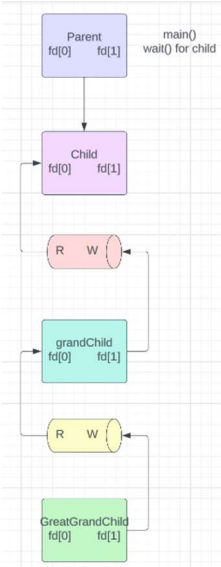


## Program1C

### Processes.cpp Report

- 1) Explain your design and the algorithm for your Processes.cpp, support your explanation using flowcharts or other figures.

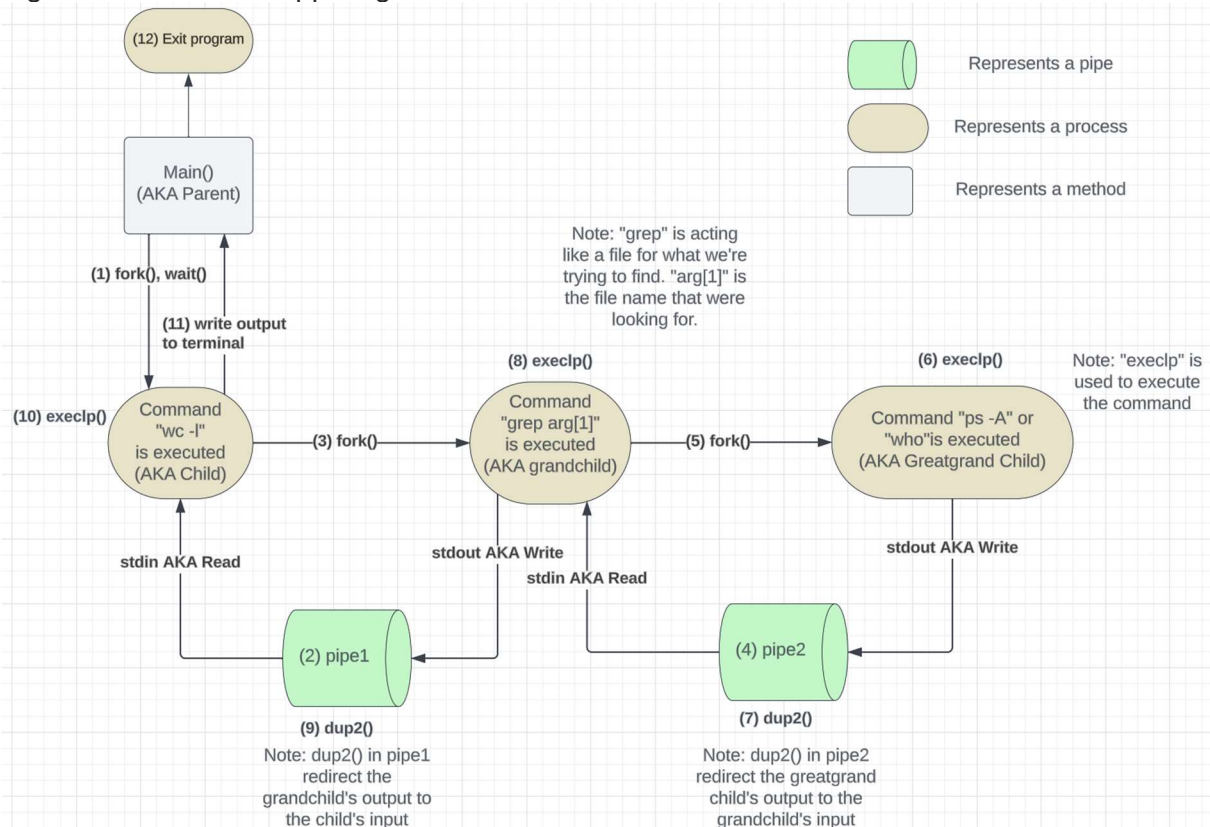
Figure 1.1 – Chapter 3 Lecture Notes



Using figure 1.1 as a reference to create the Processes.cpp file which is shown in figure 1.2.

Seen in figure 1.2, is the Processes.cpp file design, fork() is used to create a parent-child-grandchild-greatgrandchild process. It uses pipes for communicating between each process. The parent will create a child that runs the command "wc -l" which will count the lines. The child creates a grandchild to execute the command "grep arg[1]". The grandchild will create a greatgrandchild which will execute "ps -A" which will list the processes, taken from the input of the first pipe which is sent out through the second pipe.

Figure 1.2 – Process.cpp diagram



## Program1C

### 2) Explain how to test your Processes.cpp.

Figure 2.1 – Compiling and Testing Processes.cpp

The screenshot shows a MobaXterm terminal window with a menu bar (Terminal, Sessions, View, X server, Tools, Games, Settings, Macros, Help) and a toolbar. On the left is a file explorer showing the directory structure of the user's home directory, including folders like .cache, .gnupg, .vscode-server, and files like .bash\_history, .bash\_logout, .bashrc, .profile, .viminfo, .wget-hsts, a.out, jack.txt, processes, and Processes.cpp. The terminal window displays the following text:

```

tjcaole@csslab9: ~
23/01/2024 22:20.10 /home/mobaxterm ssh tjcaole@csslab9.uwb.edu
Please login with your UW NetID password.
X11 forwarding request failed on channel 0
UW Bothell's remote csslab

Visit https://csswiki.uwb.edu for useful lab information.
For lab support questions, please email UWBIT@uw.edu

Check system live resources using your web browser
by navigating to hostname.uwb.edu:9090 e.g. csslab14.uwb.edu:9090

Login with your UW NetID and password.

Last login: Tue Jan 23 17:41:13 2024 from 10.19.200.12
tjcaole@csslab9:~$ g++ Processes.cpp -o processes
tjcaole@csslab9:~$ ./processes tty
1
tjcaole@csslab9:~$ ./processes sys
13
tjcaole@csslab9:~$ ./processes user
0
tjcaole@csslab9:~$

```

Seen in figure 2.1, to compile the Processes.cpp file by typing the following command, noting that the command is case sensitive, “g++ Processes.cpp -o processes”. To test the code we can type the following command “./processes NAMEOFPROCESS”, note that the NAMEOFPROCESS can be any file name, which will result in a number of files that have the same name of NAMEOFPROCESS.

## Program1C

### 3) Output:

Figure 3.1 –Terminal Results

```
tjcaole@csslalab9:~$ ps -A | grep tty | wc -l
1
tjcaole@csslalab9:~$ ps -A | grep sys | wc -l
13
tjcaole@csslalab9:~$ ps -A | grep user | wc -l
0
```

Figure 3.2 – Processes.cpp Results

```
tjcaole@csslalab9:~$ ./processes tty
1
tjcaole@csslalab9:~$ ./processes sys
13
tjcaole@csslalab9:~$ ./processes user
0
```

### Notes:

- 1) Open MobaXterm, don't forget to connect Big-IP Edge Client
- 2) Log in using the following,:  
ssh <netID>@csslalab<9-12>.uwb.edu E.g: ssh tjcaole@csslalab9.uwb.edu  
password: your netid password E.g
- 3) Create a Clion account to download and install the Clion IDE
- 4) Create a cpp. And start coding
- 5) There will be an error stating that function pipe(), wait(), fork(), is "use of undeclared identifier" and the import folder syst/wait.h "file is not found". However when compiled in MobaXterm using, noting cap sensitive, pro