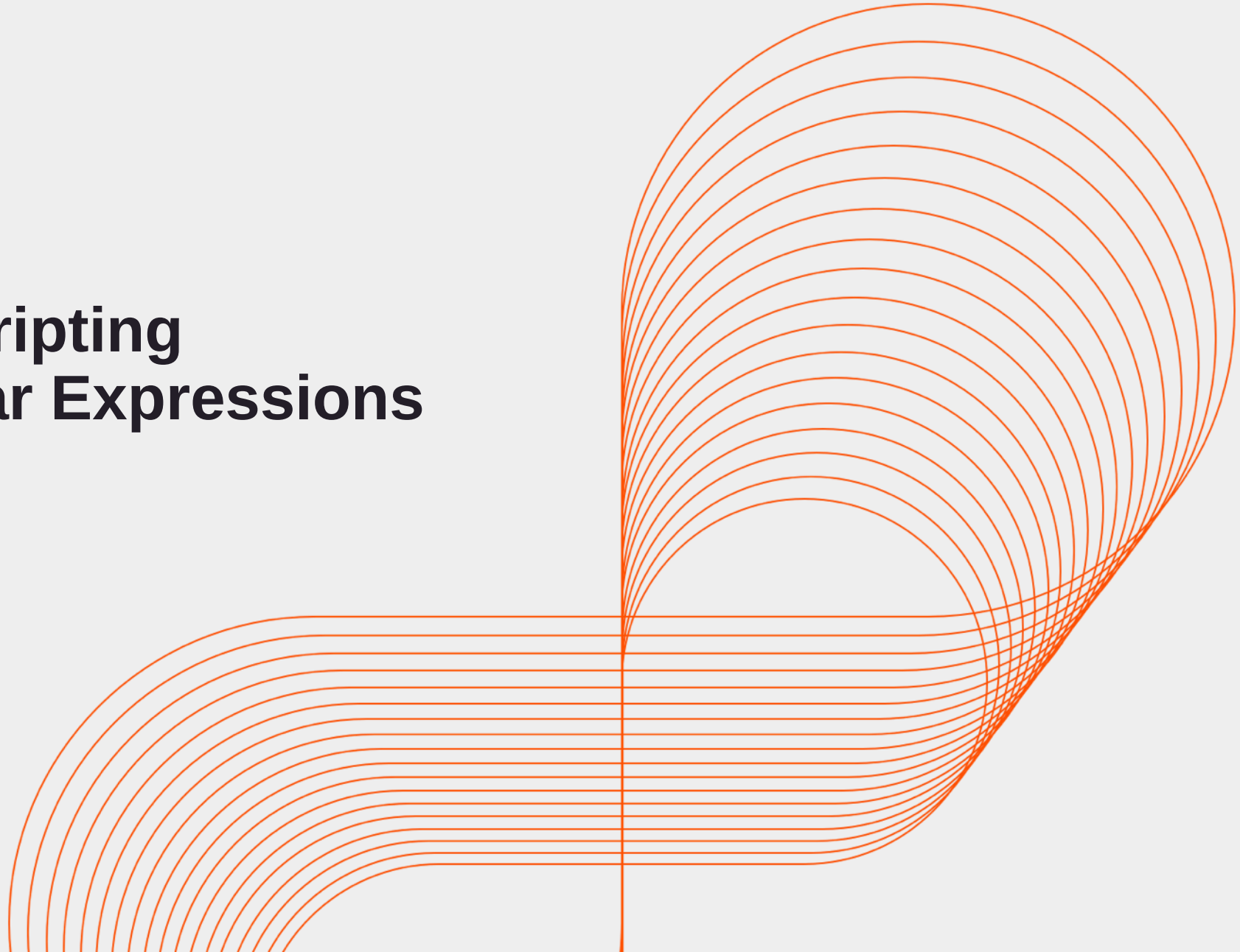
Persistent

Bash Shell Scripting
Grep & Regular Expressions

Persistent University



Contents

1. GREP and Regular expressions
2. What is GREP?
3. Syntax of GREP command.
4. GREP options.
5. Regular Expression.
6. Extended Regular Expression.

What is grep?

- Grep (Global regular expression print)
  - The grep command lets you locate the lines in a file that contain a particular word or a phrase.
  - The command is derived from a feature of the original UNIX text editor, “ed”.
  - It has three variant programs : egrep, fgrep, rgrep.
- Syntax:
  - grep [options] PATTERN [FILE...]
  - grep [options] [-e PATTERN | -f FILE] [FILE...]
- Example:
- **$ cat textfile**
- This is line 1, of which there is only one instance.
- This is the only instance of line 2.
- this is line 3, another line.
- This is line 4.
- **$ grep 'the' textfile**
- This is line 1, of which there is only one instance.
- This is the only instance of line 2.
- This is line 3, another line.

GREP options

- Case insensitive search.

$ grep ‘This’ textfile

This is line 1, of which there is only one instance.
This is the only instance of line 2.
This is line 4.

$ grep –i ‘This’ textfile

This is line 1, of which there is only one instance.
This is the only instance of line 2.
this is line 3, another line.
This is line 4.

Print line number.

- **$ grep -n 'This' textfile**
- 1:This is line 1, of which there is only one instance.
- 2:This is the only instance of line 2.
- 4:This is line 4.

GREP options

- Invert match
  - **$ grep -vn 'This' textfile**
  - 3:this is line 3, another line.
- Count number of lines.
  - **$ grep -c 'This' textfile**
  - 3
- Print filenames.
  - **$ grep -l 'This‘ ***
  - func_test.sh
  - Textfile
- Exact match line
  - **$ grep -x ‘This' textfile**

GREP options

- Whole word match:
- **$cat wfile**
- boot
- boot1
- boots
- books
- **$ grep 'boot' wfile**
- boot
- boot1
- boots
- **$ grep -w 'boot‘ wfile**
- boot

GREP options (Cntd.)

- Recursive search:
- **$ grep 'boot‘ ***
- wfile:boot
- wfile:boot1
- wfile:boots
- **$ grep -r 'boot' ***
- wfile:boot
- wfile:boot1
- wfile:boots
- test/sfile:boot
- test/sfile:boot1
- test/sfile:boots

GREP options (Cntd.)

- Print trailing lines after matching line.
- **$ grep 'boots‘ wfile**
- boots
- **$grep -A1 'boots' wfile**
- boots
- books

Regular Expressions

- Regular Expression:
  - set of characters (metacharacters) that specify a pattern.
- Structure of regular expression:
  - **Anchors** : Specifies position of pattern.
  - **Quantifiers** : Specifies number of symbols that could appear in string.
  - **Character classes** : Match one or more characters in a single position.

Anchors

- ^ - Matches the beginning of a line or string
- **$ cat textfile**
- This is line 1, of which there is instance one
- This is the only instance of line 2
- this is line 3, another line
- This is 4th line
  - **$ grep '^th' textfile**
  - this is line 3, another line
- $ - Matches the end of line or string
- **$ grep 'ne$' textfile**
- This is line 1, of which there is instance one
- this is line 3, another line
- This is 4th line
- **$ grep ‘ine$' textfile**
- this is line 3, another line
- This is 4th line

Anchors

- \<, \> - Matches beginning, end of string
- **$ grep 'the' textfile**
- This is line 1, of which there is instance one
- This is the only instance of line 2
- this is line 3, another line
- **$ grep ‘\<the\>' textfile**
- This is the only instance of line 2

- \b – Matches beginning or end of string
- **$ grep '\bthe' textfile**
- This is line 1, of which there is instance one
- This is the only instance of line 2
- **$ grep 'the\b' textfile**
- This is the only instance of line 2

- \B – Matches beginning or end of string
- **$grep 'her\B' textfile**
- This is line 1, of which there is instance one

Quantifiers

- Match any single character
- **$ cat afile**
- abc
- abbc
- ac
- abbbc

grep 'ab.c' afile

abbc

Quantifiers (Cntd.)

- Match zero or one of preceding element
- **$ grep 'ab\?c' afile**
- Abc
- Ac
- Match zero or more of preceding element.
- **$ grep 'ab*c' afile**
- abc
- abbc
- ac
- abbbc

Quantifiers

- Match one or more of preceding element.
  - **$ grep 'ab\+c' afile**
  - abc
  - abbc
  - Abbbc
- Match preceding element n times.
  - **$ grep 'ab\{3\}c' afile**
  - abbbc
  - **$ grep 'ab\{2,\}c' afile**
  - abbc
  - Abbbc
  - **$ grep 'ab\{0,2\}c' afile**
  - abc
  - abbc
  - Ac

Normal Pattern Language

- **Lets understand normal pattern language–**
  - **Sequence** **Description**
  - ***** Matches **any string**, including the null string (empty string)
  - **?** Matches any **single character**
  - **{n}** The preceding item is matched exactly n times
  - **{n,m}** The preceding item is matched at least n times, but not more than m times.
  - \\ Matches a backslash
  - […] Defines a pattern **bracket expression** (see below). Matches any of the enclosed characters at this position

Extended Pattern Language

- Lets understand extended pattern language –

Sequence	Description
?(<PATTERN-LIST>)	Matches zero or one occurrence of the given patterns
*(<PATTERN-LIST>)	Matches zero or more occurrences of the given patterns
+(<PATTERN-LIST>)	Matches one or more occurrences of the given patterns
@(<PATTERN-LIST>)	Matches one of the given patterns
!(<PATTERN-LIST>)	Matches anything except one of the given patterns

Character classes

- A bracket expression:
  - [A-Z] : Uppercase A to Z characters.
  - [abc] : “a”, “b” or “c”.
  - [A-Za-z0-9] : Alphanumeric characters.
  - [0-9] : 0 to 9 digits.
  - [^abc] : character other than “a”, ”b” or “c”.
  - [^a-z] : character that is not in lowercase.
- Escape characters:
  - \.* : dot character any number of times.
  - \[.\] : any single character surrounded by "[" and "]“.
- Alteration:
  - abc|def : matches “abc” or “def”.
  - cat|dog : matches “cat” or “dog”.

Extended regular expression

- Additional metacharacters added to basic set.
- Characters ?, +, { }, ( ), |
  - **?** : matches zero or one of previous RE
  - **+** : matches one or more previous RE
  - **{ }** : number of occurrences of preceding RE
  - **( )** : encloses group of REs
  - **|** : matches alternate characters
- Examples:
  - **$ cat myfile.txt**
  - People who read seem to be better informed than those who do not.
  - Red is the color of extremes. Yellow-based reds are "tomato reds".
  - The clarinet produces sound by the vibration of its reed.
  - **$ egrep 're(a|e)d' myfile.txt**
  - People who read seem to be better informed than those who do not.
  - The clarinet produces sound by the vibration of its reed.

regex examples

- **Print all lines with exactly two characters in file.**
  - grep '^..$' filename
- **Display any lines starting with a dot and digit:.**
  - grep '^\.[0-9]' filename
- **How to do OR with grep?**
  - grep -E 'word1|word2' filename
  - OR
  - egrep 'word1|word2' filename
  - OR
  - grep 'word1\|word2' filename

egrep examples

- User can test how often a character must be repeated in sequence using below syntax:
  - {N}
  - {N,}
  - {N,M}
- **Match a character "v" two times:**
  - egrep "v{2}" filename
- **The following will match both "col" and "cool":**
  - egrep 'co{1,2}l' filename
- **The following will match any row of at least three letters 'c'.**
  - egrep 'c{3,}' filename

Quiz

- The word *grep* stands for _____.
- The option ____ used to print only the filenames of files that have lines that match the search string.
- There are three important parts in the structure of regular expression : _____.
- The character ___ matches zero or one occurrence of the previous character.
- If the first character of the list is the ____, then it matches any character not in the list.

Quiz Solution

- The word grep stands for Globally Regular Expression Print.
- The option -l used to print only the filenames of files that have lines that match the search string.
- There are three important parts in the structure of regular expression : Anchors, Quantifiers and Character classes.
- The character ? matches zero or one occurrence of the previous character.
- If the first character of the list is the caret(^), then it matches any character not in the list.

Summary

- In this module we have learnt about Grep and Regular Expressions.
- Now, you should be able to answer following questions:
  - What is grep?.
  - The syntax of grep command.
  - How to use grep options?.
  - What is Regular Expression and Extended Regular Expression?.

Reference material

- http://www.computerhope.com/unix/ugrep.htm
- http://www.linuxtopia.org/online_books/advanced_bash_scripting_guide/x13357.html
- https://en.wikipedia.org/wiki/Regular_expression

Key contacts

- **Persistent Interactive**

  Bhimashankar Gavkare

  bhimashankar_gavkare@persistent.com

Persistent

Persistent

Thank You !!!

Persistent University