Agenda for Today

1. Assignment Q&A
2. Processes
3. Limited Direct Execution (on Monday)
4. Wrap Up
Assignment Q&A

Can we submit to gradescope multiple times? Yes. As many times as you want up to the deadline. There is no penalty for multiple submissions.

Is it true that we cannot read the entire input at once? Yes. It could be infinitely long, but your program should still produce the correct output up until the moment we stop waiting for it.

If the input is "hello" and I read one character with fgetc, what will it read the next time I call fgetc? The first read will return 'h', and the second will return 'e'.
Processes
Processes: Basics

What is a process?
A running program uses a process. An executable file describes the operations a program should perform when it runs, but it isn't running.
The process can be in one of a few different states (determined by the OS)
The process captures the position in the program, local variable values, other values stored in memory, and more.

Why does the OS provide this abstraction?
A process is a useful container for all the things we need for a running program.
The OS can switch between processes silently, and each can pretend it has its own processor that it runs on.
A process is isolated from other processes (good for security/privacy & reliability)
pid_t fork()

What does this function do?

Calling fork() creates a new process. The calling process is the parent, and the new process is the child.

The child process is an almost-identical copy of the parent. The child will have a different process ID.

The fork() function appears to return twice: once in the parent (returns the child process ID), and once in the child (returns zero). This allows the programmer to specify what the parent and child should do separately.
pid_t wait(int* status)

What does this function do?
A parent process can call wait() to wait for one of its child processes to exit. The return value is the ID of the child that exited (or -1 if there was an error).
The status parameter is an output. We pass pointer, and wait will write through that pointer to give us some information.
The value written to status tells us how the child process died (exited or terminated via signal) and some other information.
Typical usage:
int status;
pid_t result = wait(&status);
What does this function do?

Starts a program at "pathname" in this process. The state of the currently-running program is completely replaced by the new program, which starts at main(). If execv fails, it returns -1. If it succeeds, there is no return.
Demo: fork, wait, and exec
Wrap Up
Reminders

Reading
Read about Address Spaces and Virtual Memory for Monday.

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

Need help?
Stop by mentor office hours from 2–5pm on Sunday (in this room)
Book a time during my Monday office hours (10:30am–noon)