Introduction to Make

and

Python Scripting

CS 35L

Spring 2018 - Lab 3

Getting Started

- Login to seasnet
- Download coreutils to a temporary directory
  - how can we download a file?
- Untar\Unzip it
  - How do you unzip a file?
  - man tar
  - *cd* into the newly created coreutils folder

Tar commands

- `tar -cvf <tarfilename.tar> <target directories>` - creates tar file.
- `tar -tvf <tarfilename.tar>` - list tar file contents
- `tar -xvf <tarfilename.tar>` - extracts tar file
- Can add `-z` flag for newer LINUX distros with gnuzip for automatic compress/decompress (.gz suffux).
- Otherwise try compress command (`.Z` suffix)
- USAGE:
  - Always create tarfile in target directory (relative file/directory names)
  - Always list tarfile before extracting (insure relative file names)
  - Always extact tarfile in target directory (relative file/directory names)
- Example:

```
tar -xzvf ~/bb-1_3a_tar.gz
```

Compiling from scratch

- Common scenario
  - You download a utility from the internet to your unix machine
  - There are no binaries, but source code and makefile is available
  - Compile and build to install it
  - Reading text files in the program folder gives clues how to install the program
    - Usually INSTALL, README, readme.txt, install.txt and so on

Compiling from scratch

- Common scenario
  - The order of compilation is usually:
    - `./configure`
    - **`make`**
    - **`make install`**
  - man make for more details
  - view Makefile in the programs folder for details
    - e.g. search for “install:”

Configure script

- Designed to aid in developing a program to be run on a wide number of different computers

- `configure` is application specific
  - software provides it's own configure script

- Creates the Makefile
  - Can change default behavior with options
  - `./configure --` help for more info

Makefile and make

- We need a file that instructs make how to compile and link a program. Called a Makefile.
- The make program allows you to use macros, which are similar to variables to codify how to compile a set of source code.
  - Macros are assigned as BASH variable:
    - CFLAGS= -O -systype bsd43
    - LIBS = "-lncurses -lm -lsdl“
- Makefile is invoked with
  - make <target_name>

Standard “targets”

- People have come to expect certain targets in Makefiles. You should always browse first, but it's reasonable to expect that the targets all (or just make), install, and clean will be found
  - **`make`** - compile the default target
  - **`make all`** - should compile everything so that you can do local testing before installing things.
  - **`make install`** - should install things in the right places. But watch out that things are installed in the right place for your system.
  - **`make clean`** - should clean things up. Get rid of the executables, any temporary files, object files, etc.
- Link to more info on makefile

Makefile example

- Makefile:

```
SHELL = /bin/sh
MAKE = make
CC = g++
LIBS=
CFLAGS=-DSIGSETJMP -O

hello: main.o hello.o factorial.h
        ${CC} ${CFLAGS} -o $@  main.o hello.o ${LIBS}

clean:
        rm -f *.o
```

- Run as make; make clean

Compiling coreutils

- Go into coreutils directory. This is what you just unzipped.
- Read the INSTALL file on how to configure "make," particularly the --prefix flag.
- Run the configure script so that when everything is done, coreutils will be installed into your temporary directory
- Compile it: make
- Install it: make install

Bug appears with newly built coreutils

```
[User:-)@lnxsrv07 ~/cs35L/lab3/coreutils/bin]$ ls -l /bin/bash
-rwxr-xr-x 1 root root 960376 Jul  8  2015 /bin/bash
```

```
[User:-)@lnxsrv07 ~/cs35L/lab3/coreutils/bin]$ ./ls -l /bin/bash
-rwxr-xr-x 1 root root 960376 2015-07-08 04:11 /bin/bash
```

Notice the difference between invoking ls commands above

Applying the Patch

- Read the patch bug report
  - lists.gnu.org/archive/html/bug-coreutils/2009-09/msg00410.html
- Understand what part of the code is being fixed

Applying the Patch

```
diff --git a/src/ls.c b/src/ls.c
    index 1bb6873..4531b94 100644
    --- a/src/ls.c
    +++ b/src/ls.c
    @@ -2014,7 +2014,6 @@ decode_switches (int argc, char **argv)
                  break;

                case long_iso_time_style:
    -           case_long_iso_time_style:
                  long_time_format[0] = long_time_format[1] = "%Y-%m-%d %H:%M";
                  break;

    @@ -2030,13 +2029,8 @@ decode_switches (int argc, char **argv)
                       formats.  If not, fall back on long-iso format.  */
                    int i;
                    for (i = 0; i < 2; i++)
    -                 {
    -                   char const *locale_format =
    -                     dcgettext (NULL, long_time_format[i], LC_TIME);
    -                   if (locale_format == long_time_format[i])
    -                     goto case_long_iso_time_style;
    -                   long_time_format[i] = locale_format;
    -                 }
    +                 long_time_format[i] =
    +                   dcgettext (NULL, long_time_format[i], LC_TIME);
                  }
                }
              /* Note we leave %5b etc. alone so user widths/flags are honored.  */
```

Apply the Patch

- Just use an editor (eg. emacs, vim).
- Recompile it: make
- A new executable ls file is created
- Compare results between new ‘correct’ executable and ‘buggy’ installed version
- Did the patch fix the bug?

General tarball/make example

```
tar -vxzf <gzipped-tar-file>
cd <dist-dir>
./configure
make
make install
make clean
```

Homework

Python Scripting

Running Python scripts

- Download *randline.py* from assignment website
- Make sure it has executable permission:

  chmod +x randline.py
- Run it, for example

  ./randline.py –n 4 filename

  n: is an option indicating the number of lines to write

  4: is an argument to n (you can use any number)

  Filename: is a program argument

Example run

- I downloaded text version of ‘Alice in Wonderland’ for testing the script.
  - Available free from:
    - www.gutenberg.org/ebooks/19033.txt.utf-8
  - Ran the script on file

```
> ./randline.py  -n 4 alice_in_wonderland.txt
Gutenberg-tm electronic work under this agreement, disclaim all
[Illustration]
with the permission of the copyright holder, your use and distribution
that she had put on one of the Rabbit's little white kid-gloves while
```

What does the script do?

Python Walk-Through

```
#!/usr/bin/python

import getopt, random, sys

class randline(file):
       def __init__(self, filename):
              f = file (filename, 'r')
              self.lines = f.readlines ()
              f.close ()

       def chooseline(self):
              choice = random.randrange (len
     (self.lines))
              return self.lines[choice]

def usage (e):
       sys.stderr.write ('randline.py: %s\n' % e)
       sys.stderr.write ('''\
Usage: randline.py [OPTION]... FILE

Output a line selected randomly from FILE. Options:

       -n LINES Output LINES lines (default 1).
''')
       sys.exit (1)
```

Tells the shell which interpreter to use

Import statements, similar to include statements

The beginning of the class statement: randline
- The constructor
  - Creates a file handle
  - Reads the file into array of strings called lines
  - Close the file

- The beginning of a function belonging to randline: chooseline
  - Randomly select a number between 0 and the size of lines
  - Returns the line corresponding the randomly selected number

The beginning of a function: usage
- Write the error
- Write instructions on how to use randline.py.
- ***It follows similar formatting syntax as printf in C***
- ***Note the use of the "triple quote"***
- ***Note the use of the "line continuation character"*** ****
- ***You don't normally need \ when you're using triple quotes, but we threw it in there for kicks***

Exit with error

Note the use of indentation for encapsulation.

Python Walk-Through

```
if __name__ == '__main__':
	opt = {}
	opt['n'] = 1

	try:
		opts, args = getopt.getopt (sys.argv[1:], 'n:')
			for option, value in opts:
			v = int (value)
		if v < 0:
			raise ValueError, 'negative count: %d' % v
		opt[option[1]] = v
	except (getopt.error, ValueError), e:
		usage (e)

	if len (args) != 1:
		usage ('wrong number of operands')
	input_file = args[0]
	line_count = opt['n']

	try:
		generator = randline (input_file)
		for i in range (line_count):
			sys.stdout.write (generator.chooseline ())
	except IOError, e:
		sys.stderr.write ('randline.py: %s\n' % e)
					sys.exit (1)
```

This tests whether or not this script is being imported or run. When its name is “__main__” its being run.
Initialize an array

Start an exception catching block
Get option and program args. If an option requires an arg, include “:”
For each option-value pair
Get the value
If the value is less than 0
Throw/Raise an exception
Store the value in opt array at option[1]
Handle the exception ValueError
Print the usage info

If the number of program arguments is not 1
Print the usage info
Store the name of the input_file
Store the number of lines from the opt array

Start an exception catch block
Create a randline object
Range returns an array [0,1,…,line_count].
Write out the line
Handle the exception IOError
Write error
Exit with an error