Learn Ruby (v1.8.7) With the Edgecase Ruby Koans *Online*

about_regular_expressions

Click to submit Meditation or press Enter while in the form.

```
# -*- coding: utf-8 -*-

class AboutRegularExpressions < EdgeCase::Koan
  def test_a_pattern_is_a_regular_expression
    assert_equal Regexp , /pattern/.class
  end


  def test_a_regexp_can_search_a_string_for_matching_content
    assert_equal "match" , "some matching content"[/match/]
  end


  def test_a_failed_match_returns_nil
    assert_equal nil , "some matching content"[/missing/]
```

end

Please meditate on the following.
def test_question_mark_means_optional
  assert_equal [redacted], "abbcccddddeeeee"[/ab?/]
  assert_equal [redacted], "abbcccddddeeeee"[/az?/]
end

Please meditate on the following.
def test_plus_means_one_or_more
  assert_equal [redacted], "abbcccddddeeeee"[/bc+/]
end

Please meditate on the following.
def test_asterisk_means_zero_or_more
  assert_equal [redacted], "abbcccddddeeeee"[/ab*/]
  assert_equal [redacted], "abbcccddddeeeee"[/az*/]
  assert_equal [redacted], "abbcccddddeeeee"[/z*/]
  # THINK ABOUT IT:
  #
  # When would * fail to match?
end

THINK ABOUT IT:
#
We say that the repetition operators above are "greedy."
#
Why?

Please meditate on the following.
def test_the_left_most_match_wins
  assert_equal [redacted], "abbccc az"[/az*/]
end

Please meditate on the following.
def test_character_classes_give_options_for_a_character
  animals = ["cat", "bat", "rat", "zat"]

```
  assert_equal [redacted], animals.select { |a| a[/[cbr]at/] }
end


Please meditate on the following.
def test_slash_d_is_a_shortcut_for_a_digit_character_class
  assert_equal [redacted], "the number is 42"[/[0123456789]+/]
  assert_equal [redacted], "the number is 42"[/\d+/]
end


Please meditate on the following.
def test_character_classes_can_include_ranges
  assert_equal [redacted], "the number is 42"[/[0-9]+/]
end


Please meditate on the following.
def test_slash_s_is_a_shortcut_for_a_whitespace_character_class
  assert_equal [redacted], "space: \t\n"[/\s+/]
end


Please meditate on the following.
def test_slash_w_is_a_shortcut_for_a_word_character_class
  # NOTE:  This is more like how a programmer might define a word.
  assert_equal [redacted], "variable_1 = 42"[/[a-zA-Z0-9_]+/]
  assert_equal [redacted], "variable_1 = 42"[/\w+/]
end


Please meditate on the following.
def test_period_is_a_shortcut_for_any_non_newline_character
  assert_equal [redacted], "abc\n123"[/a.+/]
end


Please meditate on the following.
def test_a_character_class_can_be_negated
  assert_equal [redacted], "the number is 42"[/[^0-9]+/]
end


Please meditate on the following.
def test_shortcut_character_classes_are_negated_with_capitals
  assert_equal [redacted], "the number is 42"[/\D+/]
  assert_equal [redacted], "space: \t\n"[/\S+/]
```

```
  assert_equal [redacted], "variable_1 = 42"[/\W+/]
end


# ---------------------------------------------------------------

Please meditate on the following.
def test_slash_a_anchors_to_the_start_of_the_string
  assert_equal [redacted], "start end"[/\Astart/]
  assert_equal [redacted], "start end"[/\Aend/]
end


Please meditate on the following.
def test_slash_z_anchors_to_the_end_of_the_string
  assert_equal [redacted], "start end"[/end\z/]
  assert_equal [redacted], "start end"[/start\z/]
end


Please meditate on the following.
def test_caret_anchors_to_the_start_of_lines
  assert_equal [redacted], "num 42\n2 lines"[/^\d+/]
end


Please meditate on the following.
def test_dollar_sign_anchors_to_the_end_of_lines
  assert_equal [redacted], "2 lines\nnum 42"[/\d+$/]
end


Please meditate on the following.
def test_slash_b_anchors_to_a_word_boundary
  assert_equal [redacted], "bovine vines"[/\bvine./]
end


# ---------------------------------------------------------------

Please meditate on the following.
def test_parentheses_group_contents
  assert_equal [redacted], "ahahaha"[/(ha)+/]
end


# ---------------------------------------------------------------
```

Please meditate on the following.
```
def test_parentheses_also_capture_matched_content_by_number
  assert_equal [illegible], "Gray, James"[/(\w+), (\w+)/, 1]
  assert_equal [illegible], "Gray, James"[/(\w+), (\w+)/, 2]
end
```

Please meditate on the following.
```
def test_variables_can_also_be_used_to_access_captures
  assert_equal [illegible], "Name:  Gray, James"[/(\w+), (\w+)/]
  assert_equal [illegible], $1
  assert_equal [illegible], $2
end
```

```
# --------------------------------------------------------------
```

Please meditate on the following.
```
def test_a_vertical_pipe_means_or
  grays = /(James|Dana|Summer) Gray/
  assert_equal [illegible], "James Gray"[grays]
  assert_equal [illegible], "Summer Gray"[grays, 1]
  assert_equal [illegible], "Jim Gray"[grays, 1]
end
```

```
# THINK ABOUT IT:
#
# Explain the difference between a character class ([...]) and alternation (|).

# --------------------------------------------------------------
```

Please meditate on the following.
```
def test_scan_is_like_find_all
  assert_equal [illegible], "one two-three".scan(/\w+/)
end
```

Please meditate on the following.
```
def test_sub_is_like_find_and_replace
  assert_equal [illegible], "one two-three".sub(/(t\w*)/) { $1[0, 1] }
end
```

Please meditate on the following.
```
def test_gsub_is_like_find_and_replace_all
  assert_equal [illegible], "one two-three".gsub(/(t\w*)/) { $1[0, 1] }
```

```
  end

end
```

Content

Thanks for trying the Ruby Koans Online. We hope you're having fun.

Issues

Submit issues to the Github project.

Cheers

Blog about it or send us an email.

Get More

Download the official Ruby Koans and get the full experience.