Linux/UNIX

grep

Student learning objectives and outcomes, at the successful conclusion the student will be able to:

- Login into a UNIX/Linux based web-server remotely
- Explain and perform basic file permission commands
- Perform directory commands
- Use grep to search for a pattern and display lines

UNIX has a special family of command for handling search requirements, and the principal member of this family is the **grep** command. The command grep scans its input for a pattern and can display the selected pattern, the line numbers or the filenames where the pattern occurs.

Syntax:

grep [options] pattern [files]

Options Description

-c : This prints only a count of the lines that match a pattern

-h : Display the matched lines, but do not display the filenames.

-i : Ignores, case for matching

-l : Displays list of a filenames only.

-n : Display the matched lines and their line numbers.

-v : This prints out all the lines that do not matches the pattern

-e exp : Specifies expression with this option. Can use multiple times.

-f file : Takes patterns from file, one per line.

-E : Treats pattern as an extended regular expression (ERE)

-w : Match whole word

-o : Print only the matched parts of a matching line,
with each such part on a separate output line.

Assignment:

Copy the file emp.list from the /tmp to your homework directory.
You will be referring to this file for this set of exercises.

This is a fixed-format text file containing 15 lines of a personnel database. There are six colon-delimited fields –emp-id, name, designation, department, date of birth and salary.

Using the emp.list

Type the following commands and answer the questions (record the output and explain what the command is doing):

1. **grep "sales" emp.list**

   What is the command doing?

   This command displays all lines from the file "emp.list" that contain the pattern "sales".

   What is the output?

   ```
   2233:charles harris  :g.m.     :sales     :12/12/52: 90000
   1006:gordon lightfoot:director :sales     :09/03/38:140000
   1265:p.j. woodhouse  :manager  :sales     :09/12/63: 90000
   2476:jackie wodehouse:manager  :sales     :05/01/59:110000
   ```

2. **grep "charles" emp.list > foo**
   cat foo

   What is the command doing?

   This command puts all lines from the file "emp.list" that contain the pattern "charles" into the file "foo" and then displays the contents of "foo".

   What is the output?

   ```
   2233:charles harris  :g.m.     :sales     :12/12/52: 90000
   ```

3. **grep "director" emp.list emp.list2**

   What is the command doing?

   This command displays all lines from either "emp.list" or "emp.list2" that contain the pattern "director". Because "emp.list2" does not exist, it displayed an error message.

   What is the output?

   ```
   emp.list:9876:bill johnson    :director :production:03/12/50:130000
   emp.list:2365:john woodcock   :director :personnel :05/11/47:120000
   emp.list:1006:gordon lightfoot:director :sales     :09/03/38:140000
   emp.list:6521:derryk o'brien  :director :marketing :09/26/45:125000
   grep: emp.list2: No such file or directory
   ```

4. **grep gordon lightfoot emp.list**

   What is the command doing?

   This command displays all lines from either the file "lightfoot" or the file "emp.list" that contain the pattern "gordon". grep only accepts one pattern, and because there was a space between "gordon" and "lightfoot", "lightfoot" was considered to be a directory.

   What is the output?

   ```
   grep: lightfoot: No such file or directory
   emp.list:1006:gordon lightfoot:director :sales     :09/03/38:140000
   ```

5. **grep ‘gordon light’ emp.list**

   What is the command doing?

   This command displays all lines from the file “emp.list” that contain the pattern “gordon light”.

   What is the output?

```
1006:gordon lightfoot:director :sales     :09/03/38:140000
```

   What is the difference between this command and the command above it?

   The most important difference between this command and the previous one is that this command uses quotes around the intended pattern while the previous one does not. This let grep correctly identify the intended pattern and file.

6. **grep “marketing” emp.list**

   What is the command doing?

   This command displays all lines from the file “emp.list” that contain the pattern “marketing”.

   What is the output?

```
5678:robert dylan    :d.g.m.   :marketing :04/19/43: 85000
6521:derryk o'brien  :director :marketing :09/26/45:125000
2345:james wilcox    :g.m.     :marketing :03/12/45:110000
0110:julie truman    :g.m.     :marketing :12/31/40: 95000
```

7. **grep president emp.list**

   grep can also behave silently. It simply returns the prompt when the pattern can’t be located.

   What is the command doing?

   This command displays all lines from the file “emp.list” that contain the pattern “president”.

   What is the output?

   There was no output.

8. **grep “wo[od][de]house” emp.list**

   What is the command doing?

   his command displays all lines from the file “emp.list” that contain a pattern that goes “wo”, either “o” or “d”, either “d” or “e”, “house”.

   What is the output?

```
1265:p.j. woodhouse  :manager  :sales     :09/12/63: 90000
2476:jackie wodehouse:manager  :sales     :05/01/59:110000
```

   A regular expression uses an elaborate metacharaacter set that overshadows the shell’s wild cards (*). grep uses this expression to match multiple similar patterns. Unlike wild cards, however, a regular expression is a feature of the command that uses it and has nothing to do with the shell. Regular expressions take care of some common query and substitution requirements. **Regular expressions are interpreted by the command and not by the shell. Quoting ensures that the shell isn’t able to interfere and interpret the metacharacters in its own way.

9. **grep *yourusername* /etc/passwd**

   What is the command doing?

   This command displays all lines from the file “/etc/passwd” that contain the pattern “tkarmesin

   What is the output?

   ```
   tkarmesin:x:1061:1061::/home/tkarmesin:/bin/bash
   ```

10. **Research most important grep commands. List 3 and explain the commands.**

    a. grep “pattern” filename
       i. This is the most basic and common use of the grep command.
       ii. It will display all lines of the “filename” file that contain “pattern”.

    b. grep -i “pattern” filename
       i. This command is the same as above, except it will ignore case.
       ii. For example, this would display a line that contained “PaTtErN”, whereas the previous command would not.

    c. grep -v “pattern” filename
       i. This command is the inverse of basic grep. Rather than display every line that contains “pattern”, it will display every line that doesn’t.

Change this file to a pdf. Upload to the server and submit the URL.