

## Reading

- Glass and Ables, pp. 84–87
- Glass and Ables, pp. 665–668
- Manual page for `grep`
- [http://en.wikipedia.org/wiki/Regular\\_expression](http://en.wikipedia.org/wiki/Regular_expression)  
(Has a nice summary of POSIX character classes)

## Background

The `grep` utility is used to search for strings in a text file. It allows a rich collection of string patterns to be described with a *regular expression*, an idea with origins in theoretical computer science. For this assignment, we will use *extended regular expressions* (denoted with `grep`'s `-E` option switch) throughout.

Here are a few examples of how `grep` might be used to search for patterns within the text file named `sample`:

Command	Effect
<code>grep -E 'red' sample</code>	Displays all lines in <code>sample</code> with substring <code>red</code>
<code>grep -E '^red' sample</code>	Displays all lines in <code>sample</code> which begin with <code>red</code>
<code>grep -E 'red\$' sample</code>	Displays all lines in <code>sample</code> which end with <code>red</code>
<code>grep -E '^red\$' sample</code>	Displays all lines in <code>sample</code> with <code>red</code> (and nothing else)

Although the `grep` utility had its origins in Unix, today it exists on many different systems.

**Exercise 0.** Create a `hw02` folder and place a copy of `hw02.tex` in it. Other files you create for this assignment should be placed in this folder, since you will be submitting the entire folder.

**Exercise 1.** Using a text editor of your choice (`vi`, `nano`, `Aquamacs`, etc.), create a file named `sample` which will allow you to experiment with the examples given above. Your file should have at least 10 lines in it and the substring `red` should appear multiple times. Describe, in words, the effect of the following `grep` options. (Refer to the manual page for `grep` as needed.)

```
grep -Ec 'red' sample
grep -En 'red' sample
grep -Ev 'red' sample
```

Experiment with the other regular expressions given in the Background section above; make sure you agree with the results.

*Solution.*

**Exercise 2.** In Unix, a *filter* is a program that takes its input from the **standard input** and produces its output on the **standard output**. Is `grep` capable of acting as a filter? To find out, try this command:

```
grep -E 'red'
```

Summarize your findings.

*Solution.*

**Exercise 3.** Unix programs produce an **exit status**, a value between 0 and 255. (A value of 0 typically indicates success.) When interacting with the shell, the exit status of the last program executed can be displayed with the command:

```
echo $?
```

Try these commands, then summarize what you learned:

```
echo 'credit' | grep -Eq 'red' ; echo $?
echo 'bread' | grep -Eq 'red' ; echo $?
```

Consult the manual page for `grep` as needed.

*Solution.*

**Exercise 4.** Design extended regular expressions for each of the following patterns. Supply an appropriate command which will output all the lines in the `sample` file which match the following patterns. (In the solution below, copy and paste each `grep` command from a terminal window into a `\verb!!` environment.) Add lines to your `sample` file in order to thoroughly test your commands.

- (a.) Begins with a decimal digit: 0 through 9.
- (b.) Begins with a hexadecimal digit: 0 through 9, a through f, or A through F.
- (c.) Entire line is a three-digit, decimal value.
- (d.) Entire line consists of hexadecimal digits.
- (e.) Entire line consists of alphabetic characters, either lower- or upper-case.
- (f.) Line contains a phone number of the form (217)<sub>U</sub>xxx-xxxx.
- (g.) Line contains a phone number of the form (312)<sub>U</sub>xxx-xxxx or (708)<sub>U</sub>xxx-xxxx.
- (h.) Line has at least one period.
- (i.) Line has a human-readable IP address<sup>1</sup>.
- (j.) Line includes a quoted string; i.e., text enclosed within double quotes.
- (k.) Line includes a dollar amount with dollars and cents, such as \$123.46. There must be at least one digit for the dollar amount and exactly two digits for the number of cents.
- (l.) Line is longer than 10 characters.
- (m.) Line is shorter than 10 characters.

---

<sup>1</sup>An IP (Internet Protocol) address is a 32-bit quantity, subdivided into four 8-bit quantities. Viewed as unsigned integers, each of these four values is a number between 0 and 255. For example, a typical IP address is 74.125.224.72, which happens to belong to Google. For the purpose of this exercise, you can use a poor man's substitute: look for a pattern of the form  $a.b.c.d$ , where  $0 \leq a, b, c, d \leq 999$ .

*Solution.*

(a.) your `grep` command

**Exercise 5.** Create a shell script, named `ex5`, based on your answers to the previous exercise. The contents of this script should have the following format:

```
#!/bin/sh
echo 'Results produced by <your name here>'

echo '(a)'
your grep command for part (a)
echo

echo '(b)'
your grep command for part (b)
echo

... follow this pattern for all remaining parts ...

exit 0
```

Each part from Exercise 4 contributes three lines to this script — an `echo` announcing the part, the `grep` command used, and an `echo` to produce a blank line. Make this script executable, then run it to see the results of all parts of Exercise 4.

**Exercise 6.** The `grep` utility is a useful tool with many serious applications, but that doesn't mean we can't also use it for recreational wordplay! Determine appropriate `grep` commands which will search `/usr/share/dict/words` for words that meet the following requirements. (In the solution below, copy and paste each `grep` command from a terminal window into a `\verb!!` environment.)

- (a.) All 7-letter words of the form `b_L_L_L_da_L`. (For crossword puzzle help: clue was “two-legged.”)
- (b.) All words, exclusively lower-case, in which `i` immediately follows `q`. (Scrabble, anyone?)
- (c.) All words which begin with a capital letter, in which `i` immediately follows `q`. (Just curious.)
- (d.) All words with either 22 or 23 letters. (Impress your friends with your vocabulary!)
- (e.) All words which have all five vowels (`a`, `e`, `i`, `o`, and `u`)—in that order, interspersed with other non-vowels. (In case your friends weren't impressed before.)

*Solution.*

(a.) your `grep` command

**Exercise 7.** Create a shell script, named `ex7`, based on your answers to the previous exercise. Follow a similar format for the script file you wrote for Exercise 5. Running this script will produce all results from Exercise 6. Each part should be labeled and a blank line should separate each part.

## What to Submit

Drag your entire `hw02` folder onto the EIU submit icon. This folder should include (at least) the following files:

```
hw02.pdf    ex5    ex7    sample
```