Programming Assignment 1

CS450 Fall, 2021

1. This assignment is an individual effort. It is due on 9/16/2021

2. **Requirements:**
   This programming assignment is inspired by one given at MIT
   (https://pdos.csail.mit.edu/6.828/2012/homework/xv6-shell.html). The MIT assignment
   asks the students to implement the pipe "|" and IO indirection ">" operators and
   somewhat more. We ask you to implement the sequence command execution operator
   ";" and the parallel execution operator "&" and more. You should do this assignment by
   modifying the skeleton code attached. Invoke your shell from the shell that comes with
   xv6 or from a robust UNIX.
   1) After your shell has started, it will give a prompt to the user. You should use
      "$CS450" for the prompt.
   2) If the user types `cmd1;cmd2;…;cmdn` after the prompt, `cmd1` will get
      executed, followed by `cmd2` and so on. After `cmdn` is executed, your shell will
      give a new prompt in a new line, ready to execute the next command line.
   3) If the user types `cmd1&cmd2&…&cmdn`, all commands will get executed in
      parallel. After all terminate, your shell will give a new prompt in a new line,
      ready to execute the next command line.
   4) The commands `cmd1` etc. are real commands supported by xv6 and they take
      arguments. An example is the command `echo  "A"`.
   5) Your shell shall support command lines with 3 or more commands connected by
      the ";" and "&" operators. The operators are of equal rank and they will be
      executed from left to right. As an example, the string "`cmd1&cmd2;cmd3`" will
      see `cmd1` and `cmd2` executed in parallel, then `Cmd3` after they both terminate.
   6) A command string terminated by a "&" is illegal; but it is fine with ";"'

3. **Deliverables:**
   1) Source and executable objects with a README on how to build and execute
      them.
   2) Screenshots showing your shell and test cases being executed. An example is
      attached.
   3) A copy of the modified skeleton code, showing the modifications that you make
      and with comments that explains how your code works.
   4) The test data that you use and explanation on why the test data is of good
      quality. If you use the equivalence partitioning method to generate your test
      data, describe your equivalence partitions.