Capgemini

UNIX

Lesson 10 : Pattern Matching Using grep,egrep,fgrep

Lesson Objectives

- In this lesson, you will learn:
  - Regular expression

10.1: grep
grep Command

The syntax for grep command is as follows:

```
grep <options> <pattern> <filename(s)>
```

- **Example:** The following example will search for the string Unix in the file **books.lst**. The lines which match the pattern will be displayed.

```
grep 'Unix' books.lst
```

grep Command

Options of grep:

- c : It displays count of lines which match the pattern.
- n : It displays lines with the number of the line in the text file which match the pattern.
- v : It displays all lines which do not match pattern.
- i : It ignores case while matching pattern.
- -w : It forces grep to select only those lines containing matches that form whole words

grep Command

- **Example 1:** To print all lines containing “rose” regardless of case:

```
$grep -i rose flower.txt
```

- **Example 2:** To print all lines containing “rose” as a word:

```
$grep -w rose flower.txt
```

- **Example 3:** To print all lines not containing “rose”:

```
$grep -v rose flower.txt
```

grep Command

- Regular Expression:

Expression	Description
^ (Caret)	match expression at the start of a line, as in ^A.
$ (Question)	match expression at the end of a line, as in A$.
\ (Back Slash)	turn off the special meaning of the next character, as in \^.
[ ] (Brackets)	match any one of the enclosed characters, as in [aeiou]. Use Hyphen "-" for a range, as in [0-9].
[^]	match any one character except those enclosed in [ ], as in [^0-9].
. (Period)	match a single character of any value, except end of line.
* (Asterisk)	match zero or more of the preceding character or expression.
\{x,y\}	match x to y occurrences of the preceding.
\{x\}	match exactly x occurrences of the preceding.
\{x,\}	match x or more occurrences of the preceding.

grep Command

- Examples of Regular Expression:

Example	Description
grep “smile“ files	**search *files* for lines with ‘smile’**
grep '^smile' files	**'smile' at the start of a line**
grep 'smile$' files	**'smile' at the end of a line**
grep '^smile$' files	**lines containing only 'smile'**
grep '\^s' files	**lines starting with '^s', "\" escapes the ^**
grep '[Ss]mile' files	**search for ‘Smile' or 'smile'**
grep 'B[oO][bB]' files	**search for BOB, Bob, BOb or BoB**
grep '^$' files	**search for blank lines**
grep '[0-9][0-9]' file	**search for pairs of numeric digits**

10.2: fgrep
fgrep Command

The fgrep command is similar to grep command.

Syntax:

```
$fgrep  [ -e pattern_list] [-f pattern-file] [pattern] [Search file]
```

The fgrep command is useful to search files for one or more patterns, which cannot be combined together.

It does not use regular expressions. Instead, it does direct string comparison to find matching lines of text in the input.

fgrep Command

Options of fgrep command:

- -e pattern_list :
  - It searches for a string in pattern-list.
- -f pattern-file :
  - It takes the list of patterns from pattern-file.
- pattern
  - It specifies a pattern to be used during the search for input.
  - It is same as grep command.
- E.g To search employee file for all patterns stored in mypattern file

  $ fgrep –f mypattern employee.lst

10.3: egrep
egrep Command

The egrep command works in a similar way. However, it uses extended regular expression matching.

- Syntax:

```
egrep [ -e pattern_list ] [-f file ] [ strings ] [ file]
```

- **Example:** To find all lines with name “aggrawal” even though it is spelled differently:

```
$ egrep '[aA]gg?[ar]+wal' stud.lst
```

SUMMARY

- In this lesson, you have learnt:
  - Basic and extended regular expression
  - Grep family

Review Questions

- Question 1: With _____ we can not use regular expression.
- Question 2: Which are the extended regular expression?