CS240 - Operating Systems
Assignment #5
Spring 2017

For this exercise, you are to add to your "shell" program started in assignment 4. Implement the following additions:

- Your shell is to allow for inputting multiple commands separated by ";" e.g., `ls ; ps` will execute first `ls` then `ps` .
- Upon starting your shell, it should execute simple one-line commands stored in a file called `uishrc` . Such a feature is used to set up your shell with commands that the user would otherwise have to execute manually every time upon starting the shell.
- Include variable name substitution in your shell. A variable value is defined like: `VAR=test` . Its value can be substituted in a shell command by including the variable name, preceded by a `$`, where the variable value is to be substituted. For example:

```
VAR=test
echo $VAR
```

- Implement the `PATH` variable mechanism, like that used in `bash` and other shells. Specifically, when your shell tries to execute a "command," it prepends the values of the `PATH` variable to the command name, then executes the program corresponding to that path.
- Implement a "history" mechanism, which will allow the user to display the last 20 shell commands (including duplicate commands and those that were unsuccessful). Note that `bash` includes this feature, although it is not limited to just 20 lines.
  - Typing `history` should display up to the last 20 lines that have been typed, preceded by the command number. Even though only 20 lines are saved, the command number should increment for each command that has been typed.
  - Typing `!!` should re-execute the last command.
  - Any of the commands in the list can be re-executed by typing a `!` followed by the command number. For instance, typing `!123` should execute command number `123` (assuming that command 123 is one of the 20 saved commands in the list.)
- Implement simple piping, e.g., `ls | more` will pipe the output of `ls` to `more`.

As in the last assignment, your shell should parse the input, then use the fork/exec mechanism to implement the execution of the command.