Mutation Testing

@DonSchado

cologne.rb | 16.10.2013

RailsLove

disclaimer

This presentation contains:

- memes
- a definition
- lots of screenshots
- overdrawn code examples
- a short quiz time
- surviving mutants
- no ponies

```
def mutation_testing
  <<-eos

    is used to evaluate the quality of existing software tests

    modifying the program's source code

    based on well-defined mutation operators

    each 'mutated' version is called a mutant

    a test which detects the mutation (fail) will kill the mutant

    mutants without failing tests are alive

  eos
end
```

```
def mutation_testing
  <<-eos

    is used to evaluate the quality of existing software tests

    modifying the program's source code

    based on well-defined mutation operators

    each 'mutated' version is called a mutant

    a test which detects the mutation (fail) will kill the mutant

    mutants without failing tests are ALIVE

  eos
end
```


```
gem 'mutant', '~> 0.3.0.rc3'
```

Rubygem mutant

Total Downloads	Releases	Current Version	Released	First Release
↑ 24803	45	0.3.0.rc3	28 days ago	about a year ago

Depends on following gems

abstract_type, adamantium, backports, descendants_tracker, diff-lcs, equalizer, ice_nine, inflecto, rspec, to_source

Depending Gems

9

Popular gems depending on mutant

crystalline, guard-mutant, devutils-metrics, ruote-synchronize, ruote-resque, mutant-rails, citrus-core, meta_module, develry

Github mbj/mutant

Watchers	Forks	Development activity	Last commit	First commit
213	15	↑ Active	24 days ago	

Top contributors

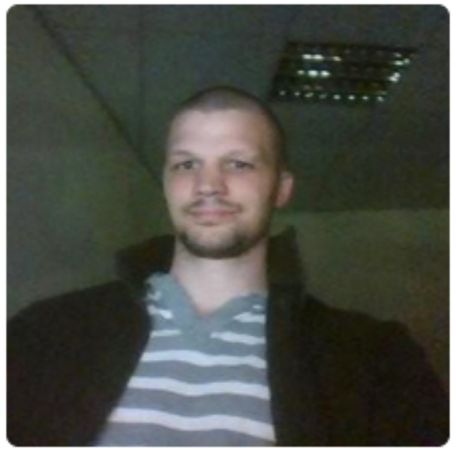
mbj, dkubb, solnic, snusnu, clekstro, splattael, postmodern, benmoss, nevir, and jessekempf

Contributors

12

Issues

36



Markus Schirp
mbj

self-employed / freelancing
Essen, Germany
mbj@schirp-dso.com
https://twitter.com/_m_b_j_
Joined on Aug 21, 2009

43 followers **141** starred **4** following

Organizations

Piotr Solnica
solnic

Powow AS
Poland, Kraków
http://solnic.eu
Joined on Feb 27, 2008

230 followers **268** starred **157** following

Organizations

Dan Kubb
dkubb

Freelancer
Mission, BC, Canada
dan.kubb@gmail.com
Joined on Feb 03, 2008

302 followers **100** starred **4** following

Organizations

Projects using Mutant

The following projects adopted mutant, and aim 100% mutation coverage:

- axiom
- axiom-types
- rom-mapper
- rom-session
- event_bus
- virtus
- quacky
- substation
- large_binomials

mbj/mutant

Summary of **September 9th - 15th**

54 files changed, 1,016 insertions, 878 deletions

One class/module was **added**. 29 days ago

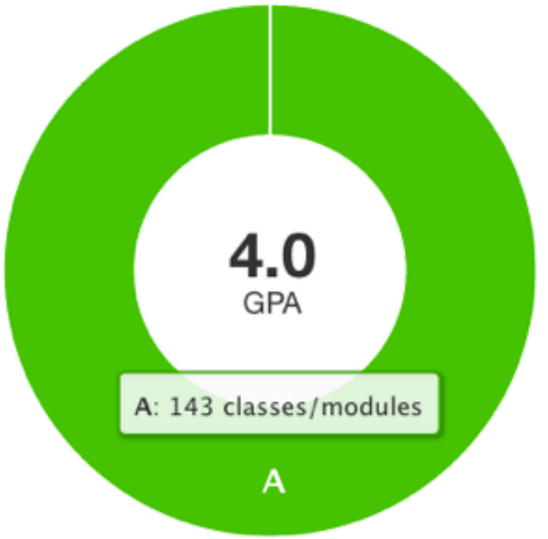
- A Mutant::Predicate::Matcher

B → A Mutant::CLI::Classifier has **improved**. about a month ago

Two classes/modules were **added**. about a month ago

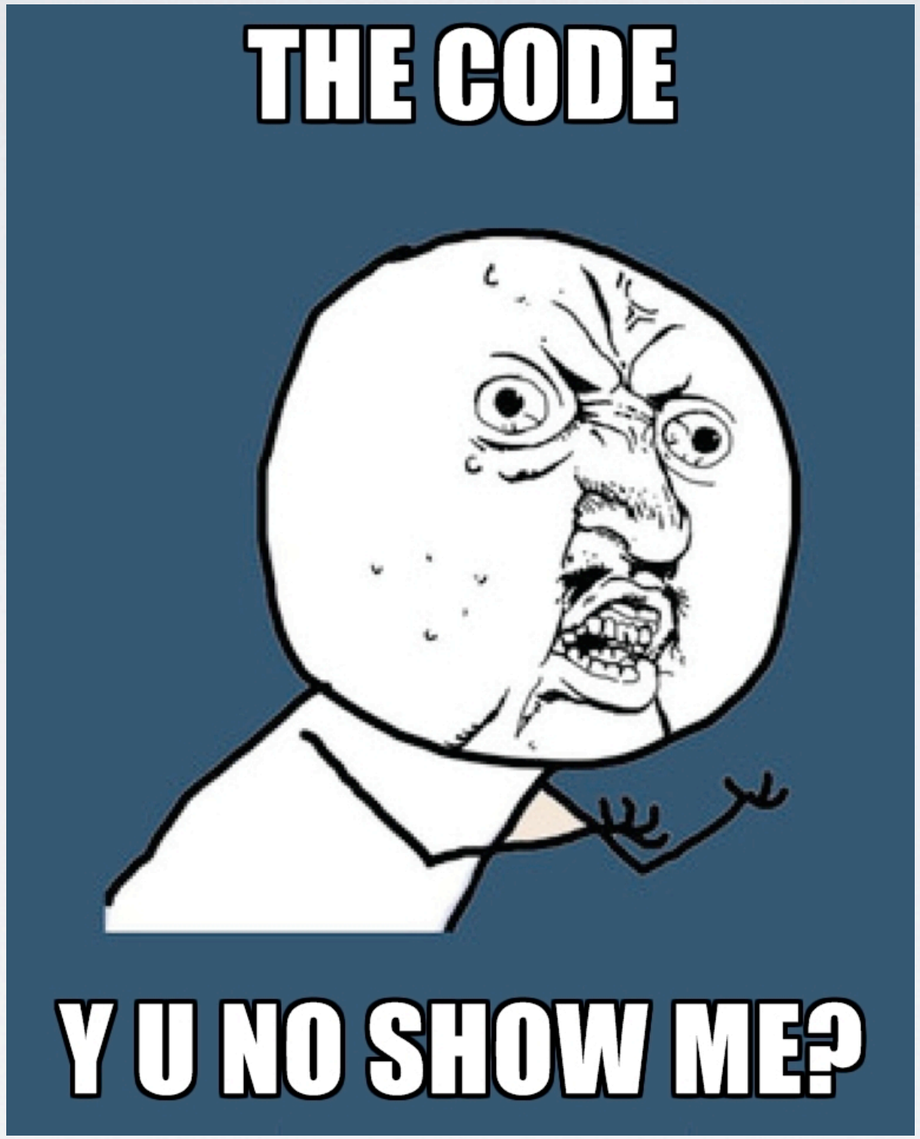
- A Mutant::Mutator::Node::Blockarg
- A Mutant::Mutator::Node::Dsym

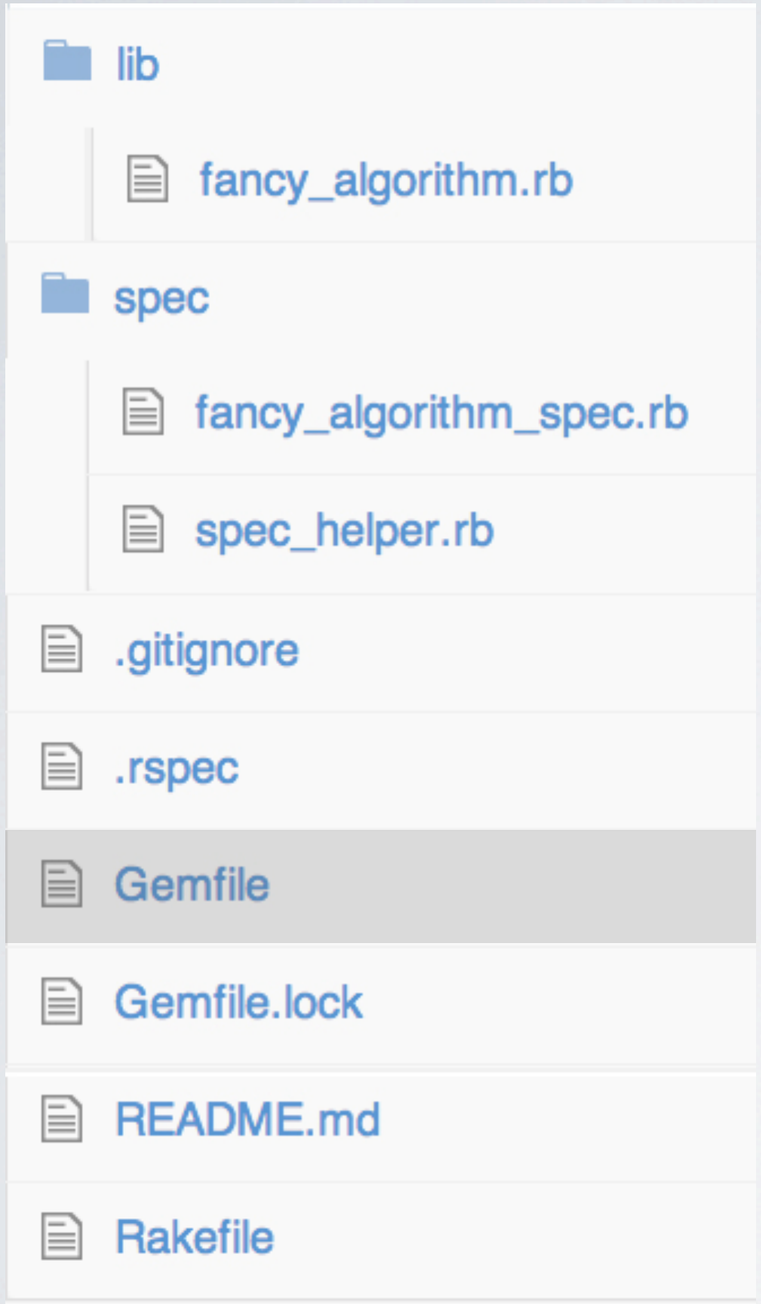
Classes by Rating



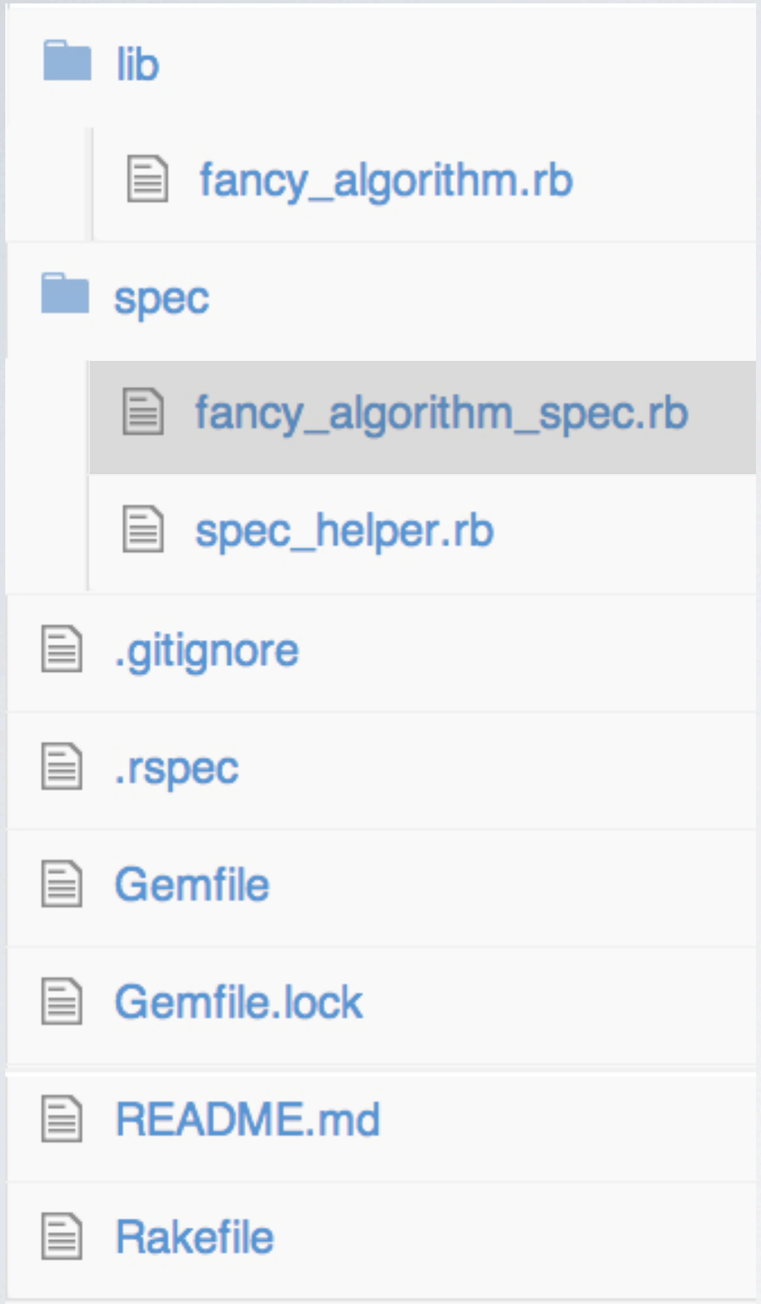
Hotspots

Huzzah! This repo has no classes or modules worse than a B.

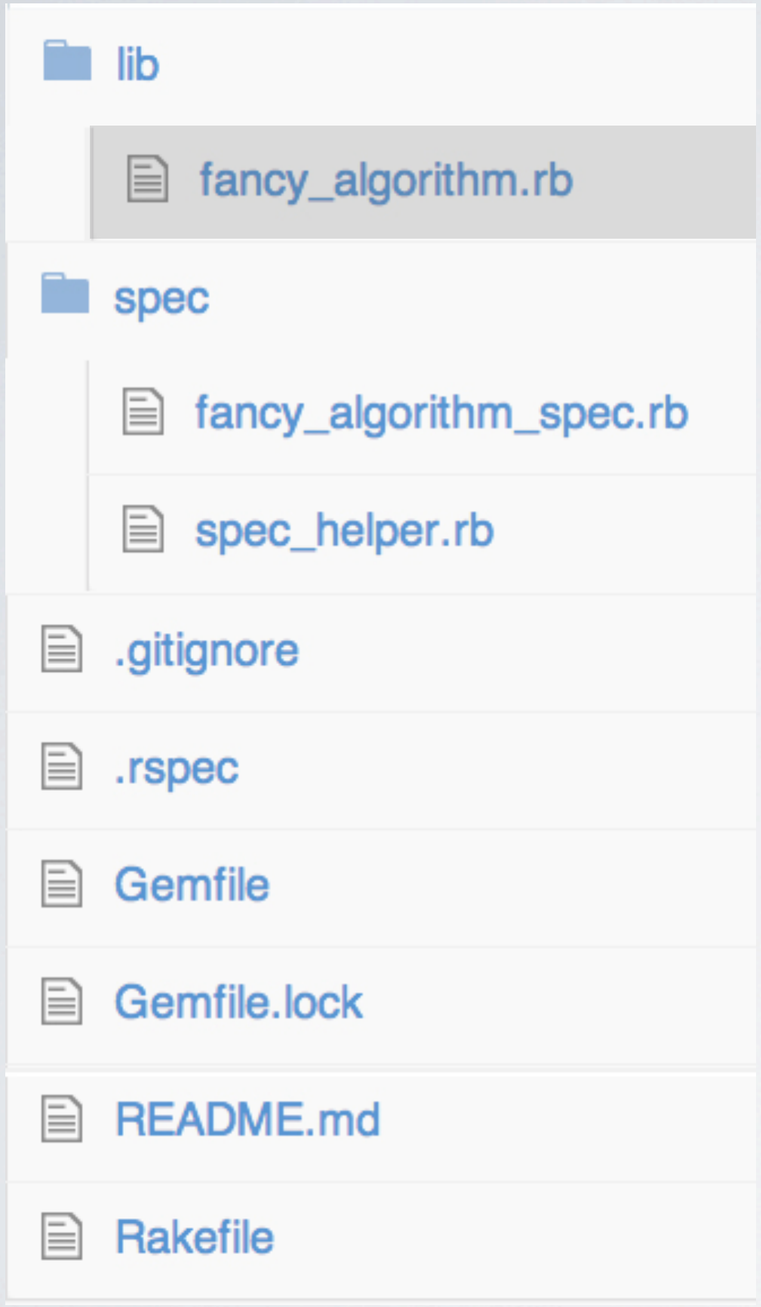

THE CODE
Y U NO SHOW ME?




```
source "https://rubygems.org"

gem 'simplecov', "~> 0.8.0.pre2"
gem "rake", "~> 10.1.0"
gem "rspec", "~> 2.14.0"
gem "mutant", "~> 0.3.0.rc3"
gem "rspec-pride", "~> 2.2.0", :require => false
gem "activesupport", "~> 4.0.0", :require => false
```




```ruby
require 'spec_helper'

describe 'FancyAlgorithm' do

  let(:random_list) { %w[2 1 3 5 6 4 9 8 7] }
  let(:sorted_list) { %w[1 2 3 4 5 6 7 8 9] }

  context ':sort' do
    subject { FancyAlgorithm.new(:sort) }
    it { expect(subject.perform(random_list)).to eql(sorted_list) }
  end

  context ':unsort' do
    subject { FancyAlgorithm.new(:unsort) }
    it { expect(subject.perform(random_list)).to_not eql(sorted_list) }
  end

end
```




```ruby
class FancyAlgorithm
  attr_accessor :strategy

  def initialize(strategy)
    @strategy = strategy.to_s
  end

  def perform(list)
    self.send("#{strategy}_algorithm!", list)
  end

  private

    def sort_algorithm!(list)
      return list if list.size <= 1
      0.upto(list.size - 1) do |i|
        (list.size - 1).downto(i + 1) do |j|
          if list[j] < list[j - 1]
            list[j], list[j - 1] = list[j - 1], list[j]
          end
        end
      end
      list
    end

    def unsort_algorithm!(list)
      return list.shuffle unless list.empty?
    end
end
```

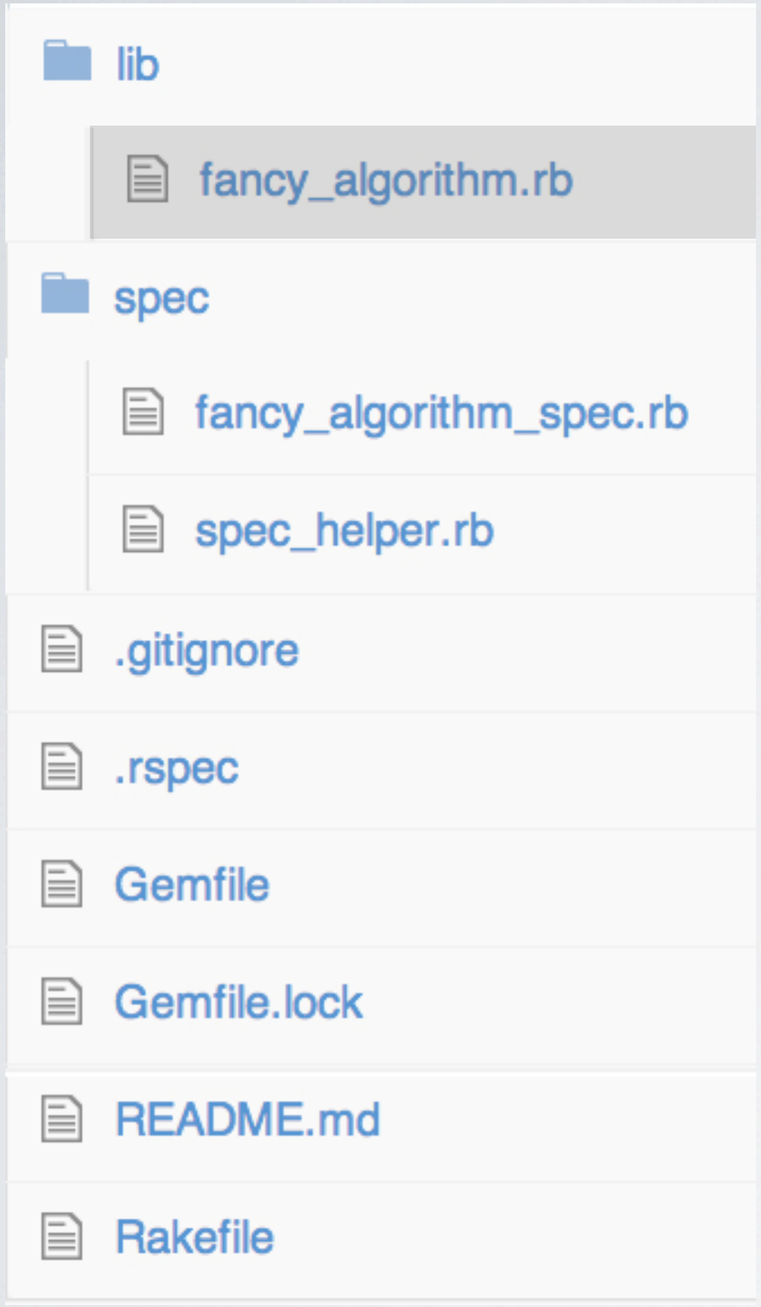
- lib
  - fancy_algorithm.rb
- spec
  - fancy_algorithm_spec.rb
  - spec_helper.rb
- .gitignore
- .rspec
- Gemfile
- Gemfile.lock
- README.md
- Rakefile

```ruby
class FancyAlgorithm
  attr_accessor :strategy

  def initialize(strategy)
    @strategy = strategy.to_s
  end

  def perform(list)
    self.send("#{strategy}_algorithm!", list)
  end

  private

    def sort_algorithm!(list)
      return list if list.size <= 1
      0.upto(list.size - 1) do |i|
        (list.size - 1).downto(i + 1) do |j|
          if list[j] < list[j - 1]
            list[j], list[j - 1] = list[j - 1], list[j]
          end
        end
      end
      list
    end

    def unsort_algorithm!(list)
      return list.shuffle unless list.empty?
    end
end
```

This is obviously more advanced than Array#sort.

http://bigocheatsheet.com/

http://www.igvita.com/2009/03/26/ruby-algorithms-sorting-trie-heaps/

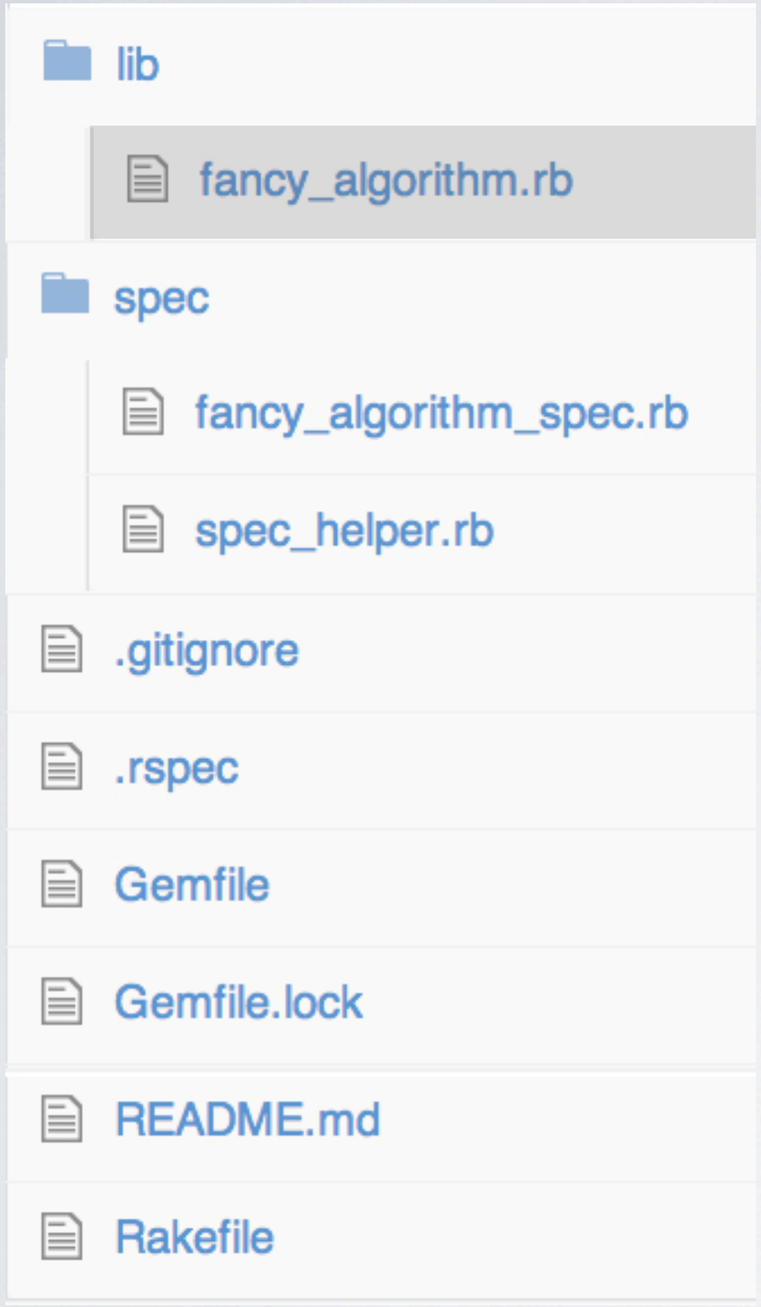



```ruby
class FancyAlgorithm
  attr_accessor :strategy

  def initialize(strategy)
    @strategy = strategy.to_s
  end

  def perform(list)
    self.send("#{strategy}_algorithm!", list)
  end

  private

    def sort_algorithm!(list)
      return list if list.size <= 1
      0.upto(list.size - 1) do |i|
        (list.size - 1).downto(i + 1) do |j|
          if list[j] < list[j - 1]
            list[j], list[j - 1] = list[j - 1], list[j]
          end
        end
      end
      list
    end

    def unsort_algorithm!(list)
      return list.shuffle unless list.empty?
    end
end
```

Quiztime:
Which sorting algorithm is this?

http://bigocheatsheet.com/

http://www.igvita.com/2009/03/26/ruby-algorithms-sorting-trie-heaps/

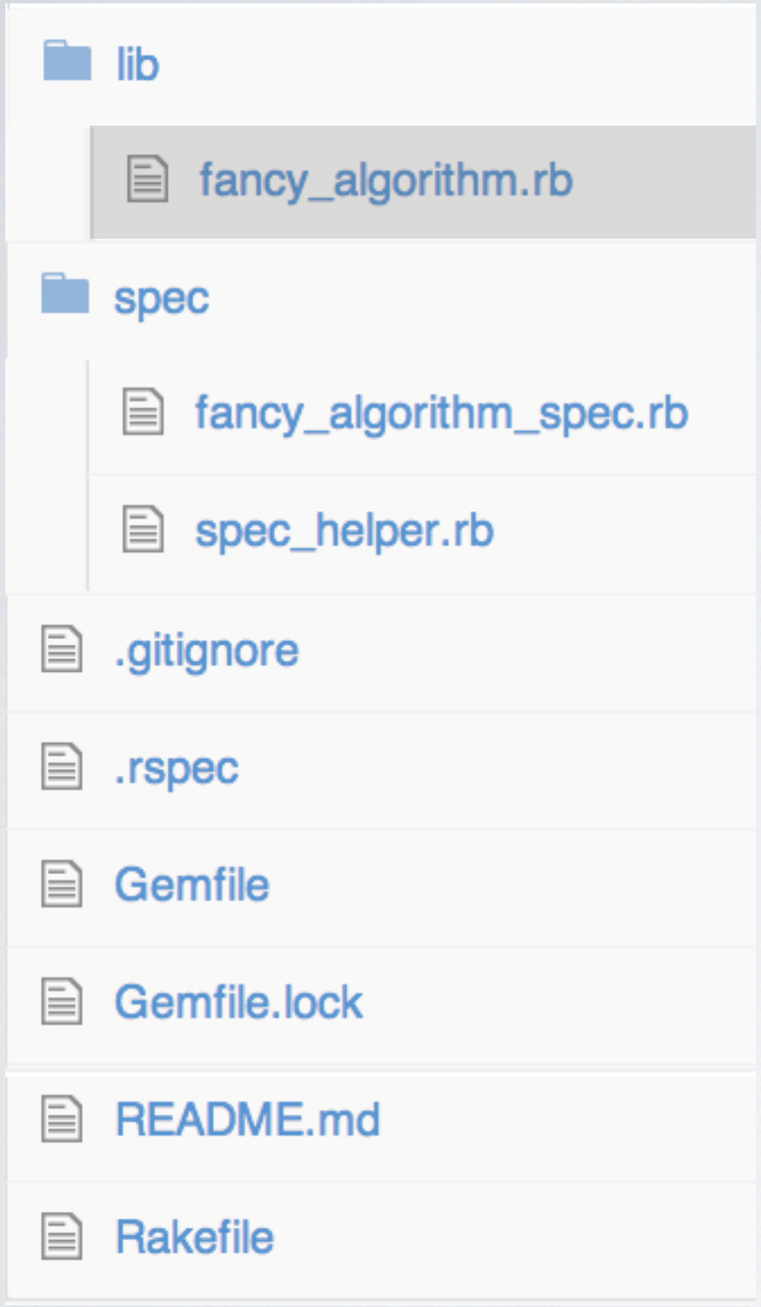



```ruby
class FancyAlgorithm
  attr_accessor :strategy

  def initialize(strategy)
    @strategy = strategy.to_s
  end

  def perform(list)
    self.send("#{strategy}_algorithm!", list)
  end

  private

    def sort_algorithm!(list)
      return list if list.size <= 1
      0.upto(list.size - 1) do |i|
        (list.size - 1).downto(i + 1) do |j|
          if list[j] < list[j - 1]
            list[j], list[j - 1] = list[j - 1], list[j]
          end
        end
      end
      list
    end

    def unsort_algorithm!(list)
      return list.shuffle unless list.empty?
    end
end
```

Quiztime:
What's the average Big-O complexity of this sorting algorithm?

http://bigocheatsheet.com/

http://www.igvita.com/2009/03/26/ruby-algorithms-sorting-trie-heaps/




```ruby
class FancyAlgorithm
  attr_accessor :strategy

  def initialize(strategy)
    @strategy = strategy.to_s
  end

  def perform(list)
    self.send("#{strategy}_algorithm!", list)
  end

  private

    def sort_algorithm!(list)
      return list if list.size <= 1
      0.upto(list.size - 1) do |i|
        (list.size - 1).downto(i + 1) do |j|
          if list[j] < list[j - 1]
            list[j], list[j - 1] = list[j - 1], list[j]
          end
        end
      end
      list
    end

    def unsort_algorithm!(list)
      return list.shuffle unless list.empty?
    end
end
```

Quiztime:
What's the sorting algorithm behind Ruby's Array#sort?

http://bigocheatsheet.com/

http://www.igvita.com/2009/03/26/ruby-algorithms-sorting-trie-heaps/

```
Fabulous tests in 0.001248 seconds
2 examples, 0 failures, 0 pending
```

lib/fancy_algorithm.rb

100.0 % covered

16 relevant lines. 16 lines covered and 0 lines missed.

```
1.  class FancyAlgorithm                                          1
2.    attr_accessor :strategy                                     1
3.
4.    def initialize(strategy)                                    1
5.      @strategy = strategy.to_s                                 2
6.    end
7.
8.    def perform(list)                                           1
9.      self.send("#{strategy}_algorithm!", list)                 2
10.   end
11.
12.   private                                                     1
13.
14.     def sort_algorithm!(list)                                 1
15.       return list if list.size <= 1                           1
16.
17.       0.upto(list.size - 1) do |i|                            1
18.         (list.size - 1).downto(i + 1) do |j|                  9
19.           if list[j] < list[j - 1]                            38
20.             (list[j], list[j - 1] = list[j - 1],              6
list[j])
```

```
$ mutant --help
usage: mutant STRATEGY [options] MATCHERS ...


Strategies:
        --rspec                      kills mutations with rspec
        --rspec-level LEVEL          set rspec expansion level
        --ignore-subject MATCHER     ignores subjects that matches MATCHER
        --zombie                     Run mutant zombified
    -I, --include DIRECTORY          Add DIRECTORY to $LOAD_PATH
    -r, --require NAME               Require file with NAME

Options:
        --version                    Print mutants version
        --code FILTER                Adds a code filter
        --fail-fast                  Fail fast
    -d, --debug                      Enable debugging output
    -h, --help                       Show this message
```

```
$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm#initialize
```

```
$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm#initialize
Mutant configuration:
Matcher:        #<Mutant::Matcher::Method::Instance cache=#<Mutant::Cache>
                 scope=FancyAlgorithm method=#<UnboundMethod: FancyAlgorithm#initialize>>
Subject Filter: Mutant::Predicate::CONTRADICTION
Strategy:       #<Mutant::Strategy::Rspec level=0>
FancyAlgorithm#initialize:/../lib/fancy_algorithm.rb:4
......F...
(09/10)  90% - 0.79s
FancyAlgorithm#initialize:/../lib/fancy_algorithm.rb:4
evil:FancyAlgorithm#initialize:/../lib/fancy_algorithm.rb:4:b34d0
@@ -1,4 +1,4 @@
 def initialize(strategy)
-  @strategy = strategy.to_s
+  @strategy = strategy
 end
(09/10)  90% - 0.79s
Subjects:  1
Mutations: 10
Kills:     9
Runtime:   0.86s
Killtime:  0.79s
Overhead:  8.84%
Coverage:  90.00%
Alive:     1
```

```
$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm#initialize
Mutant configuration:
Matcher:        #<Mutant::Matcher::Method::Instance cache=#<Mutant::Cache>
                 scope=FancyAlgorithm method=#<UnboundMethod: FancyAlgorithm#initialize>>
Subject Filter: Mutant::Predicate::CONTRADICTION
Strategy:       #<Mutant::Strategy::Rspec level=0>
FancyAlgorithm#initialize:/../lib/fancy_algorithm.rb:4
......F...
(09/10)  90% - 0.79s
FancyAlgorithm#initialize:/../lib/fancy_algorithm.rb:4
evil:FancyAlgorithm#initialize:/../lib/fancy_algorithm.rb:4:b34d0
@@ -1,4 +1,4 @@
 def initialize(strategy)
-  @strategy = strategy.to_s
+  @strategy = strategy
 end
(09/10)  90% - 0.79s
Subjects:  1
Mutations: 10
Kills:     9
Runtime:   0.86s
Killtime:  0.79s
Overhead:  8.84%
Coverage:  90.00%
Alive:     1
```

```ruby
def initialize(strategy)
  @strategy = strategy.to_s
end

def perform(list)
  self.send("#{strategy}_algorithm!", list)
end
```

implicit!

```
$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm
```

```
$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm
Mutant configuration:
...

FancyAlgorithm#initialize:/../lib/fancy_algorithm.rb:4
.......F...
(09/10)  90% - 0.80s
FancyAlgorithm#perform:/../lib/fancy_algorithm.rb:8
........F..........
(17/18)  94% - 1.44s
FancyAlgorithm#sort_algorithm!:/../lib/fancy_algorithm.rb:14
............FFF.F..FFFFF.........................F........
(47/57)  82% - 4.87s
FancyAlgorithm#unsort_algorithm!:/../lib/fancy_algorithm.rb:27
....FF.FFFFFFF.FFF
(06/18)  33% - 1.52s

Subjects:  4
Mutations: 103
Kills:     79
Runtime:   9.46s
Killtime:  8.62s
Overhead:  8.84%
Coverage:  76.70%
Alive:     24
```

```
$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm
Mutant configuration:
...

FancyAlgorithm#initialize:/../lib/fancy_algorithm.rb:4
.......F...
(09/10)  90% - 0.80s
FancyAlgorithm#perform:/../lib/fancy_algorithm.rb:8
........F..........
(17/18)  94% - 1.44s
FancyAlgorithm#sort_algorithm!:/../lib/fancy_algorithm.rb:14
............FFF.F..FFFFF.........................F........
(47/57)  82% - 4.87s
FancyAlgorithm#unsort_algorithm!:/../lib/fancy_algorithm.rb:27
....FF.FFFFFFF.FFF
(06/18)  33% - 1.52s

Subjects:  4
Mutations: 103
Kills:     79
Runtime:   9.46s
Killtime:  8.62s
Overhead:  8.84%
Coverage:  76.70%
Alive:     24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```
-   self.send("#{strategy}_algorithm!", list)
+   send("#{strategy}_algorithm!", list)
```

```ruby
def perform(list)
  self.send("#{strategy}_algorithm!", list)
end
```

```
Subjects:   4
Mutations:  103
Kills:      79
Runtime:    9.46s
Killtime:   8.62s
Overhead:   8.84%
Coverage:   76.70%
Alive:      24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```ruby
def sort_algorithm!(list)
  return list if list.size <= 1
  0.upto(list.size - 1) do |i|
    (list.size - 1).downto(i + 1) do |j|
      if list[j] < list[j - 1]
        list[j], list[j - 1] = list[j - 1], list[j]
      end
    end
  end
  list
end
```

```
Subjects:   4
Mutations:  103
Kills:      79
Runtime:    9.46s
Killtime:   8.62s
Overhead:   8.84%
Coverage:   76.70%
Alive:      24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```
-  if list.size <= 1
+  if list.size <= -1
```

```ruby
def sort_algorithm!(list)
  return list if list.size <= 1
  0.upto(list.size - 1) do |i|
    (list.size - 1).downto(i + 1) do |j|
      if list[j] < list[j - 1]
        list[j], list[j - 1] = list[j - 1], list[j]
      end
    end
  end
  list
end
```

```
Subjects:  4
Mutations: 103
Kills:     79
Runtime:   9.46s
Killtime:  8.62s
Overhead:  8.84%
Coverage:  76.70%
Alive:     24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```
-  if list.size <= 1
+  if list.size <= -1

-  if list.size <= 1
+  if list.size <= 2
```

```ruby
def sort_algorithm!(list)
  return list if list.size <= 1
  0.upto(list.size - 1) do |i|
    (list.size - 1).downto(i + 1) do |j|
      if list[j] < list[j - 1]
        list[j], list[j - 1] = list[j - 1], list[j]
      end
    end
  end
  list
end
```

```
Subjects:  4
Mutations: 103
Kills:     79
Runtime:   9.46s
Killtime:  8.62s
Overhead:  8.84%
Coverage:  76.70%
Alive:     24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```
-  if list.size <= 1
+  if list.size <= -1

-  if list.size <= 1
+  if list.size <= 2

-  if list.size <= 1
+  if list.size <= nil
```

```ruby
def sort_algorithm!(list)
  return list if list.size <= 1
  0.upto(list.size - 1) do |i|
    (list.size - 1).downto(i + 1) do |j|
      if list[j] < list[j - 1]
        list[j], list[j - 1] = list[j - 1], list[j]
      end
    end
  end
  list
end
```

```
Subjects:  4
Mutations: 103
Kills:     79
Runtime:   9.46s
Killtime:  8.62s
Overhead:  8.84%
Coverage:  76.70%
Alive:     24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```
-  if list.size <= 1
+  if list.size <= -1

-  if list.size <= 1
+  if list.size <= 2

-  if list.size <= 1
+  if list.size <= nil

-  if list.size <= 1
+  if list.size <= false
```

```ruby
def sort_algorithm!(list)
  return list if list.size <= 1
  0.upto(list.size - 1) do |i|
    (list.size - 1).downto(i + 1) do |j|
      if list[j] < list[j - 1]
        list[j], list[j - 1] = list[j - 1], list[j]
      end
    end
  end
  list
end
```

```
Subjects:  4
Mutations: 103
Kills:     79
Runtime:   9.46s
Killtime:  8.62s
Overhead:  8.84%
Coverage:  76.70%
Alive:     24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```ruby
def sort_algorithm!(list)
  return list if list.size <= 1
  0.upto(list.size - 1) do |i|
    (list.size - 1).downto(i + 1) do |j|
      if list[j] < list[j - 1]
        list[j], list[j - 1] = list[j - 1], list[j]
      end
    end
  end
  list
end
```

```
Subjects:   4
Mutations:  103
Kills:      79
Runtime:    9.46s
Killtime:   8.62s
Overhead:   8.84%
Coverage:   76.70%
Alive:      24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```
-   return(list)
+   return(nil)
```

```ruby
def sort_algorithm!(list)
  return list if list.size <= 1
  0.upto(list.size - 1) do |i|
    (list.size - 1).downto(i + 1) do |j|
      if list[j] < list[j - 1]
        list[j], list[j - 1] = list[j - 1], list[j]
      end
    end
  end
  list
end
```

```
Subjects:   4
Mutations:  103
Kills:      79
Runtime:    9.46s
Killtime:   8.62s
Overhead:   8.84%
Coverage:   76.70%
Alive:      24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```
-   return(list)
+   return(nil)


-   if list.size <= 1
-     return(list)
-   end
+   nil
```

```ruby
def sort_algorithm!(list)
  return list if list.size <= 1
  0.upto(list.size - 1) do |i|
    (list.size - 1).downto(i + 1) do |j|
      if list[j] < list[j - 1]
        list[j], list[j - 1] = list[j - 1], list[j]
      end
    end
  end
  list
end
```

```
Subjects:  4
Mutations: 103
Kills:     79
Runtime:   9.46s
Killtime:  8.62s
Overhead:  8.84%
Coverage:  76.70%
Alive:     24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```
-   return(list)
+   return(nil)


-   if list.size <= 1
-     return(list)
-   end
+   nil


-   if list.size <= 1
-     return(list)
-   end
```

```ruby
def sort_algorithm!(list)
  return list if list.size <= 1
  0.upto(list.size - 1) do |i|
    (list.size - 1).downto(i + 1) do |j|
      if list[j] < list[j - 1]
        list[j], list[j - 1] = list[j - 1], list[j]
      end
    end
  end
  list
end
```

```
Subjects:  4
Mutations: 103
Kills:     79
Runtime:   9.46s
Killtime:  8.62s
Overhead:  8.84%
Coverage:  76.70%
Alive:     24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```ruby
def unsort_algorithm!(list)
  return list.shuffle unless list.empty?
end
```

```
Subjects:  4
Mutations: 103
Kills:     79
Runtime:   9.46s
Killtime:  8.62s
Overhead:  8.84%
Coverage:  76.70%
Alive:     24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```
-   unless list.empty?
+   unless nil
```

```ruby
def unsort_algorithm!(list)
  return list.shuffle unless list.empty?
end
```

```
Subjects:  4
Mutations: 103
Kills:     79
Runtime:   9.46s
Killtime:  8.62s
Overhead:  8.84%
Coverage:  76.70%
Alive:     24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```
-   unless list.empty?
+   unless nil

-   unless list.empty?
+   unless !list.emtpy?
```

```ruby
def unsort_algorithm!(list)
  return list.shuffle unless list.empty?
end
```

```
Subjects:   4
Mutations:  103
Kills:      79
Runtime:    9.46s
Killtime:   8.62s
Overhead:   8.84%
Coverage:   76.70%
Alive:      24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```
-   unless list.empty?
+   unless nil

-   unless list.empty?
+   unless !list.emtpy?

-   unless list.empty?
+   unless false
```

```ruby
def unsort_algorithm!(list)
  return list.shuffle unless list.empty?
end
```

```
Subjects:  4
Mutations: 103
Kills:     79
Runtime:   9.46s
Killtime:  8.62s
Overhead:  8.84%
Coverage:  76.70%
Alive:     24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```
-   unless list.empty?
+   unless nil

-   unless list.empty?
+   unless !list.emtpy?

-   unless list.empty?
+   unless false

-   unless list.empty?
+   if list.empty?
```

```ruby
def unsort_algorithm!(list)
  return list.shuffle unless list.empty?
end
```

```
Subjects:  4
Mutations: 103
Kills:     79
Runtime:   9.46s
Killtime:  8.62s
Overhead:  8.84%
Coverage:  76.70%
Alive:     24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```ruby
def unsort_algorithm!(list)
  return list.shuffle unless list.empty?
end
```

```
Subjects:  4
Mutations: 103
Kills:     79
Runtime:   9.46s
Killtime:  8.62s
Overhead:  8.84%
Coverage:  76.70%
Alive:     24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```
-   return(list.shuffle)
+   return(list)
```

```ruby
def unsort_algorithm!(list)
  return list.shuffle unless list.empty?
end
```

```
Subjects:  4
Mutations: 103
Kills:     79
Runtime:   9.46s
Killtime:  8.62s
Overhead:  8.84%
Coverage:  76.70%
Alive:     24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```
-   return(list.shuffle)
+   return(list)

-   return(list.shuffle)
+   list.shuffle
```

```ruby
def unsort_algorithm!(list)
  return list.shuffle unless list.empty?
end
```

```
Subjects:  4
Mutations: 103
Kills:     79
Runtime:   9.46s
Killtime:  8.62s
Overhead:  8.84%
Coverage:  76.70%
Alive:     24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```
-   return(list.shuffle)
+   return(list)

-   return(list.shuffle)
+   list.shuffle

-   return(list.shuffle)
+   return(nil)
```

```ruby
def unsort_algorithm!(list)
  return list.shuffle unless list.empty?
end
```

```
Subjects:  4
Mutations: 103
Kills:     79
Runtime:   9.46s
Killtime:  8.62s
Overhead:  8.84%
Coverage:  76.70%
Alive:     24
```

$ mutant --rspec -I lib -r fancy_algorithm FancyAlgorithm

```
-   return(list.shuffle)
+   return(list)

-   return(list.shuffle)
+   list.shuffle

-   return(list.shuffle)
+   return(nil)

-   unless list.empty?
-     return(list.shuffle)
-   end
+   nil
```

```ruby
def unsort_algorithm!(list)
  return list.shuffle unless list.empty?
end
```

```
Subjects:  4
Mutations: 103
Kills:     79
Runtime:   9.46s
Killtime:  8.62s
Overhead:  8.84%
Coverage:  76.70%
Alive:     24
```

- Try it by yourself
- clone the repo
- play around
- write specs to kill the mutants
- run mutant in other projects
- create issues
- help improve the documentation
- mutate all the things

example project (including this slides):
https://github.com/DonSchado/colognerb-on-mutant

in addition:
https://github.com/DonSchado/bob-the-mutant

obviously:
https://github.com/mbj/mutant

for reference:
http://slid.es/markusschirp/mutation-testing/

http://solnic.eu/2013/01/23/mutation-testing-with-mutant.html

rails?
https://github.com/mockdeep/mutant-rails