**Program 1C Report**

In Figure 1, you can see that we start in the login shell. We first create the two pipes necessary for the children to intercommunicate or read and write to each other. Then we fork from the login shell to the children until we get to the great-grand-child. In our case we have three commands, so we fork three times. For each fork and pipe command we do check for errors. For each child we are going call dup2 system call to create a copy of the file descriptor of the standard outputs and inputs and then execute the commands using exec. For each child we are also going to call wait to make sure that we wait for each child to finish before we allow the parent to execute their command. In the figure below we can see the direction in which each child is going to write and read to the pipe using standard inputs and outputs. When we execute the last child, wc -l, the parent’s wait command will tell the parent that all the children is finished. To know when we are finished we cout “done!” to the terminal and then exit successfully.

A diagram of a shell

Description automatically generated

**Figure 1 – Shell Diagram for command “ps-A | grep argv[-1] | wc -l”**

How to test:

1. Open new terminal
2. To make executable type command: g++ processes.cpp -o processes
3. To execute executable type command: ./processes argv[1] (where argv[1] is the processes name we want to the number of)
4. You will see the number of that process and done! After showing that we have successfully exited.

Outputs

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated