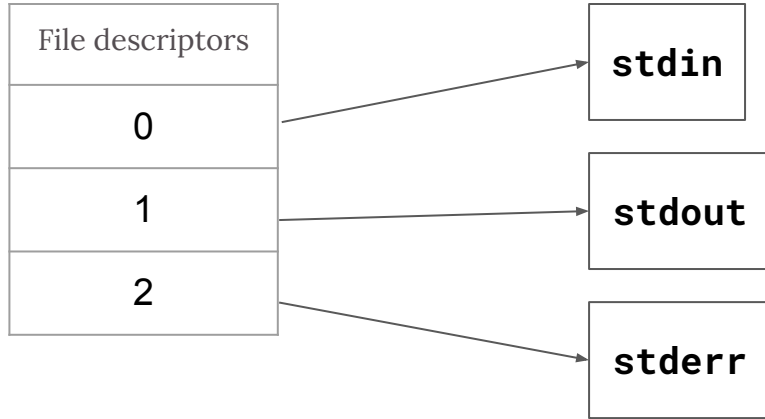
```
read(pipefd[0], &buf, 1);
```



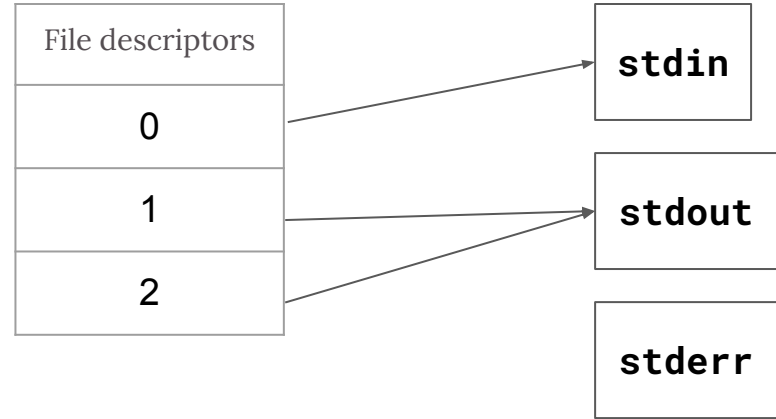
dup ()

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

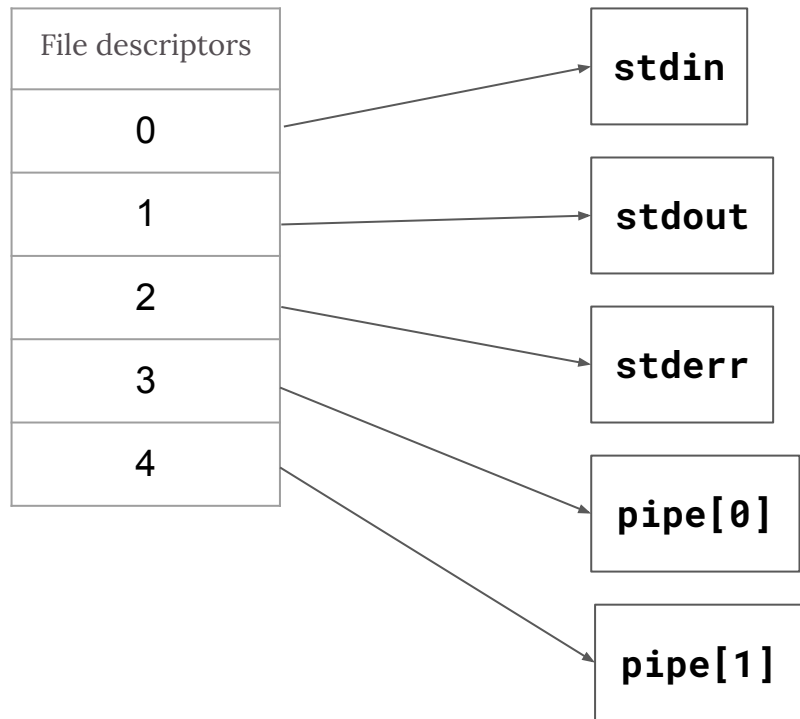

Default



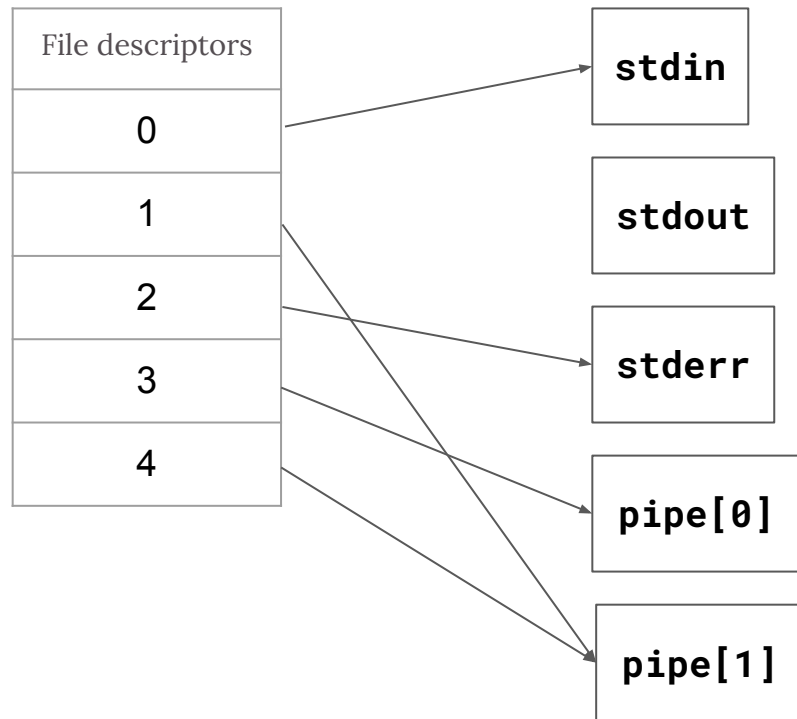
`dup(2, 1);`



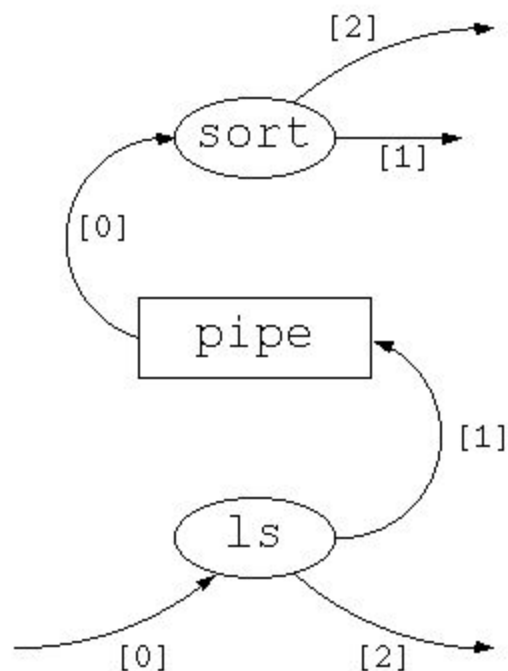
Child



`dup(1, 4);`



\$ ls | sort



sort

file descriptor table

[0]	pipe <i>read</i>
[1]	<i>standard output</i>
[2]	<i>standard error</i>

ls

file descriptor table

[0]	<i>standard input</i>
[1]	pipe <i>write</i>
[2]	<i>standard error</i>

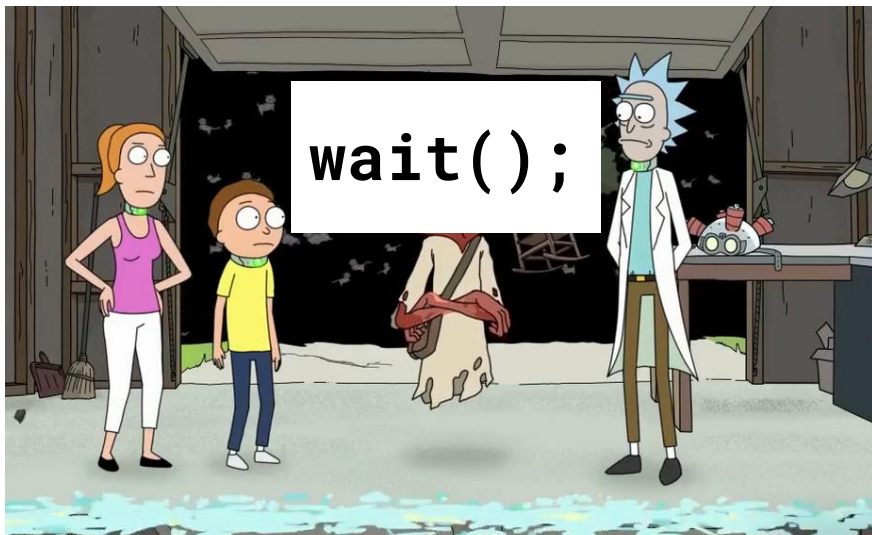
execv()

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

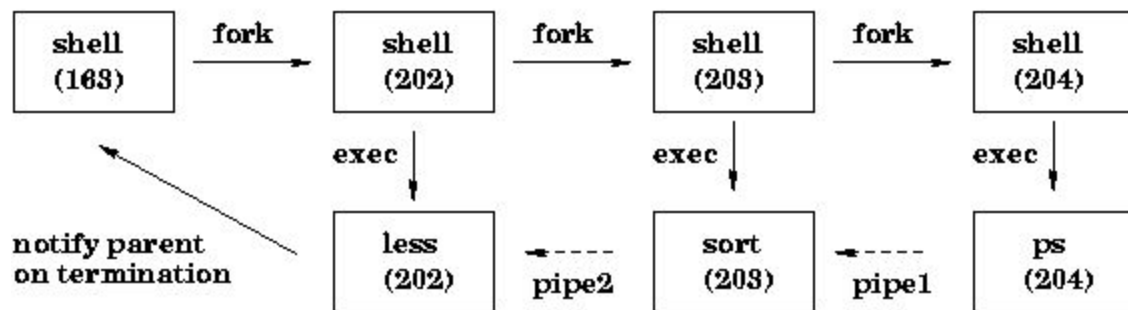


wait()

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

ps | sort | less



Recomendación:
¡lean las man pages!