CS114: Lecture 9
More shell scripting

HW3 due Wednesday...

A note on mountpoints

- What disk is `/lib` on?
- Mount doesn't show an entry for `/lib`
- But it must be on SOME disk...
- Perl is `/usr/bin/perl`; mount has an entry for `/usr`, but not `/usr/bin`. So where is Perl?

Comments and newlines

- # to end of line is totally ignored
  - (like // in Java or C++)
- # is the “comment character” in many scripting languages
  - sh, csh, Perl, Python, Tcl, ruby, ...
  - Why?
- \ at end of line continues to next line (useful for looooong pipes)

Math in shell scripts

- `expr 1 + 3`
  - Prints 4
- `expr 2 \* 4`
  - Prints 8
- Spaces are important!
- `expr 1+ 3` – error
- `expr 1+3`
  - Prints `1+3` (it's a string)

Bourne shell scripting: While loops

```
#!/bin/sh

i=""

while [ "$i" \!= xxxxx ]; do
   echo Iteration $i
   i=x$i
done
```

- How many loops?
- Why "`$i`" above?
- **`until ... do .. done`** (just like **`while`** but backwards)

`case` – many tests at once

```
#!/bin/sh


case $1 in
  [a-zA-Z]*) echo "$1 starts with a
  letter" ;;
  [0-9]*) echo "$1 starts with a
  number" ;;
  *) echo "$1 starts with a
  punctuation mark" ;;
esac
```

`case` in general

```
case word in
   pattern) command ;;
   pattern) command ;;
   ...
esac
```

Command-line options in a script

- `shift`
  - (not valid code) 1 = $2; 2 = $3; 3 = $4 ...
  - `shift 3` (1=$4; 2=$5; ...)
- `getopts hs:f:x arg`
  - Accepts `-h, -s` *`argument`*`, -f` *`argument`*`, -x`
  - Sets `OPTARG` = value of argument
  - Sets `OPTIND` = index of argument

Command-line idiom

```
while getopts hs:f:x arg; do
  case $arg in
  h) echo "Got -h" ;;
  s) echo "Got -s $OPTARG" ;;
  f) echo "Got -f $OPTARG" ;;
  x) echo "Got -x" ;;
  esac
done
shift `expr $OPTIND - 1`
```

“here-docs”

- Including a bunch of text in your shell script

```
#!/bin/sh

person=$1

cat <<EOM | sed 's/hb/Happy Birthday/'
hb, dear $person
hb to you
EOM

echo '(Blow out candles now)'
```

Subshells

- What if I want to change some settings *just for one command*?
- `(cd /usr/bin; ls ??) | sort -r | head > usr-bin-last`
- Any sequence of commands in () is executed by a separate shell process

Shell functions

```
#!/bin/sh

hb() {
   cat <<MSG
Happy Birthday, $1
and many more...
MSG
}

hb Jay
for person in Alice Bob Charlie; do
  hb $person; done
```