Week 02M: Review & Shell Scripting

Thuy Vu

Reminders

- Office Hours
  - Thuy Vu — 09–10a Tuesdays & Thursdays, BH2432
- Assignment 2
  - Shell Scripting
  - `web.cs.ucla.edu/classes/winter17/cs35L/assign/assign2.html`
  - Time due 23:55 next Saturday, January 21
- hard links vs soft links?

Assignment-2 — Last Week

1. pipeline `program_1 | program_2 | program_3 | ...`
   - `program_1`'s stdout ≡ `program_2`'s stdin
   - `me$cat assign1.html | grep HTML`
2. locale
   - `echo -e "v 1000\nv10 01\nv 1002" | LC_ALL=en_US.UTF-8 sort`
   - `echo -e "v 1000\nv10 01\nv 1002" | LC_ALL=C sort`
3. `grep, sort, comm, tr`
   - `me$echo "password a1b2c3" | tr -d [:digit:]` → `abc`
   - `me$echo -e "a\nb\nc" > file1`
   - `me$echo -e "a\nd\nc" > file2`
   - `me$comm file1 file2`
   - `me$comm -23 file1 file2`
4. sed stream editor
   - substitution – `s/regular expression/replacement/flags`
   - `me$echo "<td>hello</td>" | sed "s/<[^>]*>//g"`
   - `me$sed "13q;d" hwnwdseng.htm`
5. interpreter v.s. compiler

Shell Programming

- shell is the traditional user interface of UNIX; e.g. `bash`, `csh`, `ksh`, ...
- shell is a command-line *interpreter*; e.g. `/bin/bash`, `/bin/csh`, ...
- *shell-script* is a file containing shell commands, to execute:
  1. `me$./scriptfile` (to specify, e.g. `#!/bin/sh`, in 1^{st} line)
  2. `me$sh scriptfile`
  3. `me$.  ./scriptfile` (read *dot-space-dot-slash*)
  4. `me$source scriptfile`

Bourne shell

```
#!/bin/sh
if [ $days -gt 365 ]
then
   echo This is over a year.
fi
```

```
#!/bin/sh
i=2
j=1
while [ $j -le 10 ]
do
   echo '2 **' $j = $i
   i=`expr $i '*' 2`
   j=`expr $j + 1`
done
```

C shell

```
#!/bin/csh
if ( $days > 365 ) then
   echo This is over a year.
endif
```

```
#!/bin/csh
set i = 2
set j = 1
while ( $j <= 10 )
   echo '2 **' $j = $i
   @ i *= 2
   @ j++
end
```

https://en.wikipedia.org/wiki/C_shell

Shell Programming

- shell declaration header `#!/bin/sh`
- tracing `set -x` to turn on; `set +x` to turn off
- print `echo -e "hello\nworld"` or

  `printf "%.3e\n" 12345.54321`
- variable ``nfiles = `ls -A $1 | wc -l` #number of files``
- quote
  - single quote ' ' – literal meaning; e.g. `me$ str='$hello'; echo $str`
  - double quote " " – $ and `; e.g. ``me$ echo "file count:`ls | wc -l`"``
  - backtick ` ` – execute the command

Spell-checking Hawaiian

- Spell-checking Hawaiian
  - `http://mauimapp.com/moolelo/hwnwdseng.htm`
  - `sed`, `cat`, `tr`, `grep`, `sort`, `comm`

Find Duplicate Files

- Find duplicate files `sameln D` ⟶ all duplicated regular files to hard links.
  - should keep file name starting with . or first-lexicographically-ordered (.Y, .X, X, A, B)
  - regular files, ignore symbolic links and directories
  - names contain space, asterisk, leading "–", return character, ...