Agenda for Today

1. Assignment Q&A
2. Stack Implementation
3. Processes
4. Limited Direct Execution
5. Wrap Up
Assignment Q&A

When will office hours be scheduled?

I will only be available on Monday, but I'm not sure when yet. I will announce soon. Mentors may be available sooner.
C Practice:
Stack Implementation
Writing a Stack in C

Discuss the following questions with your neighbor(s).

What are the parameters and return types for each operation?

How will we store the stack's contents? Think about data types we will need.
Writing a Stack in C  *(responses)*

**What are the parameters and return types for each operation?**

- `int stack_pop(int_stack_t* stack)`
- `void stack_push(int_stack_t* stack, int value)`
- `int stack_peek(int_stack_t* stack)`
- `void stack_init(int_stack_t* stack)`
- `void stack_destroy(int_stack_t* stack)`

**How will we store the stack's contents? Think about data types we will need.**

- An array: more efficient use of space
- A linked list: easy/efficient to append
Live Coding
Processes
Processes: Basics

What is a process?
A running program. Multiple processes can run at the same moment. There is a parent-child relationship between processes. The process holds the machine instructions that run the program, its current state, and an address space where the process runs.

Why does the OS provide this abstraction?
A process is a self-contained representation of a running program that we can pause and resume for time sharing/multitasking. Running in processes isolates them from each other. Good for security/privacy and reliability.
What does this function do?

Fork produces a child process that is a copy of the calling process. The process that called fork() is the parent of the new child process.

In the child process, fork() returns 0. In the parent, fork() returns the ID of the child.
What does this function do?

Calling `wait` in the parent process will wait for any child process to exit. The `wait` function returns the ID of the child that exited.

The caller of `wait()` should pass in a pointer to an integer where values will be written by `wait`. Those values tell us the exit code from the child, how the process exited, and a few other details.
int execv(const char* pathname,
           char* const argv[])
Demo: fork, wait, and exec
Wrap Up
Reminders

Reading
Read about gdb to prepare for Friday's in-class exercise.

Assignment
The ngram assignment is due Monday at 11:59pm. Start early!