Shell Scripting (1)

CS 35L - Winter 2020 - Week 2

Reference: Darrell Carbajal 2007 slides

Interpreted Languages vs. Compiled Languages

- Compiled languages: The **compiler** converts the source code to machine code, which is then executed by the machine. E.g. C/C++.
- Interpreted languages: The **interpreter** directly executes statements from the source code.

Interpreted Languages vs. Compiled Languages

- A language is sometimes not accurately categorized.
- E.g. Java - converted to Java Bytecode and then interpreted/executed by Java Virtual Machine(JVM).
- E.g. CPython - converted to Python Bytecode (.pyc files) then interpreted by Python Virtual Machine.

Interpreted Languages vs. Compiled Languages

- Advantages:
    - Compiled Languages: Performance - the compiled program often runs faster than interpreting the source code on the fly.

    - Interpreted Language: Portability/ Platform independence.

Shell

- Shell is a user interface for access to an operating system’s services (usually CLI).
- An outermost layer around the OS Kernel.
- Common Unix shells (for Unix-like OS):
  - Bash, zsh, sh, csh...

Shell

- Shell is a user interface for access to an operating system's services (usually CLI).
- An outermost layer around the OS Kernel.
- Common Unix shells (for Unix-like OS):
  - Bash, zsh, sh, csh...

- But before Shell scripting, we need to know **Regular Expressions** (REGEX)!

Regular Expressions

- A sequence of characters to define a **search pattern**.
- Usually used for texts.
- E.g. “[a-z]*CS35L”: zero or more occurance of lowercase alphabets followed by the string “CS35L”.
- Basic vs. Extended
  - https://www.gnu.org/software/sed/manual/html_node/BRE-vs-ERE.html
  - https://www.gnu.org/software/grep/manual/html_node/Basic-vs-Extended.html

RegExLib.com Regular Expression Cheat Sheet (.NET)

ers Defined	Metacharacter	
f a string.	^abc	abc, ab
a string.	abc$	abc, en
aracter (except \n newline)	a.c	abc, aa
tion.	bill|ted	ted, bil
quantifier notation.	ab{2}c	abbc
set of characters to match.	a[bB]c	abc, aB
grouping of part of an expression.	(abc){2}	abcabc
ore of previous expression.	ab*c	ac, abc
ore of previous expression.	ab+c	abc, ab
of previous expression; also forces minimal matching when an expression might match several within a search string.	ab?c	ac, abc
	a\sc	a c
ng one of the above, it makes it a literal instead of a special character. Preceding a special matching er, see below.		

acter Escapes http://tinyurl.com/5wm3wl

ry cters	Characters other than . $ ^ { [(|)] } * + ? \ match themselves.
	Matches a bell (alarm) \u0007.
	Matches a backspace \u0008 if in a []; otherwise matches a word boundary (between \w and \W characters).
	Matches a tab \u0009.
	Matches a carriage return \u000D.
	Matches a vertical tab \u000B.
	Matches a form feed \u000C.
	Matches a new line \u000A.
	Matches an escape \u001B.
	Matches an ASCII character as octal (up to three digits); numbers with no leading zero are backreferences if they have only one digi correspond to a capturing group number. (For more information, see Backreferences.) For example, the character \040 represents a
	Matches an ASCII character using hexadecimal representation (exactly two digits).
	Matches an ASCII control character; for example \cC is control-C.
0	Matches a Unicode character using a hexadecimal representation (exactly four digits).
	When followed by a character that is not recognized as an escaped character, matches that character. For example, ***** is the same a

r Classes http://tinyurl.com/5ck4ll

	Matches any character except \n. If modified by the Singleline option, a period character matches any character. For more information, see Regular Expression Options.
	Matches any single character included in the specified set of characters.
	Matches any single character not in the specified set of characters.
	Use of a hyphen (–) allows specification of contiguous character ranges.
	Matches any character in the named character class specified by {name}. Supported names are Unicode groups and block ranges. For exampl Ll, Nd, Z, IsGreek, IsBoxDrawing.
	Matches text not included in groups and block ranges specified in {name}.
	Matches any word character. Equivalent to the Unicode character categories [\p{Ll}\p{Lu}\p{Lt}\p{Lo}\p{Nd}\p{Pc}]. If ECMAScript-compliant behavior is specified with the ECMAScript option, \w is equivalent to [a-zA-Z_0-9].
	Matches any nonword character. Equivalent to the Unicode categories [^\p{Ll}\p{Lu}\p{Lt}\p{Lo}\p{Nd}\p{Pc}]. If ECMAScript-compliant behavior is specified with the ECMAScript option, \W is equivalent to [^a-zA-Z_0-9].
	Matches any white-space character. Equivalent to the Unicode character categories [\f\n\r\t\v\x85\p{Z}]. If ECMAScript-compliant behavior is specified with the ECMAScript option, \s is equivalent to [\f\n\r\t\v].
	Matches any non-white-space character. Equivalent to the Unicode character categories [^\f\n\r\t\v\x85\p{Z}]. If ECMAScript-compliant behav is specified with the ECMAScript option, \S is equivalent to [^ \f\n\r\t\v].
	Matches any decimal digit. Equivalent to \p{Nd} for Unicode and [0-9] for non-Unicode, ECMAScript behavior.
	Matches any nondigit. Equivalent to \P{Nd} for Unicode and [^0-9] for non-Unicode, ECMAScript behavior.

tr

- Translate characters.
- Copies the standard input to the standard output with substitution or deletion of selected characters.
- E.g.
  - tr "[:lower:]" "[:upper:]" < file1
  - Translate the content of file1 to upper-case.

grep

- Search for patterns in a file.
- Grep: simple patterns and basic regular expressions (BREs)
- Egrep: extended regular expressions (EREs).
- Fgrep: only for fixed patterns.
- E.g. “grep ‘Eggert’ ucla_cs.txt”: grep all lines containing..
- E.g. “grep ‘^abc’ foo”: grep all lines starting with abc.
- E.g. “egrep '19|20|25' calendar”: look for 19 or 20 or 25 in the file calendar.

sed

- Stream editor. Can replace part of the file.
- sed ‘(s)/regex/replText’
- E.g 1. sed ‘s/:.*//’ myFile: remove everything after the first colon in myFile. (s: substitution - “//”: substitute with “”)
- E.g 2. Sed ‘5 s/A/B/2’ foo: replace the 2nd occurrence of A in the 5th line with B. (‘s/A/B/g’: all occurrence of A in a line)
- E.g 3. sed '/^#/d;/^$/d' a.txt: delete all lines that are empty or start with #.

locale

- Define user’s language, region, etc.
- Need to set locale: export LC_ALL=’C’
- So that “sort” will give us the correct order.

Redirection

- “<” takes file as input.
  - E.g. “sort < test.txt”
- “>” outputs to file.
  - E.g. “sort < test.txt > sorted_test.txt”
- “>>” appends to file
- “p1|p2”: pipelining: p1’s output is p2’s input.

Pipelining

```
$ who                          Who is logged on
tolstoy tty1 Feb 26 10:53
tolstoy pts/0 Feb 29 10:59
tolstoy pts/1 Feb 29 10:59
tolstoy pts/2 Feb 29 11:00
tolstoy pts/3 Feb 29 11:00
tolstoy pts/4 Feb 29 11:00
austen pts/5 Feb 29 15:39 (mansfield-park.example.com)
austen pts/6 Feb 29 15:39 (mansfield-park.example.com)

$ who | grep -F austen      Where is austen logged on?
austen pts/5 Feb 29 15:39 (mansfield-park.example.com)
austen pts/6 Feb 29 15:39 (mansfield-park.example.com)
```

Hints for Lab2

- Set locale first!
- Make sure you know what each part of the below command means:

```
tr -cs 'A-Za-z' '[\n*]' | sort -u | comm -23 - words # ENGLISHCHECKER
```

- You will need to write similar commands for HawaiianChecker.
- Use Wget/curl to download the Hawaiian dict page:
  - Mostly sed and tr (maybe sort in the end)
  - Use pipelining in the script to “concatenate” commands.