Fork, Pipe, Wait

When do you use a fork system call?

- When you want to create a new process but only to create a duplicate of the running process
- Whenever you want to create a new process — fork is the only way.
- When you need a way for one process to communicate with another
- When you don’t have a knife or spoon call

What is the different between the child and parent immediately after fork?

- Nothing
- Their PID's
- The return value from the fork call
- The values of all the variables in memory
- The addresses of the variables in memory

Return value from fork()

0 in the child

PID of child in the parent

-1 if fork failed

Which process executes first after the fork call?

- The parent
- The child
- Either the parent or the child — depending on whose code is written first in the program
- Either the parent or the child — depending on the OS scheduler

Wait Review

- What is wrong with this code example?

```
// fork a child and then in the parent do
int status;
wait(status);
```

Review

- What is wrong with this code example?

```
// fork a child and then in the parent do
int status;
wait(&status);
```

Review

- Would this work?

```
// fork a child and then in the parent do
int *status;
wait(status);
```

Review

- Continuing on

```
// fork a child and then in the parent do
int status;
wait(&status);
```

Review

- What's wrong now?

```
// fork a child and then in the parent do
int status;
wait(&status);

printf("My child returned %d\n", status);
```

Review

- What's wrong now?

```
// fork a child and then in the parent do
int status;
wait(&status);

printf("My child returned %d\n", status);
```

Review

```
int status;
wait(&status);

if WIFEXITED(status) {
   printf("My child returned %d\n",
      WEXITSTATUS(status));
}
```

Pipe Review

If you want two processes to communicate through a pipe

- You need to call fork() then pipe()
- You need to call pipe() then fork()
- You need to call pipe() but may or may not call fork()
- The processes must be parent and child
- The processes must be related

It is important to close the unused ends of the pipe because

- The process on the other end of the pipe uses the fact that the pipe is closed.
- Pipes only work with a single open read end and a single open write end.
- You will run out of file-descriptors of you have too many pipe ends left open.
- Your tobacco will spill out of pipe ends left open