Project #1:
Week 2

Principles of Operating Systems (LAB)

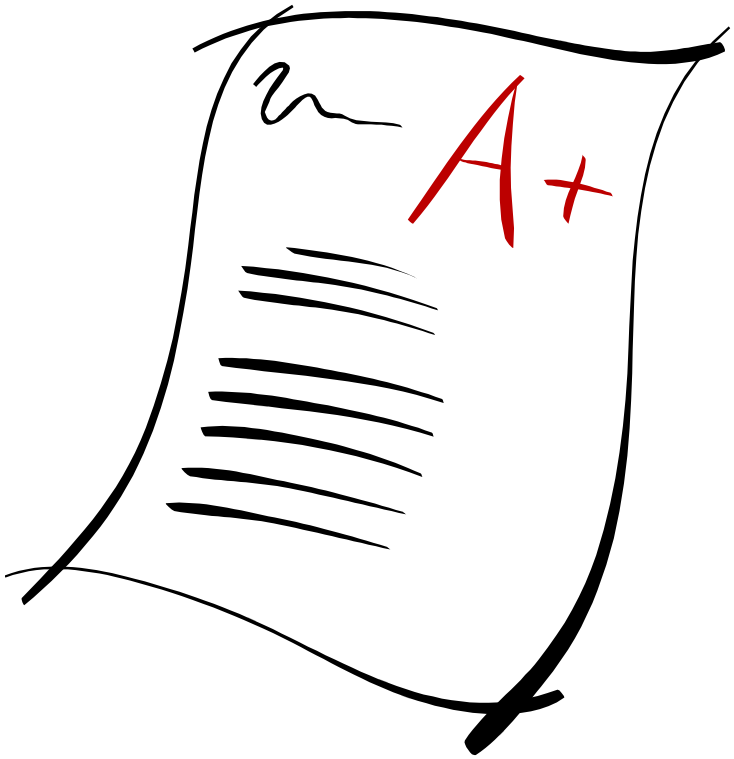
COP4610/CGS 5765

Overview

- Lab Information
- Teams
- How to get started
- Environment Variables
- Searching for a command using **`$PATH`**
- Starting Programs
- Questions

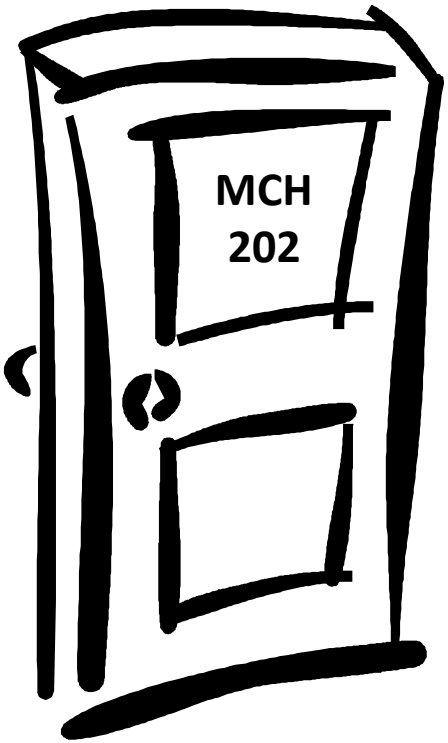
Project Weights

- Project #1: **30%**
- Project #2: **40%**
- Project #3: **30%**



Lab

- You should have access to the lab outside of recitation
  - Exceptions include other class meeting times as listed on course website
  - If you cannot access the lab (i.e., your card/pin does not unlock the door), let me know
- Make sure the lab door is closed if you are the last one to leave!



Teams for Project #1

- Everyone should have a partner
- If you have not done so, email me your team (i.e., the two members)

How Should I Get Started?

- Understand the requirements
- Design
  - Break the project into smaller pieces
  - Flow of program
    - Event-driven (what are the events?)
    - Time-driven (what are the time instants?)
  - Pseudo code
    - Repeating sequences (loops)
  - State diagram
- Discuss with team member

How Should I Get Started?

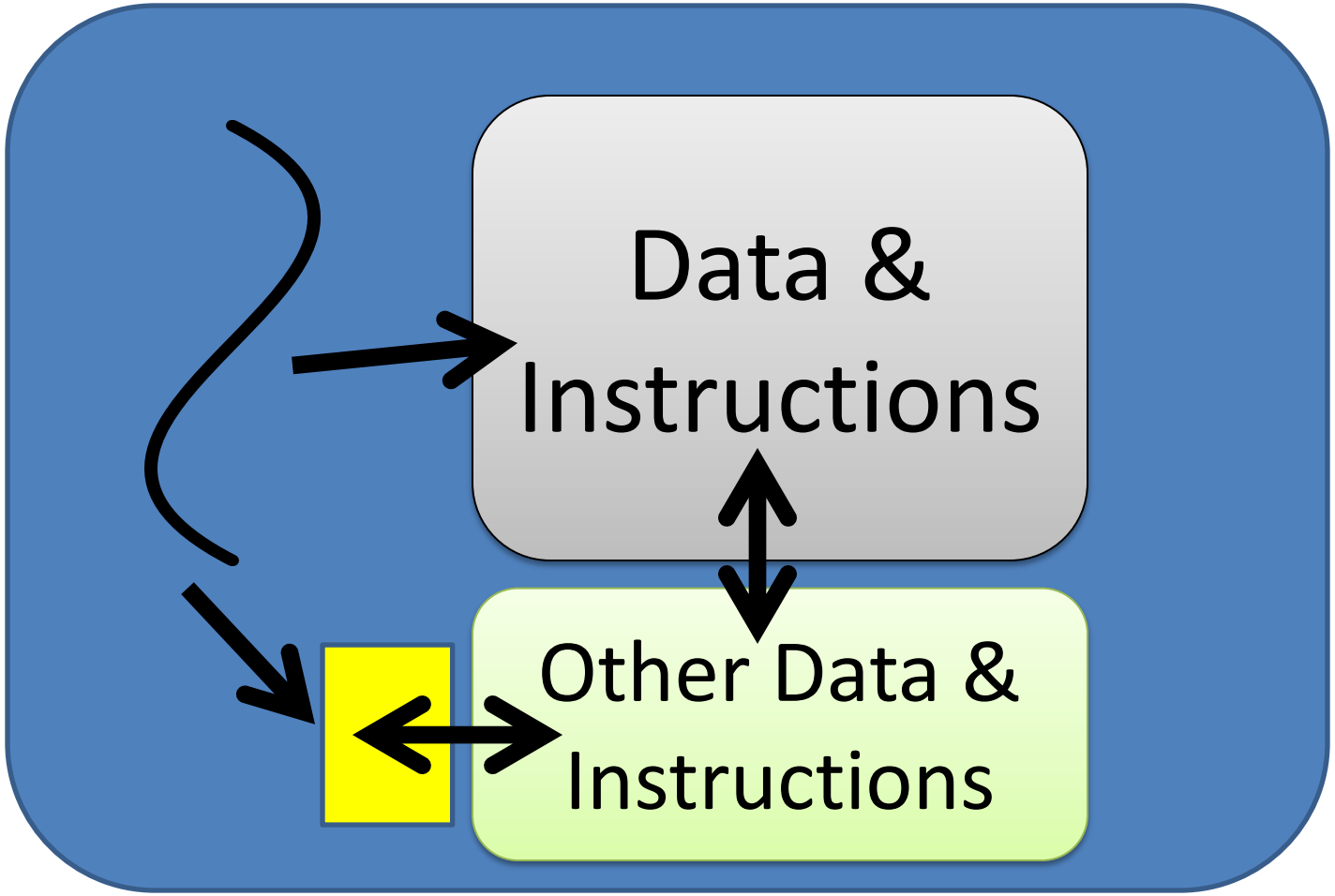
- Coding

```
int main()
{
   <small piece of project>
   return 0;
}
```

- see
  http://www.cs.fsu.edu/~cop4610t/assignments/project1/code_snippets/waitpid/main.c

ENVIRONMENT VARIABLES

Process



Environment Variables

- Mechanism to allow a program's operation to change based on a given value
  - E.g.,
    - $PATH
    - $HOME
    - $CC

Environment Variables

- SYNOPSIS

```
#include <stdlib.h>
char *getenv(const char *name);
```

- Returns
  - pointer to the value of the ***name*** environmental variable, if it exists
  - null pointer if ***name*** does not exist

Environment Variables (Example)

```
#include <stdlib.h>
...
const char *name = "HOME";
char *value;

value = getenv(name);
if(value) {
    ...
```

SEARCHING FOR AN EXECUTABLE

$PATH

- Allows one to easily run executables
- Colon separated list of directories
- Used by shell to find command, unless the command:
  - is a built-in command (e.g., **`cd`**)
  - has a slash (e.g., **`./a.out`**)
- E.g.,

```
<prompt> ls
```

Example

```
PATH=/usr/local/bin:/usr/bin
<prompt> ls
```

- Potential command locations

  /usr/local/bin/ls

  /usr/bin/ls

WAITPID()

`waitpid()`

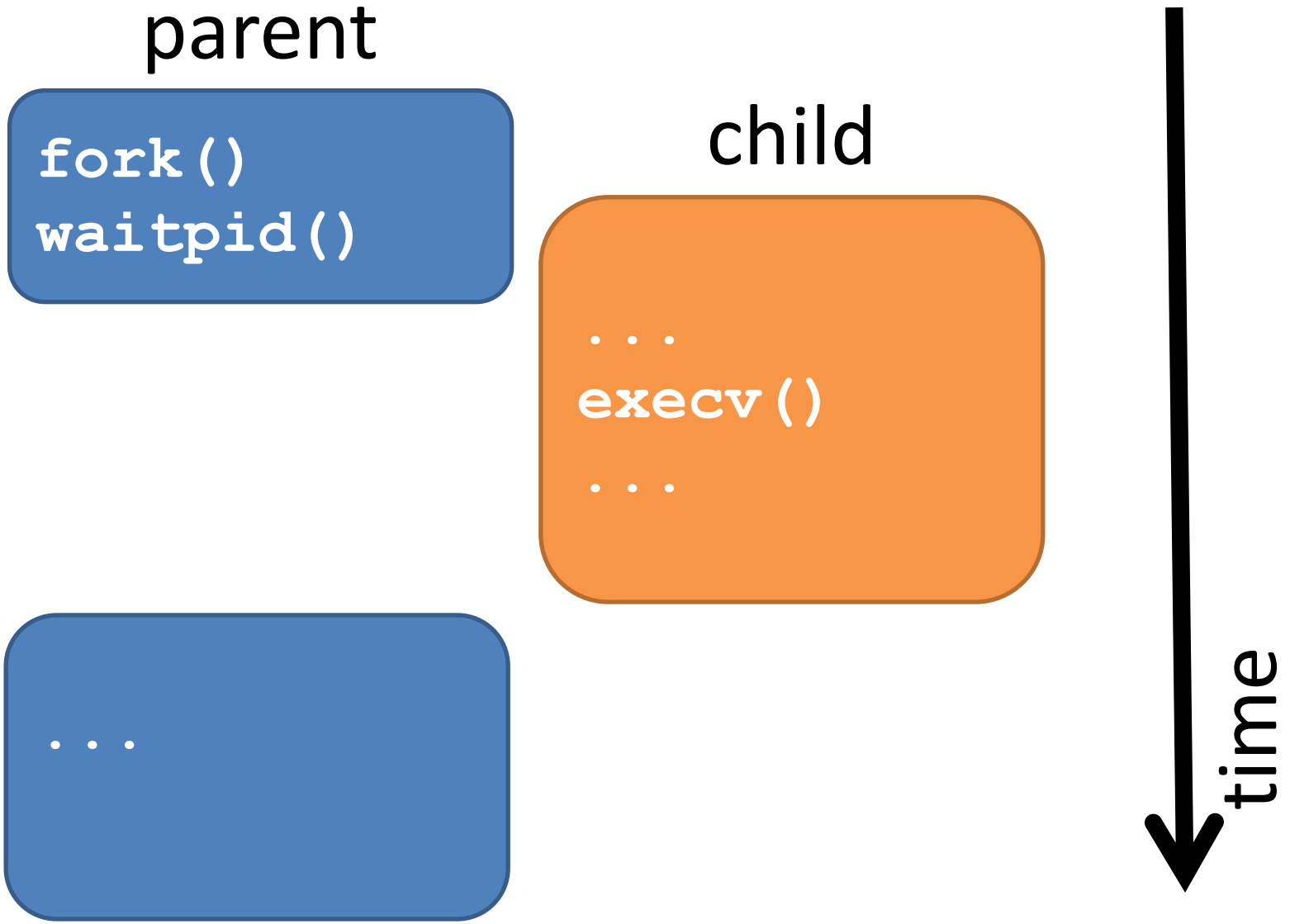
- Mechanism to operate based on status of child process(es)
  - Child exited?
  - Child still running?

`waitpid()` common usage

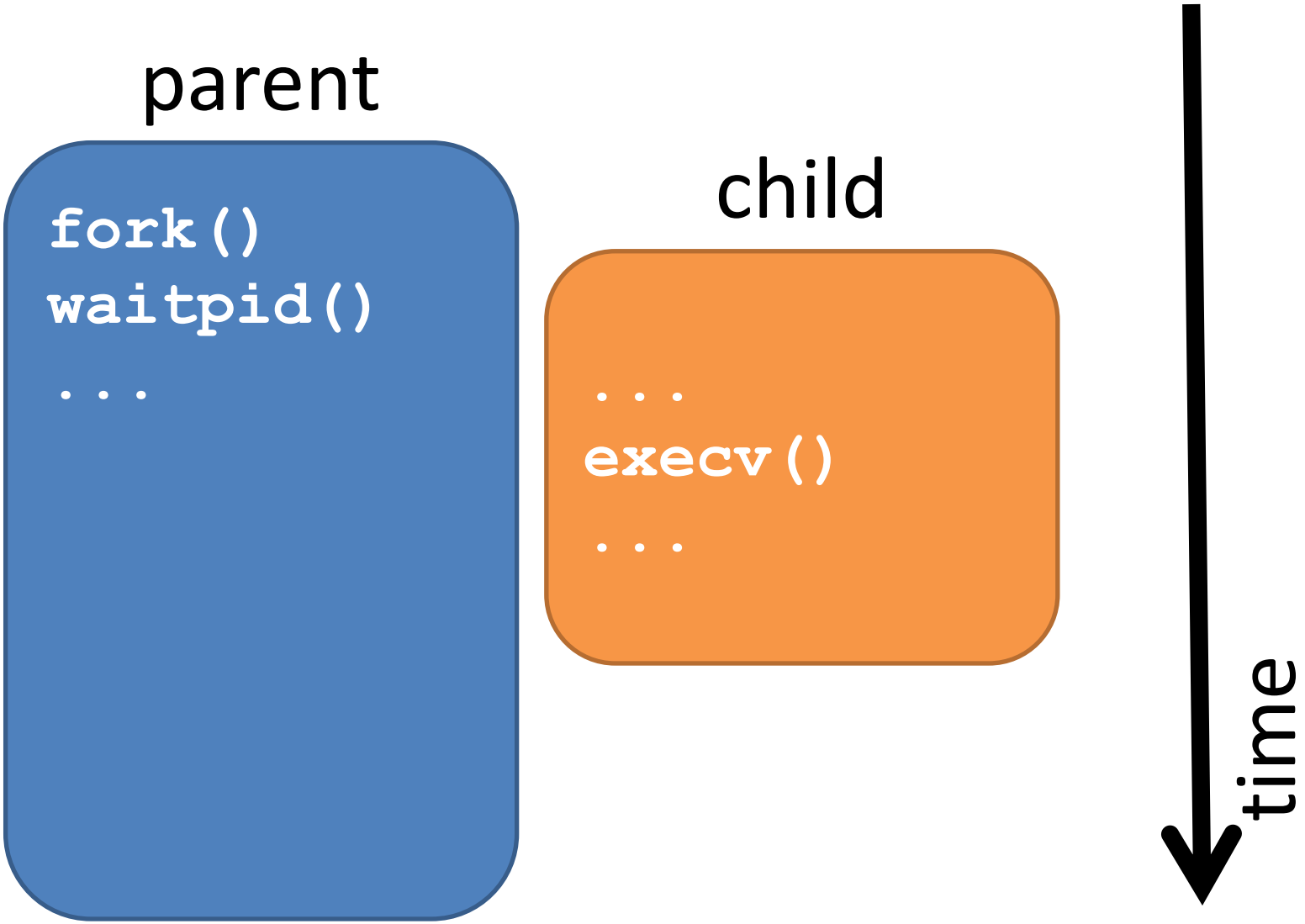
- Blocking
  ```
  waitpid(-1, (int *)NULL, 0);
  ```
- Non-blocking
  ```
  waitpid(-1, (int *)NULL, WNOHANG);
  ```

- **`-1`**
  - status is requested for any child process
- **`(int *) NULL`**
  - if the value of the argument ***stat_loc*** is not a null pointer, information shall be stored in the location pointed to by ***stat_loc***
- **`WNOHANG`**
  - shall not suspend execution of the calling thread if status is not immediately available for one of the child processes specified by ***pid***.

`waitpid()` (foreground)



`waitpid()` (background)



QUESTIONS