CS 100 Systems Programming
Homework 5

- Linux (10 pts): Define a complete Makefile for your programs below (do not use any default rules). Be sure it adds symbol table info for gdb. Define a target "clean" that removes all rebuildable files. Define a target called "strip" that removes symboltable info from my_shell. Define a target called "print" that will generate a PDF listing of the header files and the .cpp files involved in your shell.
- C++ (50 pts): Define a class hierarchy with the following structure. You get to pick the problem domain, and I suggest you pick something you would like to work on in the future (do ***not*** use provided example, pick *your own* domain). Here are the details:
    - (10 points) Define an abstract base class (e.g., Shape) with one pure virtual function (e.g., area). This class should describe an abstract category (example a Shape). Be sure this class has a *constructor* and *at least one data member*.
    - (20 points) Derive *three* classes directly from your abstract base class and make them concrete by providing implementations for the virtual function you introduced in your abstract base class. *The methods for each of these three classes must actually **behave differently** from one anothe*r (example: Circle area vs. Square area vs. Triangle area each use a different formula).
    - (10 points) Write one *interesting* polymorphic function that operates on any type derived from your abstract base class. It should take a list (or array) of instances of your abstract base class (e.g., Shape *) and call the virtual function on each to do something interesting. Also, have each of these virtual functions print something so you can see which is being called.
    - (10 points) Write a main program that builds a list (or array) of several instances of each of your concrete classes, then pass this off to your polymorphic function.
- Systems Programming (40 pts): **Command shell, next step.** Write the next version of the program called my_shell, which is the next step towards a simple command shell. Add the following features:
    a. (20 pts) add *pipelines* of commands specified with '|' between commands
    b. (20 pts) add *background jobs* when the command ends with '&'
    c. You can test my_shell with these commands:
    % tee newOutputFile > newOutputFile < existingInputFile &
    % cat < existingInputFile | tee newOutputFile > newOutputFile
    % cat < existingInputFile | tee newOutputFile > newOutputFile &
    % cat < existingInputFile | tr A-Z a-z | tee newOutputFile | tr a-z A-Z > newOutputFile
    % cat < existingInputFile | tr A-Z a-z | tee newOutputFile | tr a-z A-Z > newOutputFile &