Slide 1

Administrivia

- Homework 1 due today at 5pm. Hardcopy please.

Slide 2

Shell Scripting — Review/Recap

- What you type at the command line in most UNIX shells forms a programming language; in a sense an interactive shell is a REPL for the scripting language? The language has variables, conditional execution, and loops.
- “Shell script” is a program in a shell language. By convention the file starts with something that says which shell is to be used.
- Shell scripts often make heavy use of “filter” programs, plus I/O redirection and pipelines. “Command substitution” is another way to combine programs.

Slide 3

Shell Functions and Parameters — Review

- Define functions as described previously — `function` followed by name, parentheses, then function definition in curly brackets. Separate/end commands with `;` or newlines.
- Parameters for functions and shell scripts are positional — `$0` for function name, then `$1`, etc. `$*` is a list of all parameters; `$#` is the count of parameters, not including `$0`.
- Call functions or shell scripts by giving name and then parameters, separated by whitespace. (If a parameter should include whitespace, use quoting or escape characters.)

Slide 4

Arithmetic

- Most basic/portable way probably `expr`. Example: `n=‘expr $n + 1‘`.
- In `bash`, can also use double parentheses. Example: `n=$((n + 1))`.
- (But if you're doing significant calculations, you should probably be using some other tool — `awk`, `bc`, `dc`, or a program in a "real" programming language.)

Reading from Standard Input

Slide 5

- To read from shell's / script's standard input: `read`. Example:

```
echo "Do you really want to do this?  (y/n)"
read ans
if [ "$ans" = "y" ] ....
```

"Here" Documents

Slide 6

- We talked about redirecting input and output. One more option for input, useful in scripts, is to get it from the script itself — "here" document. Example:

```
#!/bin/sh
mail -s "a subject" bmassing << EOF
hello
I am here
who are you?
is this fun?
EOF
```

Other Useful Things

- `getopt` — process command-line options (to help you write scripts that accept options in any order, in the same way most UNIX commands do).
- Remember `pushd` and `popd`, for temporarily changing to another directory and coming back.

Slide 7

Shell Script Examples

- (We'll write these in class.)
- Script to rename all `.htm` files to `.html` (or something similar). (Tricky part is making it deal correctly with filenames that contain spaces.)
- Script with a recursive function to compute factorial.
- Other examples as time permits?

Slide 8

Minute Essay

- Have you found occasion yet to use anything you've learned in this class?
- Anything else you want to know about shell scripts?