COS 333 – Advanced Programming Techniques

Programming Assignment I

Assigned: Monday February 12, 1996

Due: 11:59pm, Monday February 26, 1996

Submission Procedure: TBA

Program: In `~cs333/src` you will find a file called `sh.c` that implements a rudimentary shell. It is your task to build a simple extended shell (*sesh*) using `sh.c` as a base. This assignment is in three parts; each part implements a new shell function.

(i) Implement a `print` command as a built-in command for your shell. Your `print` command must write its arguments—separated by a space—to the standard output. After printing all arguments, the `print` command must write a newline to standard output.

You may use the code for `print` presented in lecture.

(ii) Implement variables for your shell. In your implementation, a variable named x is assigned the value v by the command:

`$` x`=`v

Note that there is no white-space in this command—your shell can therefore identify a variable assignment by searching for an '`=`' in a command.

The value of an assigned variable is always a string. A shell command can refer to the value of a variable via `$`x where x is typically a previously assigned variable. The value of a variable that has not been assigned a value is to be the empty string (`""`).[1]

(iii) Implement a basic alias mechanism. Extend your shell to recognize two new commands: `alias` and `unalias`. The first has the syntax

`alias` *new-name old-name* $sym_1 \cdots sym_n$

which your shell must interpret to mean that subsequent uses of *new-name* as a command with arguments arg_1 through arg_n are to expand[2] into:

old-name $sym_1 \cdots sym_n \; arg_1 \cdots arg_n$

The second, `unalias`, has the syntax

`unalias` *name*

and removes *name*'s alias binding if it exists.

Sample Transcript: A transcript of a typical interaction that your shell must handle is available as `~cs333/assignment1/transcript1`

---

[1] In C, the empty string is not identical to the null string—the empty string consists of the single character with ASCII value zero (0) whereas the null string is a `(char *)` pointer with value `NULL`.

[2] You need not handle recursive expansions (*i.e.* `alias cmd cmd arg1` ⋯), but you should document your shell's behavior on this case.