Lecture 8: Let's build a shell

CSE 29: Systems Programming and Software Tools

Olivia Weng

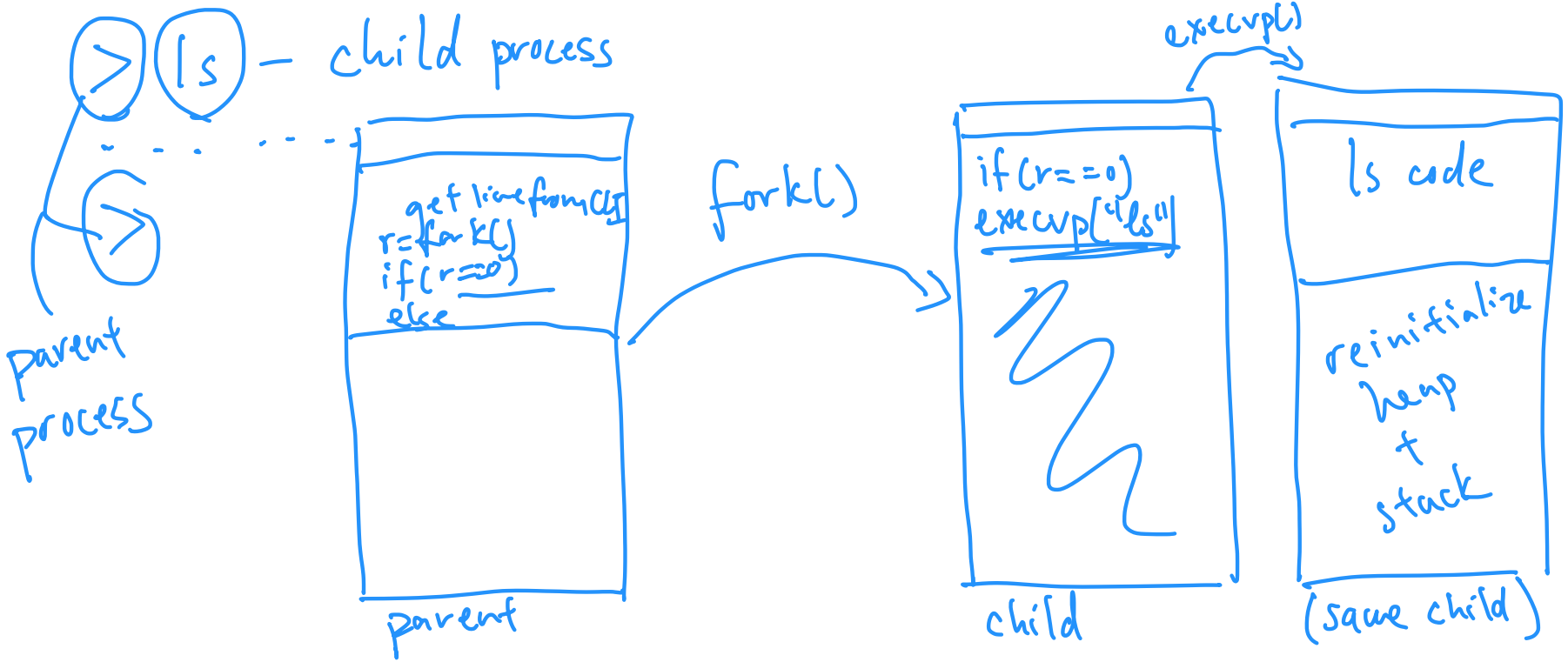
Announcements

- Problem set 2 released and due Wednesday at 10am PT

- Problem set 1 makeup/practice available on PrairieLearn

The shell creates, manages, and runs processes

- Loads new code into a process with `execvp()` to run shell commands



Load new code in a process

- `execvp();` ← syscall
  - Loads new code from a binary file (filename) into the running process
  - If it runs without error, it never returns

Demo

- Let's make a shell!

How can we parse the command line args?

- `fgets();`

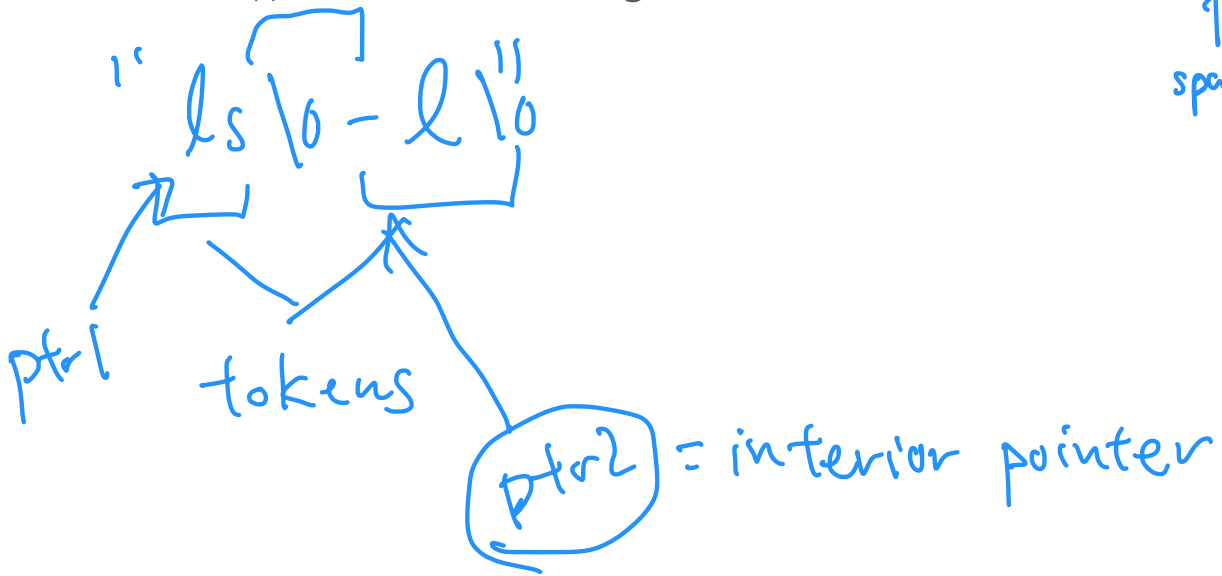
- `sscanf();`

Waiting for a process to finish

- `int wait(pid_t pid);` — syscall
  - Wait for child process with `pid` to end
  - This is called only by the parent
  - `pid = NULL` will wait for all children to finish

How can we parse the command line args?

- `strtok()` tokenizes a string



What is file I/O?

- I/O = input/output
- What can we do with files?
  - `fopen()`: Open a file
  - `fclose()`: Close a file
  - `fgets()`: Read a string from a file
  - `fprintf()`:Write to a file
  - and more!

- We take input from STDIN and output to STDOUT…
  - In Unix, everything is a file!
  - `STDIN` and `STDOUT` are files