

Finding strings in text files using grep with regular expression- 10 Examples

By coding-bootcamps.com

Grep is one of most popular tools for searching and finding strings in a text file. The name 'grep' derives from a command in the now-obsolete Unix `ed` line editor tool—the `ed` command for searching *globally* through a file for a *regular expression* and then *printing* those lines was *g/re/p*, where *re* was the regular expression you would use. Eventually, the `grep` command was written to do this search on a file when not using `ed`.

In this tutorial, we show you to run advance string searching using Grep with regular expression by giving you 10 hands-on examples on its implementations. Many examples discussed in this tutorial have practical implications meaning you can use them in your daily Linux programming. The following samples describe some regex examples for commonly searched-for patterns.

Here is the outline of 10 grep examples covered in this tutorial:

Ex 1: Find a Single Character in a Text File

Ex 2: Find a Single string in a Text File

Ex 3: Find a Single Special Character in a Text File

Ex 4: Matching Lines Beginning with Certain Text

Ex 5: Matching Lines Ending with Certain Text

Ex 6: Matching Lines of a Certain Length

Ex 7: Matching Lines That Contain Any of Some Regexps

Ex 8: Matching Lines That Contain All of Some Regexps

Ex 9: Matching Lines That Only Contain Certain Characters

Ex 10: Finding Phrases Regardless of Spacing

Ex 1: Find a Single Character in a Text File

To output lines in the file 'book' that contain a '\$' character, type:

```
$ grep '\$' book
```

Ex 2: Find a Single string in a Text File

To output lines in the file 'book' that contains the string '\$14.99', type:

```
$ grep '\$14\.99' book
```

Ex 3: Find a Single Special Character in a Text File

To output lines in the file 'book' that contain a '\' character, type:

```
$ grep '\\\' book
```

Ex 4: Matching Lines Beginning with Certain Text

Use '^' in a regexp to denote the beginning of a line.

To output all lines in 'book' beginning with 'pro', type:

```
$ grep '^pro' book
```

To output all lines in the file 'book' that begin with the text 'in the beginning', regardless of case, type:

```
$ grep -i '^in the beginning' book
```

NOTE: These regexps were quoted with ' characters; this is because some shells otherwise treat the '^' character as a special "metacharacter"

In addition to word and phrase searches, you can use `grep` to search for complex text patterns called regular expressions. A regular expression—or "regexp"—is a text string of special characters that specifies a *set* of patterns to match.

Technically speaking, the word or phrase patterns are regular expressions—just very simple ones. In a regular expression, most characters—including letters and numbers—represent themselves. For example, the regexp pattern `1` matches the string '1', and the pattern `boy` matches the string 'boy'.

There are a number of reserved characters called metacharacters that do not represent themselves in a regular expression, but they have a special meaning that is used to build complex patterns. These metacharacters are as follows: ., *, [,], ^, \$, and \. It is good to note that such metacharacters are common among almost all of [common](#) and [special](#) Linux distributions. [Here](#) is a good article that covers special meanings of the metacharacters and gives examples of their usage.

Ex 5: Matching Lines Ending with Certain Text

Use '\$' as the last character of quoted text to match that text only at the end of a line.

To output lines in the file 'book' ending with an exclamation point, type:

```
$ grep '!$' book
```

Ex 6: Matching Lines of a Certain Length

To match lines of a particular length, use that number of '.' characters between '^' and '\$'—for example, to match all lines that are two characters (or columns) wide, use '^..\$' as the regexp to search for.

To output all lines in 'book' that are exactly three characters wide, type:

```
$ grep '^...$' book
```

For longer lines, it is more useful to use a different construct: '^.{number}\$', where number is the number of lines to match. Use ',' to specify a range of numbers.

To output all lines in 'book' that are exactly twelve characters wide, type:

```
$ grep '^.{12}$' book
```

To output all lines in 'book' that are twenty-two or more characters wide, type:

```
$ grep '^.{22}$' book
```

Ex 7: Matching Lines That Contain Any of Some Regexps

To match lines that contain any of a number of regexps, specify each of the regexps to search for between alternation operators ('|') as the regexp to search for. Lines containing any of the given regexps will be output.

To output all lines in 'book' that contains either the patterns 'the book' or 'cake', type:

```
$ grep 'the book|cake' book
```

Ex 8: Matching Lines That Contain All of Some Regexps

To output lines that match *all* of a number of regexps, use `grep` to output lines containing the first regexp you want to match, and pipe the output to a `grep` with the second regexp as an argument. Continue adding pipes to `grep` searches for all the regexps you want to search for.

To output all lines in 'book' that contains both patterns 'the shore' and 'sky', regardless of case, type:

```
$ grep -i 'the shore' book | grep -i sky
```

Ex 9: Matching Lines That Only Contain Certain Characters

To match lines that only contain certain characters, use the regexp '^ [characters] *\$', where characters are the ones to match.

To output lines in 'book' that only contain vowels, type:

```
$ grep -i '[aeiou]' book
```

The '-i' option matches characters regardless of case; so, in this example, all vowel characters are matched regardless of case.

Ex 10: Finding Phrases Regardless of Spacing

One way to search for a phrase that might occur with extra spaces between words, or across a line or page break, is to remove all linefeeds and extra spaces from the input, and then `grep` that. To do this, pipe the input to `tr` with `'\r\n:\>|-'` as an argument to the `-d` option (removing all line breaks from the input); pipe that to the `fmt` filter with the `-u` option (outputting the text with uniform spacing); and pipe that to `grep` with the pattern to search for.

To search across line breaks for the string 'at the same time as' in the file 'book', type:

```
$ cat book | tr -d '\r\n:\>|- ' | fmt -u | grep 'at the same time as'
```

Summary

In this article, we reviewed 10 practical examples of using Grep Linux command for searching and finding strings in a text file. Along the way, we learned how to use regular expressions in conjunction with Grep to conduct complex searches on text files. By now you have a better idea on how powerful Linux search functions are.

Here are additional resources for those interested in learning more about Linux programming:

Resources for System Administrators

1. [Linux System Admin Guide- What is Linux Operating System and how it works](#)
2. [Linux System Admin Guide- Overview of Linux Virtual Memory and Disk Buffer Cache](#)
3. [Linux System Admin Guide- Best Practices for Monitoring Linux Systems](#)
4. [Linux System Admin Guide- Best Practices for Performing Linux Boots and Shutdowns](#)
5. [Linux System Admin Guide- Best Practices for Making and Managing Backup Operations](#)

Resources for Linux Kernel Programmers

1. [How Linux Operating System Memory Management works](#)
2. [Comprehensive Review of Linux Kernel Operating System Processes](#)
3. [What are mechanisms behind Linux Kernel task management](#)

Linux File System Dictionary

[Comprehensive Review of How Linux File and Directory System Works](#)