

# GREP vs. EGREP vs. FGREP

Search for a variety of text fragments through the most powerful commands of terminal i.e. grep, egrep and fgrep.

## grep

### Definition:

The command `grep` stands for “**global regular expression print**”, and is used to search for specified text patterns in files or program outputs.

### Syntax:

```
grep [option(s)] pattern [file(s)]
```

### Options:

Option	Description
-E (extended regexp)	Causes <code>grep</code> to behave like <code>egrep</code> .
-F (fixed strings)	Causes <code>grep</code> to behave like <code>fgrep</code> .
-G (basic regexp)	Causes <code>grep</code> , <code>egrep</code> , or <code>fgrep</code> to behave like the standard <b>grep</b> utility.
-r	To search recursively through an entire directory tree (i.e., a directory and all levels of subdirectories within it)
-I	Process a binary file as if it did not contain matching data

contain matching data.

-c	To report the number of times that the pattern has been matched for each file and to not display the actual lines.
-n	To precede each line of output with the number of the line in the text file from which it was obtained.
-v	It matches only those lines that do not contain the given pattern.
-w	To select only those lines that contain an entire word or phrase that matches the specified pattern.
-x	To select only those lines that match exactly the specified pattern.
-l	To not return the lines containing matches but to only return only the names of the files that contain matches.
-L	It is the opposite of the -l option (and analogous to the -v option) i.e. it will cause grep to return only the names of files that do not contain the specified pattern.

### Example:

- This would search all files in the current directory and in all of its subdirectories, for every line containing the string “Educative”:

```
grep -r 'Educative' *
```

<div>main.sh</div> <div>test2.txt</div> <div>test1.txt</div>	<div>grep -r 'Educative' *</div> <div>Copy</div>
<div>Play</div> <div>Save</div> <div>Undo</div> <div>Fullscreen</div>	

- To search for a word in some respective files:

```
grep Educative file1 file2 file3
```

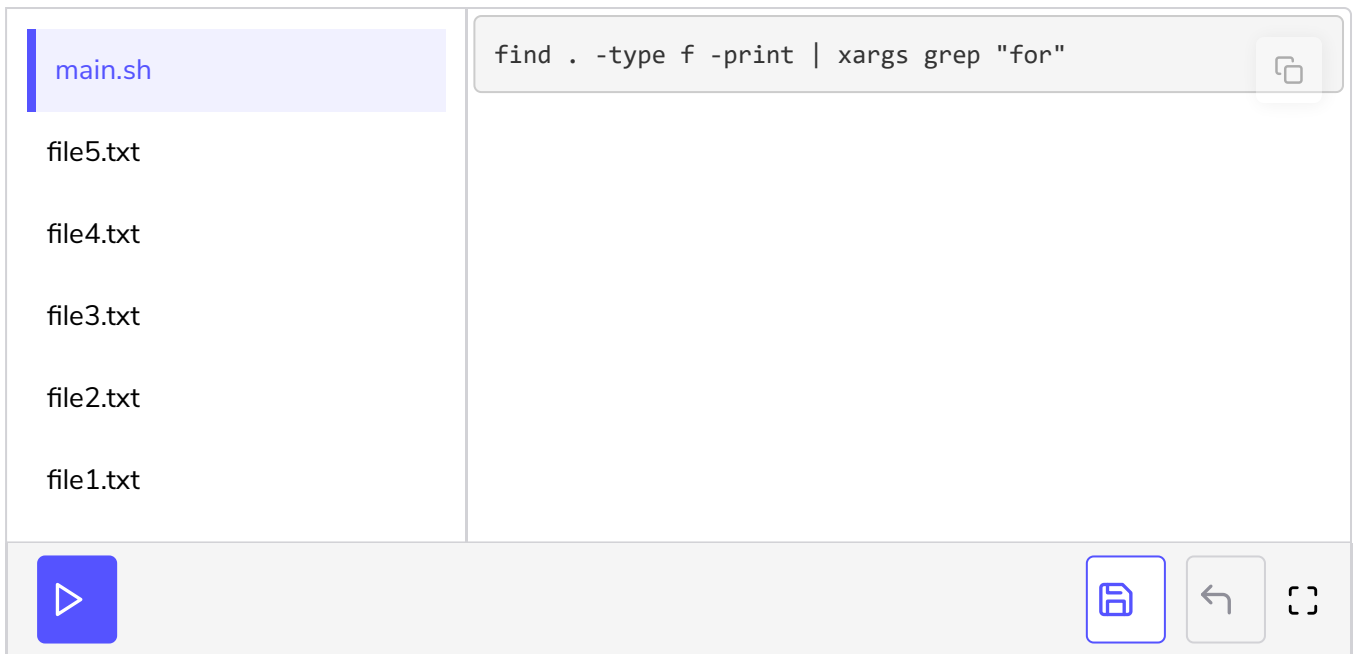
<div>main.sh</div> <div>file3</div> <div>file2</div> <div>file1</div>	<div>Random Text</div> <div>Copy</div>
<div>Play</div> <div>Save</div> <div>Undo</div> <div>Fullscreen</div>	

- `grep` can be used to search for a sequence of strings:

```
grep 'Search commands' file1 file2 file3
```

<div>main.sh</div> <div>file3</div> <div>file2</div> <div>file1</div>	<div>grep 'Not Educative' file1 file2 file3</div> <div>Copy</div>
<div>Play</div> <div>Save</div> <div>Undo</div> <div>Fullscreen</div>	

- Using `grep` to find files based on content:



# egrep

## Definition:

The command `egrep` stands for “**extended global regular expression print**”. It is used for searching particular patterns and is same as `grep` with an `-E` option:

```
grep -E is same as egrep
```

## Syntax:

```
egrep [option(s)] pattern [file(s)]
```

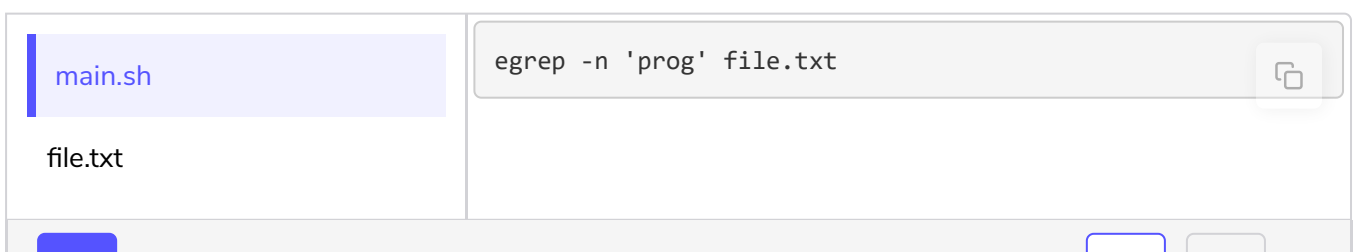
## Options:

`egrep` accepts same options as `grep` except `-E` and `-F`.

## Examples:

- Identifying every line containing a specific string:





To identify every line containing the string “prog” in **file.txt**, execute the following command. Here, `-n` shows the line numbers along with the results:



## Regex Examples





- Finding lines with specific number of vowels:

To find lines with 2 vowels:

<div>main.sh</div> <div>file.txt</div>	<div>egrep '[aeiou]{2,}' file.txt</div> <div></div>
<div></div> <div>  </div>	

- Finding lines with specific characters in them and those characters don't come at the end of line:

To find the lines that have 'in' in them and these lines do not end up on 'in':

<div>main.sh</div> <div>file.txt</div>	<div>programmed system file heirarchy progress intent output program wings insight grin</div> <div></div>
<div></div> <div>  </div>	

- Finding each line with some sequences of characters:

To find the lines having "pro" or "in":

<div>main.sh</div> <div>file.txt</div>	<div>egrep -n 'pro in' file.txt</div> <div></div>
<div></div> <div>  </div>	

- Finding number of lines with some particular character at the end:

To find the number of lines in **file.txt** with the letter ‘t’ in the end:

<pre>main.sh</pre> <pre>file.txt</pre>	<pre>egrep -c 't\$' file.txt</pre> <div></div>
<div> </div>	

- Finding lines beginning with some specific characters:

To find the lines that begin with letters ranging from ‘c’ to ‘i’:

<pre>main.sh</pre> <pre>file.txt</pre>	<pre>egrep '^[a-i]' file.txt</pre> <div></div>
<div> </div>	

## fgrep

### Definition:

**fgrep** is used to interpret pattern as a list of **fixed strings** (the whole string is interpreted literally), separated by new lines, Hence, regular expressions can’t be used.

### Example:

To find the string “*Banana*” along with the line number where it’s found:

<pre>main.sh</pre> <pre>file.txt</pre>	<pre>fgrep -n "Banana" file.txt</pre> <div></div>
<div> </div>	