Parallel and Concurrent Programming - CS3D666 - Hour 2: Pipes and Execute

- Execute:

- 1. Write a small program that prints the pid and ppid of the current process (use `getpid()` and `getppid()`).

- 2. Write a program that forks and then executes the program written in 1 using the child process. In the parent program print out the child pid and current pid to ensure they match those printed in 1.

Parallel and Concurrent Programming - CS3D666 - Hour 2: Pipes and Execute

- 3. Modify your program in 1. To accept arguments in main (see notes) and print out the contents of the arguments. Call the program from your program in 2 passing 4 strings.

- 4. Write a program that requests the user for an integer and then passes it back through the `exit()` function to the parent. The parent should wait for the child to terminate before printing the value passed back.

Pipes

- 5. Create a program that opens a pipe, writes to it and then reads from it using a single process (do not use `fork()`).
- 6. Modify the program in 5 to have the parent process write to the pipe and the child process read from it.
- 7. Create a program that creates two pipes and two children processes. Data should be sent by the parent to the children. Data should be read by the children.
- 8. What happens if both children share a single pipe?
- 9. Can a pipe be used to send data in both directions (try it send data from parent to child and then back again)? If not how can you work around this?