# Report:

## Functionality of processes.cpp

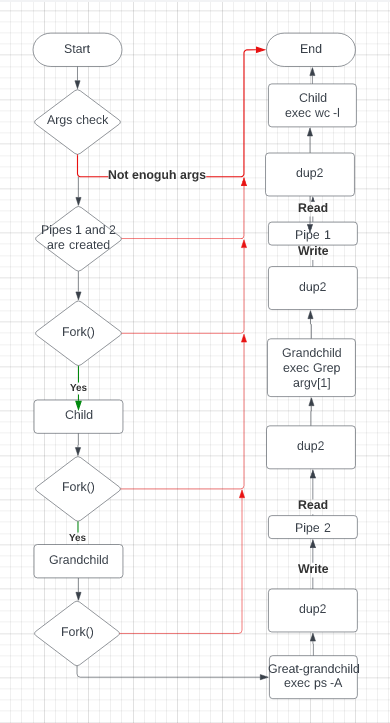
The main purpose of this program is to show the pipeline system that passed commands with the user of pipes. To do this, we needed to split the commands and assigned them through the proper pipes to read and write commands. I used a nested if statements to get this to work. First, I created 2 pipes and checked if it worked. If it did, then we continue, if not we terminate the program. Then I forked once. I had a check for every fork to see if it failed (-1 is fail). I forked 3 times all the way from the child to the great-grandchild. In which, I closed all FD1 (file descriptor) , and the write for FD2. Then I dup2() the FD2 read and executed the “wc – l” command. Then, I close FD2 read, as well as FD1 write. After dup2() for FD1 read and FD2 write, I executed the “grep argv[1]” command. After that, I closed all pipes read and write, but the FD1 write, which I dup2(). Which allowed for the final execution of the “ps -A” command. Lasty, the parent process is waiting for the children ones to be done.

## How to test processes.cpp

Needs to be done on a Linux/Unix terminal.

Use the “ps -A | grep tty | wc -l” to test how many processes of that is running, then use the compiled file for proceeses.cpp with the same argument such as “tty”, the outputs should match,

### Chart



## My Output:

Text

Description automatically generated