

Search and Replace

| | |
|---|--|
| <code>:range s/patt ern /st rin g/</code> | For each line in the range replace a match of the pattern with the string . |
| <code>cgI</code> | |
| <code>c</code> | Confirm each substitution |
| <code>g</code> | Replace all occurrences in the line (without <code>g</code> only first occurrence is replaced) |
| <code>i</code> | Ignore case for the pattern |
| <code>I</code> | Don't ignore case for the pattern |

Range of Operation, Line Addressing and Marks

| | |
|--------------------------|--|
| Specifier number | an absolute line number |
| <code>.</code> | the current line |
| <code>\$</code> | the last line in the file |
| <code>%</code> | The whole file. Same as <code>1,\$</code> |
| <code>'t</code> | position of mark "t" |
| <code>/pattern[/]</code> | the next line where text "pattern" matches |
| <code>?pattern[?]</code> | the previous line where text "pattern" matches |
| <code>V</code> | the next line where the previously used search pattern matches |
| <code>\?</code> | the previous line where the previously used search pattern matches |
| <code>\&</code> | the next line where the previously used substitute pattern matches |

Replacement String Options.

| | |
|--------------------|---|
| Backreferences | Allows you to utilize patterns grouped using <code>\ (</code> and <code>\)</code> and refer to them inside the replacement pattern by their order. |
| <code>&</code> | The whole matched pattern |
| <code>\0</code> | The whole matched pattern |
| <code>\1</code> | The matched pattern in the first pair of <code>\ (\)</code> |
| <code>\n</code> | The matched pattern in the n'th pair of <code>\ (\)</code> |
| <code>~</code> | The previous substitute string |
| Actions | Allow you to "act" |
| <code>\L</code> | The following characters are made lowercase. |

Replacement String Options. (cont)

| | |
|-----------------|--|
| <code>\U</code> | The following characters are made uppercase. |
| <code>\E</code> | End of <code>\U</code> and <code>\L</code> |
| <code>\e</code> | End of <code>\U</code> and <code>\L</code> |
| <code>\l</code> | Next character is made lowercase. |
| <code>\u</code> | Next character is made uppercase. |
| <code>\r</code> | Split line in two at this point |

Operator Precedence

| Precedence | Description | Regexp |
|------------|---|---------------------------------|
| 1 | Grouping | <code>\ (\)</code> |
| 2 | Quantifiers. | <code>\ =, \ +, \ * etc.</code> |
| 3 | Characters/Metacharacters not containing grouping or quantifiers. | <code>abc, \w</code> |
| 4 | Alternation/"OR" | <code>\ </code> |

Search and Execution

| | |
|---------------------------------------|--|
| <code>:range g/patt ern/cmd</code> | Execute the Ex command cmd on the lines within range where pattern matches. |
| <code>:range g!/pat ter n/c md</code> | Execute the Ex command cmd on the lines within range where pattern <i>does not</i> occur. |

"Escaped" characters or metacharacters

| | | | |
|----|-------------------------------|----|-------------------------------|
| . | Any character except new line | | |
| \s | Whitespace character | \S | Non-Whitespace character |
| \w | Word character | \W | Non-Word character |
| \d | Digit | \D | Non-Digit |
| \a | Alphabetic character | \A | Non-Alphabetic character |
| \l | Lowercase character | \L | Non-Lowercase character |
| \u | Uppercase character | \U | Non-Uppercase character |
| \h | Head of a word character | \H | Non-head of word character |
| \p | Printable character | \P | like \p, but excluding digits |
| \x | Hex digit | \X | Non-Hex digit |
| \o | Octal digit | \O | Non-Octal digit |



By **truenipple58**

Not published yet.
Last updated 15th August, 2025.
Page 1 of 2.

Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
<https://apollopadd.com>

Quantifiers

| | |
|----------|--|
| Greedy | Greedy quantifiers first tries to repeat the token as many times as possible, and gradually gives up matches as the engine backtracks to find an overall match. |
| * | matches 0 or more of the preceding characters, ranges or metacharacters. * matches everything including empty line. |
| \+ | matches 1 or more of the preceding characters |
| \= | matches 0 or more of the preceding characters |
| \{n,m\} | matches from n to m of the preceding characters |
| \{n\} | matches exactly n times of the preceding characters |
| \{,m\} | matches at most m (from 0 to m) of the preceding characters |
| \{n,\} | matches at least n of the preceding characters |
| Lazy | Lazy quantifier first repeats the token as few times as required, and gradually expands the match as the engine backtracks through the regex to find an overall match. |
| \{-\} | matches 0 or more of the preceding atoms, as few as possible . |
| \{-n,m\} | matches from n to m of the preceding characters, as few as possible . |
| \{-n,\} | matches at least n of the preceding characters, as few as possible . |
| \{-,m\} | matches at most m (from 0 to m) of the preceding characters, as few as possible . |
| n, m > 0 | |

Additional Resources

1. [vimregex Archived Link](#)
It is an awesome and pretty comprehensive resource with examples and also provides a good summary, but not an aesthetically printable cheatsheet. Additionally, it never mentions very magic mode.
2. [Learn By Example: Vim Regular Expressions Archived Link](#)
Yet another resource, however does mention very magic mode.



By **truenipple58**

cheatography.com/truenipple58/

Not published yet.

Last updated 15th August, 2025.

Page 2 of 2.

Sponsored by **ApolloPad.com**

Everyone has a novel in them. Finish Yours!

<https://apollopad.com>