A regular expression is a sequence of characters that specifies a set of strings, which are said to match the regular expression.

For example, in one flavor of regular expression syntax:

\[ \text{gli}..\text{ering} \rightarrow \text{set of strings that begin with "gli", followed by any two characters, followed by "ering"} \]
Some Systems That Use REs

grep

vi/emacs/other text editors

most command shells (e.g., csh, bash, Windows shell)

many programming languages

Unfortunately, this does not imply that all use the same syntax rules for REs.

For historical reasons, there are many variations (flavors) of RE syntax.

For the sake of sanity, we will restrict ourselves to the grep flavor.
RE Syntax Metacharacters

Most characters simply stand for themselves.

Metacharacters have special meaning:

**period (.)**
matches any single character
- a.c is matched by aac, abc, a)c, etc.
- b..t is matched by beet, best, boot, bart, etc.

**asterisk (*)**
matches zero or more occurrences of the preceding RE
- ab*c is matched by ac, abc, abbc, abbbc, etc.
- .* is matched by all strings

**plus (+)**
matches one or more occurrences of the preceding RE
- ab+c is matched by abc, abbc, abbbc, etc., but not by ac
RE Syntax Examples

```bash
$ grep -E gli..er MobyDick.txt
```

a fine frosty night; how Orion glitters; what northern lights! Let them glittering teeth resembling ivory saws; others were tufted with knots of footfall in the passage, and saw a glimmer of light come into the room glittering in the clear, cold air. Huge hills and mountains of casks on glittering expression--all this sufficiently proclaimed him an inheritor looked celestial; seemed some plumed and glittering god uprising from suddenly relieved hull rolled away from it, to far down her glittering
the wife sat frozen at the window, with tearless eyes, glitteringly glittering fiddle-bows of whale ivory, were presiding over the hilarious to glimmer into sight. Glancing upwards, he cried: "See! see!" and once
At the first faintest glimmering of the dawn, his iron voice was heard leeward; and Ahab heading the onset. A pale, death-glimmer lit up glittering mouth yawned beneath the boat like an open-doored marble methodic intervals, the whale's glittering spout was regularly announced the moment, intolerably glittered and glared like a glacier; and

Note the use of the –E switch in the example here. This specifies to grep to use certain extensions to the basic RE syntax; rather than fuss about the difference, we will simply invoke grep with this switch in all cases.
RE Syntax Examples

$ grep -n -E fe+d MobyDick.txt

278: Tho' stuffed with hoops and armed with ribs of whale."
746: I stuffed a shirt or two into my old carpet-bag, tucked it under my arm,
1267: Whether that mattress was stuffed with corn-cobs or broken crockery,
1381: he puffed out great clouds of tobacco smoke. The next moment the light
1822: But Faith, like a jackal, feeds among the tombs, and even from these
2644: How I snuffed that Tartar air!—how I spurned that turnpike earth!—that
2903: Hosea's brindled cow feeding on fish remnants, and marching along the
4929: own. Yet now, federated along one keel, what a set these Isolatoes were!

Note the use of the –n switch in the example here. This specifies to grep to report
line numbers along with the matching lines.

$ grep -E traveler MobyDick.txt

the great New England traveller, and Mungo Park, the Scotch one; of all
palsied universe lies before us a leper; and like wilful travellers in
more travellers than in any other part.

...
question mark (?)
matches zero or one occurrence of the preceding RE
ab?c is matched by ac and abc, but not by abbc
b.?t is matched by bt, bat, bet, bxt, etc.

logical or (|)
matches the RE before | or the RE after |
abc | def is matched by abc and def and nothing else
RE Syntax Examples

$ grep -E fee?d  MobyDick.txt

Tho' stuffed with hoops and armed with ribs of whale."
I stuffed a shirt or two into my old carpet-bag, tucked it under my arm,
Whether that mattress was stuffed with corn-cobs or broken crockery,
he puffed out great clouds of tobacco smoke. The next moment the light

$ grep -E 'equal|same' MobyDick.txt

and some other articles of the same nature in their boats, in order to
"And pray, sir, what in the world is equal to it?" --EDMUND BURKE'S
to have indirectly hit upon new clews to that same mystic North-West
nearly the same feelings towards the ocean with me.

Note the use of single-quotes around the RE in the second example; this is
absolutely necessary in the Unix shell because the '|' character has special meaning
to the shell and that takes priority; the same applies in the Windows shell except
that double-quotes are used.
RE Syntax Metacharacters

caret (^)
used outside brackets, matches only at the beginning of a line
  ^D.* is matched by any line beginning with D
see slide 10 for semantics if inside brackets…

dollar sign ($) 
matches only at the end of a line
  .*d$ is matched by any line ending with a d
RE Syntax Examples

$ grep -E ^equal MobyDick.txt

equalled by the realities of the whalemens.
equally desolate Salisbury Plain in England; if casually encountering
equal to that of the brain. Under all these circumstances, would it be
equally doubted the story of Hercules and the whale, and Arion and the
...

$ grep -E equal$ MobyDick.txt
twenty pounds; so that the whole rope will bear a strain nearly equal

The first example does not work properly in the Windows shell unless you put double-quotes around the RE.
Regular Expressions

RE Syntax Metacharacters

backslash (\)
- escapes other metacharacters
  now\. is matched by "now."

square brackets []
- specify a set of characters as a set; any character in the set will match
  [aeiou] is matched by any vowel
  [a-z] is matched by any lower-case letter
  ^ specifies the complement (negation) of the set
  [^aeiou] is matched by any character but 'a', 'e', 'i', 'o' and 'u'

parentheses ()
- forms a group of characters to be treated as a unit
  a (bc) + is matched by abc, abcbc, abcbcbc, etc.

braces {}
- specifies the number of repetitions of an RE
  [a-z] {3} is matched by any three lower-case letters
RE Syntax Examples

$ grep -E 'equal(ly)?$' MobyDick.txt

twenty pounds; so that the whole rope will bear a strain nearly equal
even now beholding him; aye, and into the eye that is even now equally

$ grep -E '^f[aeiou]t' MobyDick.txt

fathoms down, and 'the weeds were wrapped about his head,' and all the
father was a High Chief, a King; his uncle a High Priest; and on the
future investigators, who may complete what I have here but begun. If
... fetch another for a considerable time. That is to say, he would then
fathoms of rope; as, after deep sounding, he floats up again, and shows
... fitted to sustain the weight of an almost solid mass of brick and
fatal cork, forth flew the fiend, and shrivelled up his home. Now, for
...
RE Syntax Examples

$ grep -E '^f[aeiou]+t' MobyDick.txt

foot of it. But I got a dreaming and sprawling about one night, and footfall in the passage, and saw a glimmer of light come into the room fathoms down, and 'the weeds were wrapped about his head,' and all the feet high; consisting of the long, huge slabs of limber black bone taken features of the leviathan, most naturalists have recognised him for one. future investigators, who may complete what I have here but begun. If faithfully narrated here, as they will not fail to elucidate several fitted to sustain the weight of an almost solid mass of brick and . . .

$ grep -E 'br(ing){2}' MobyDick.txt

myself involuntarily pausing before coffin warehouses, and bringing up justified his bringing his harpoon into breakfast with him, and using it bringing in good interest.
. . .
According to the man page for grep, the repetition expression \{,m\} should cause a match to m or fewer occurrences of the RE to which it is applied.

So, the following should match 1, 10, 100, 1000 and 10000:

```bash
$ grep -E '1(0){,4}' SomeNumbers.txt
```

However, the GNU implementation of grep, does not implement this as described.

Instead, you should use \{0,m\}, as in:

```bash
$ grep -E '1(0){0,4}' SomeNumbers.txt
```
word boundaries (\< and \>)
specifies to only match entire words (in a loose sense)
\<fat\> is matched by "fat" but not "father" or "fathom"

```
$ grep -E '\<fat\>' MobyDick.txt
nothing certain. They grow exceeding fat, insomuch that an incredible
DUTCH SAILOR. Grand snoozing to-night, maty; fat night for that. I
exceeding richness. He is the great prize ox of the sea, too fat to be
...
```

Of course, grep doesn't "understand" English. Word boundaries are indicated by the beginnings and ends of alphanumeric sequences of characters.
More Regular Expressions

Ending a sentence with a preposition is something up with which I will not put.

W Churchill

Some people, when confronted with a problem, think "I know, I'll use regular expressions." Now they have two problems.

Jamie Zawinski
Example

How can you search a file for sentences that end with a preposition?

It seems we need to determine two things:

- what are prepositions?
- what characters might mark the end of a sentence?

The second question seems to be fairly easy: . ! ?

Some sentences end with a double-quotiation mark, but that will probably be preceded by one of the marks above. And some end with an ellipsis…

This suggests:

```
[.?!] | \.\.\.\.
```
Example

So, what are prepositions? A preposition relates a noun or pronoun to another word in a sentence.

One source says there are 150 of them and gives the following partial list:

- aboard
- along
- at
- besides
- considering
- excluding
- into
- on
- per
- than
- underneath
- via
- about
- amid
- before
- between
- despite
- following
- like
- onto
- plus
- through
- unlike
- with
- above
- among
- behind
- beyond
- down
- for
- minus
- opposite
- regarding
- to
- until
- within
- across
- anti
- below
- but
- during
- from
- near
- outside
- over
- save
- toward
- up
- without
- after
- around
- beneath
- by
- except
- in
- of
- off
- past
- since
- under
- versus

Allegedly, the most common ones are:

to, of, in, for, on, with, at, by, from, up, about, into, over, after
This suggests the regular expression used below:

```
$ grep -E '(%<to\>|%<of\>|%<in\>|%<for\>|%<on\>|%<with\>|%<at\>|%<by\>|%<from\>|%<up\>|%<about\>|%<into\>|%<over\>|%<after\>)
(%.?!|\./\./.)' MobyDick.txt
```

once a whale in Spitzbergen that was white all over." --A VOYAGE TO
up a pair of as pretty rainbows as a Christian would wish to look at.
as they possibly can without falling in. And there they stand--miles of
penny that I ever heard of. On the contrary, passengers themselves must
one lodges in.
as a looker on.
the tidiest, certainly none of the finest. I began to twitch all over.
leaving a little interval between, for my back to settle down in. But I
till spoken to. Holding a light in one hand, and that identical New
out a sort of tomahawk, and a seal-skin wallet with the hair on. Placing
he never would have dreamt of getting under the bed to put them on. At
be sure there is more in that man than you perhaps think for.
night previous, and whom I had not as yet had a good look at. They were
to. Then the Captain knows that Jonah is a fugitive; but at the same
an adventurous whaleman to embark from. He at once resolved to accompany
whom I now companied with.

...
POSIX Character Classes

The POSIX definition of extended regular expressions includes definitions of some classes of characters, including:

<table>
<thead>
<tr>
<th>POSIX</th>
<th>ASCII</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[:alnum:]</td>
<td>[A-Za-z0-9]</td>
<td>alphanumeric characters</td>
</tr>
<tr>
<td>[:alpha:]</td>
<td>[A-Za-z]</td>
<td>alphabetic characters</td>
</tr>
<tr>
<td>[:blank:]</td>
<td>[ \t]</td>
<td>space and tab</td>
</tr>
<tr>
<td>[:digit:]</td>
<td>[0-9]</td>
<td>digits</td>
</tr>
<tr>
<td>[:graph:]</td>
<td>[\x21-\x7E]</td>
<td>visible characters</td>
</tr>
<tr>
<td>[:print:]</td>
<td>[\x20-\x7E]</td>
<td>visible characters and space</td>
</tr>
<tr>
<td>[:lower:]</td>
<td>[a-z]</td>
<td>lower-case letters</td>
</tr>
<tr>
<td>[:upper:]</td>
<td>[A-Z]</td>
<td>upper-case letters</td>
</tr>
<tr>
<td>[:space:]</td>
<td>[ \t\r\n\v\f]</td>
<td>whitespace characters</td>
</tr>
<tr>
<td>[:punct:]</td>
<td>[!&quot;#$%&amp;'()*+,.:/;&lt;=?@^_`{</td>
<td>}~]</td>
</tr>
</tbody>
</table>
Example

Let's use a character class to look for digits in a file (note the syntax):

```bash
$ grep -E \[:digit:\] MobyDick.txt
```

Last Updated: January 3, 2009
Posting Date: December 25, 2008 [EBook #2701]
Release Date: June, 2001

In chapters 24, 89, and 90, we substituted a capital L for the symbol
NARRATIVE TAKEN DOWN FROM HIS MOUTH BY KING ALFRED, A.D. 890.
GREENLAND, A.D. 1671 HARRIS COLL.
"Several whales have come in upon this coast (Fife) Anno 1652, one informed), besides a vast quantity of oil, did afford 500 weight of
STRAFFORD'S LETTER FROM THE BERMUDAS. PHIL. TRANS. A.D. 1668.
northward of us." --CAPTAIN COWLEY'S VOYAGE ROUND THE GLOBE, A.D. 1729.
ON BANKS'S AND SOLANDER'S VOYAGE TO ICELAND IN 1772.
--THOMAS JEFFERSON'S WHALE MEMORIAL TO THE FRENCH MINISTER IN 1778.
"In 40 degrees south, we saw Spermacetti Whales, but did not take
"In the year 1690 some persons were on a high hill observing the
SAID VESSEL. NEW YORK, 1821.
of this one whale, amounted altogether to 10,440 yards or nearly six
--THOMAS BEALE'S HISTORY OF THE SPERM WHALE, 1839.
--FREDERICK DEBELL BENNETT'S WHALING VOYAGE ROUND THE GLOBE, 1840.
October 13. "There she blows," was sung out from the mast-head.
--J. ROSS BROWNE'S ETCHINGS OF A WHALING CRUIZE. 1846.
...

CS@VT
Computer Organization I
©2005-2012 McQuain
Example

Let's use character classes to look for strings that consist of one or more alphabetic characters followed immediately by one or more digits:

```
$ grep -E '[:alpha:]+[:digit:]+' MobyDick.txt
```

upwards of L1,000,000? And lastly, how comes it that we whalemen of Savesoul's income of L100,000 seized from the scant bread and cheese without any of Savesoul's help) what is that globular L100,000 but a fish high and dry, promising themselves a good L150 from the precious PROVIDED IN PARAGRAPH F3.  YOU AGREE THAT THE FOUNDATION, THE
Why the POSIX Classes?

Suppose you need to use a regular expression for a search on a system that does not use ASCII encoding for characters?

The order in which character codes are assigned to characters may not be compatible with ASCII.

So, it could be that A-Z doesn't define a valid range that includes all capital letters and nothing else.

Now, you might be able to figure out a workable range specification…

… but you wouldn't have a portable solution.

The POSIX classes give us a way to manage these issues in a portable manner.

Fortunately, GNU grep does support the POSIX classes described earlier.
Examples

What do you think the following searches will find?

```
$ grep -E '\<the\>\<Pequod\>' MobyDick.txt

$ grep -E '\<[Cc]aptain\>\<Ahab\>' MobyDick.txt

$ grep -E '\<[Cc]aptain\> \<Ahab\>' MobyDick.txt

$ grep -E '\<better\> \<than\> \<nothing\>' MobyDick.txt

$ grep -E 'better than nothing' MobyDick.txt
```
-i, --ignore-case
  Ignore case distinctions in both the PATTERN and the input files.

-v, --invert-match
  Invert the sense of matching, to select non-matching lines.

-w, --word-regexp
  Select only those lines containing matches that form whole words. The test is that the matching substring must either be at the beginning of the line, or preceded by a non-word constituent character. Similarly, it must be either at the end of the line or followed by a non-word constituent character. Word-constituent characters are letters, digits, and the underscore.

-x, --line-regexp
  Select only those matches that exactly match the whole line.

-c, --count
  Suppress normal output; instead print a count of matching lines for each input file. With the -v, --invert-match option (see below), count non-matching lines.

-o, --only-matching
  Print only the matched (non-empty) parts of a matching line, with each such part on a separate output line.
**-m NUM, --max-count=NUM**

Stop reading a file after NUM matching lines. If the input is standard input from a regular file, and NUM matching lines are output, grep ensures that the standard input is positioned to just after the last matching line before exiting, regardless of the presence of trailing context lines. This enables a calling process to resume a search. When grep stops after NUM matching lines, it outputs any trailing context lines. When the -c or --count option is also used, grep does not output a count greater than NUM. When the -v or --invert-match option is also used, grep stops after outputting NUM non-matching lines.

**-n, --line-number**

Prefix each line of output with the 1-based line number within its input file.

**-A NUM, --after-context=NUM**

Print NUM lines of trailing context after matching lines. Places a line containing a group separator (--) between contiguous groups of matches. With the -o or --only-matching option, this has no effect and a warning is given.

**-B NUM, --before-context=NUM**

Print NUM lines of leading context before matching lines. Places a line containing a group separator (--) between contiguous groups of matches. With the -o or --only-matching option, this has no effect and a warning is given.