Regular expressions, command grep.

Jan Trdlička

Department of Computer Systems
Faculty of Information Technology
Czech Technical University in Prague

trdlicka@fit.cvut.cz

November 16, 2017

Contents

Discussion of homework

- Create alias `lss`, which prints names of files in the working directory sorted by file size.
  - ```
    alias lss='ls -al . | tail -n +2 | sort -k5,5n | \
    tr -s " " | cut -d" " -f9 '
    ```
- Create shell script that prints names of the 10 largest files (including their sizes), which are in your home directory and in its direct subdirectories.
    ```

Discussion of homework

```
#! /bin/bash

LIST=""
cd "$HOME"

for i in * .*
do
  if [ -f "$i" ] ; then
    LIST="$(printf "%s\n%s\n" "$LIST" "$(ls -l "$i")")"
  elif [ -d "$i" -a ! \( "$i" = "." -o "$i" = ".." \) ] ; then
    for j in "$i"/* "$i"/.*
    do
      if [ -f "$j" ] ; then
        LIST="$(printf "%s\n%s\n" "$LIST" "$(ls -l "$j")")"
      fi
    done
  fi
done

echo "$LIST"

echo "$LIST" | tail -n +2 | sort -k5,5nr | head -10 | \
     tr -s ' '| cut -d' ' -f5,9-
```

Discussion of homework

- Patching of source code
  - Create directory with at least 5 files. (e.g. C project: files *.c, *.h, Makefile, ...)
  - Fill the files with some text (e.g. output of command `ls`, `date`, `man`, ...).
  - Create copy of this directory.
  - Modify some files in new directory. Remove/create some old/new one.
  - Use command `diff` to compare both directories.
  - Save the output of command `diff` to the file.
  - Use the previous file in program `patch` to create from old directory structure new directory structure.

Discussion of homework

```
#! /bin/bash

rm -r d1 d2 d1d2.patch

# Create directory d1 with some content
mkdir d1
cp /etc/passwd /etc/group /etc/hosts d1

# Create modified copy of d1
cp -r d1 d2
rm d2/passwd
echo "XXXXXX" >> d2/hosts
#cp -r /etc/protocols d2

# Verify the content of d1 and d2
ls -lR d1 d2

# Upgrade the content of d1 to d2
diff -urNp d1/ d2/ > d1d2.patch
patch -p0 -i d1d2.patch

# Verify the content of d1 and d2
ls -lR d1 d2
```

Grep – options

- What is the default behaviour of the command grep?
  - Print lines matching a pattern.

```
ls /home/* 2>/dev/null | grep novak
```

- What is the meaning of the following options?
  - -v ... invert the sense of matching, to select non-matching lines,

```
echo $PATH | tr ':' '\n' | grep -v bin
```

  - -c ... print a count of matching lines for each input file,

```
grep -c root /etc/passwd
```

  - -i ... ignore case distinctions,

```
man ls | grep -i command
```

Grep – options

- What is the meaning of the following options?
  - -l ... suppress normal output, instead print the name of each input file from which output would normally have been printed,

```
grep -l start /etc/* 2>/dev/null
```

  - -n ... prefix each line of output with the 1-based line number within its input file

```
grep -n bash /usr/bin/* 2>/dev/null
```

grep – regular expression

- What is the meaning of the following symbols in RE?
  - `^` (caret/hat) ... begin of line,
  - `$` ... end of line,
  - `\<` ... begin of word,
  - `\>` ... end of word,

  - `.` ... any single character
  - `[a-d]` or `[abcd]` ... single character from interval/list,
  - `[^a-d]` or `[^abcd]` ... single character not from interval/list,

  - `*` ... the preceding item will be matched zero or more times,
  - `\{n\}` ... the preceding item is matched exactly `n` times,
  - `\{n,\}` ... the preceding item is matched `n` or more times,
  - `\{,n\}` ... the preceding item is matched at most `n` times (GNU extension),
  - `\{m,n\}` ... the preceding item is matched at least `m` times, but not more than `n` times.

grep – regular expression

- What is the meaning of the following symbols in RE?
  - `\( \)` ... a RE enclosed between the character sequences `\(` and `\)` is a RE that matches whatever the unadorned RE matches.
  - `\n` ... matches the same string of characters as was matched by an expression enclosed between `\(` and `\)` earlier in the same RE.

    For example, the expression `^\(.*\)\1$` matches a line consisting of two repeated appearances of the same string.

grep – regular expression

- Try the following commands and explain the output.

```
cd /usr/dict      # on server fray1.fit.cvut.cz
```

```
grep o words
```

```
grep oo words
```

```
grep oo$ words
```

```
grep ^oo words
```

```
grep o.o words
```

```
grep 'o.*o' words
```

```
grep '^o.*o$' words
```

grep – regular expression

- What is the difference between the following commands?

```
grep 'o*' words
```

```
grep 'oo*' words
```

```
grep '[^o]' words
```

```
grep -v o words
```

```
grep '^a.*a$' words
```

```
grep '^\(a\).*\1$' words
```

```
grep '^\(.\).*\1$' words
```

grep – regular expression

- Use the output of the command `ps -ef`. How to print the number of processes running under the identity of the user `root`?
  Hint: try to find the correct solution for both Linux and Solaris.

```
ps -ef | grep -c '^root '      # Linux

ps -ef | grep -c '^ *root '    # Linux + Solaris
```

- Use the output of the command `ps -eo pid,user,comm` on the server `fray1.fit.cvut.cz`. How to determine how many times the program `sshd` is running on this server?

```
ps -eo user,pid,comm | grep -c '/sshd$'
```

grep – table

- Use the output of the command `ps -eo pid,user,comm` on the server `fray1.fit.cvut.cz`. How to determine how many times the program `sshd` is running under the identity of the user `root` on this server?

```
ps -eo pid,user,comm | grep -c ' root *[^ ]*/sshd$'
```

- Use the output of the command `getent passwd` on the server `fray1.fit.cvut.cz`. How to determine how many students have the account on this server?
  Hint: student's account has the flag `student` at the end of the fifth column.

```
getent passwd | grep -c 'student:[^:]*:[^:]*$'
```

grep – file

- Assume that the variable `file` is defined as follows

```
file=/home/courses/BIPS1/public/07/NAMES.TXT
```

Use the output of the command `getent passwd` on the server `fray1.fit.cvut.cz`. How to find account info about users whose names are in the file `NAMES.TXT`?

```
getent passwd | /usr/xpg4/bin/grep -f $file
```

egrep/grep – extended regular expression (ERE)

- What regular expressions are not accepted by command `egrep`?
  - `\<` and `\>`, `\{` and `\}`, `\(` and `\)` , `\n`
- What is the meaning of the following symbols in RE accepeted by `egrep`?
  - `+` ... match one or more occurrences of the previous regular expression,
  - `?` ... match zero or one occurrences of the previous regular expression,
  - `|` ... full regular expressions separated by `|` or by a NEWLINE that match strings that are matched by any of the expressions.
  - `( )` ... a full regular expression that can be enclosed in parentheses `( )` for grouping.

egrep/grep – extended regular expression (ERE)

- How to print names of the days of the week from file `/usr/dict/words` on the server `fray1.fit.cvut.cz`?

```
M o n      day
T u e s    day
W e d nes  day
T h u rs   day
F r i      day
S a t ur   day
S u n      day
```

```
grep '[MTWFS][ouehra][neduit][esnru]*day' words
```

```
egrep '(Mon|Tues|Wednes|Thurs|Fri|Satur|Sun)day' words
```

Homework

- How to print lines containing the same character at the beginning and at the end of line.

  Hint: use `man -s 5 regexp` and find meaning of symbols `\n`, `\(` and `\)` in the section 2.6 on server `fray1.fit.cvut.cz`.

- How to print all palindromes of length 2, 3, 4 and 5 characters from the file `/usr/dict/words` on server `fray1.fit.cvut.cz`.