CSS430B Spring 2014

Professor Stephen Dame

Programming Assignment 1 Report

Tell me how to test your Shell.java. Explain the algorithm of your processes.cpp and Shell.java in some statements, using some flowcharts, or whatever.

# About Shell.java

## Current Implementation Features

* Shell.java can take in any combination of commands separated by “&” or “;”
* Number of commands is irrelevant
* Number of arguments is irrelevant
* The last command will execute even if the ending “&” or “;” is not present
  + PingPong abc 1000 & PingPong xyz 1000 &
  + PingPong abc 1000 & PingPong xyz 1000
* Shell will only exit if “exit” is by itself
  + “exit” will exit Shell
  + “exit this program” will not exit Shell and will be treated like a command

**Specification**

Shell Command Interpreter

User types in command followed by a & or ;

& following a command will execute next command concurrently

; following a command will wait until command has completed before executing next command

**Test using the following commands**

PingPong abc 1000 & PingPong xyz 1000 & PingPong 123 1000 &

PingPong abc 1000 ; PingPong xyz 1000 ; PingPong 123 1000 ;

**Custom test (mixing & and ;)**

PingPong abc 1000 & PingPong xyz 1000 ; PingPong 123 1000 ;

**Current Implementation**

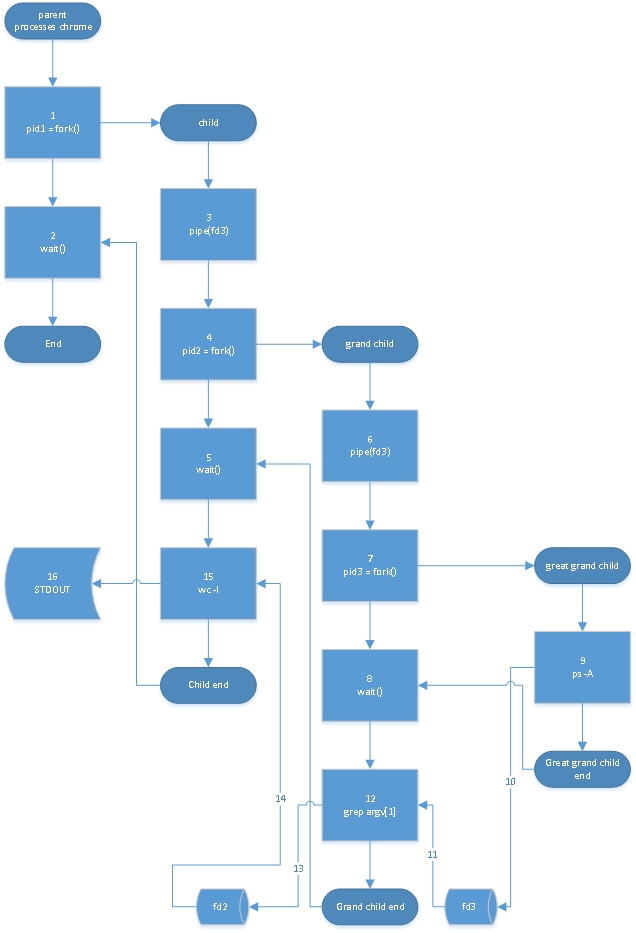
For loop to increment the prompt number

While loop to process and execute commands using starting and ending indexes

**Ideal Implementation**

I don't know, recursive solution was getting too complex

# About processes.cpp



**Specification**

Creating a C++ program that mimics the command of "ps -A | grep argv[1] | wc -l"

Command will get all processes, filter only process that match argument, and then give a final count

**Implementation details**

Parent waits for child with no change to STDIN and STDOUT

Child process executes wc -l using read end of pipe from Grand Child

Grand Child process executes grep argv[1] using read end of pipe from Great Grand Child and outputs to write end of pipe to Child

Great Grand Child process executes ps -A and outputs to write end of pipe to Grand Child

**Test commands**

processes mingetty

ps -A | grep mingetty | wc -l

processes ksand

ps -A | grep kscand | wc -l

processes sendmail

ps -A | grep sendmail | wc -l

**Current Implementation (DONE)**

Nested if statements with each process following the forking process

**Ideal Implementation (Not Started)**

Recursive function

Brainstorming Ideas

Step 1 take input into string array (in this case string would already be set)

Step 2 reverse parse string until pipe is hit

Step 3 send string, end of string, location of pipe

Step 4 create pipe and fork (somehow need to keep track of number of pipes, vector probably)

step 5 repeat step 2 using (pipe location -1) as end and (pipe location-2) as start

step 6 if start is at 0, build command using start and end

step 7 process closes the read end of the pipe that existed

step 8 process closes write end of pipe that process created

step 9 use dup2 to link pipe to STDIN and STDOUT

step 10 execute command