Slide 1

Administrivia

- Reminder: Homework 2 due Wednesday at 5pm.

Slide 2

Shell Programming — Review

- Input to many/most shells forms a programming language, with variables and constructs for selection and repetition.
- Can type these on the fly, or save in file as “shell script”.

Arithmetic

Slide 3

- Shell supports simple *integer* 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.

`dc` and `bc`

Slide 4

- Both are simple text-mode calculator programs. `dc` uses reverse Polish notation, `bc` the more familiar algebraic notion.
- Both are "arbitrary-precision", which can be useful. Both support non-integer values, but how to set "precision" can be tricky. Details in their `man` pages.
- Used interactively, `bc` may be more useful, since you can use variables within it.
- Both are useful in shell scripting, e.g.,

```
echo "2 + 3" | bc
echo "2^10" | bc
```

Slide 5

Reading from Standard Input

- 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" ] ....
```

(Why the dots? if nothing is read, `$ans` may be empty, with possibly awkward results. May be okay to omit, but a lot of shell scripts use them.)

Slide 6

"Here" Documents

- 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
```

Slide 7

Other Useful Things

- Shell option `-x` can be helpful in debugging (`set -x` in script, or `bash -x myscript`).
- `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 8

Shell Script Examples

- (Examples as time permits.)

Slide 9

Shell Scripts — “the Ugly”, Revisited

- If variables are set in a “subshell” their values seem to disappear when it exits. An example is piping something into a `while read` loop.
- How to fix? simplest way is just to find an alternative to piping (“here” documents, maybe, or other input redirection).
- (More about subshells next time.)

Slide 10

Minute Essay

- How’s the pace of the class so far? Homeworks about the right amount of outside-class time, too much, too little (“as if”?) ?