

Walter Blair

Assignment 2

No partner, sources cited below

```
walter@walter-Gazelle: ~/Desktop/Fall2017/602/assignment2/ruby-calisthenics
walter@walter-Gazelle: ~/Desktop/Fall2017/602/assign... x walter@walter-Gazelle: ~/Desktop/Fall2017/602/assign... x + v

#attr_accessor_with_history
  should remember history separately for each instance [30 points]
  when a symbol is passed [10 points]
    should define getter and setter [5 points]
    setter should return value set to [5 points]
    should work if getter used first [10 points]
    should work if setter used first [20 points]
    should remember values [10 points]
  when a string is passed [10 points]
    should define getter and setter [5 points]
    setter should return value set to [5 points]
    should work if getter used first [10 points]
    should work if setter used first [20 points]
    should remember values [10 points]

Dessert
  cake
    should have 400 calories [10 points]
    should be named cake [10 points]
    should be delicious [10 points]
    should not be healthy [10 points]
  apple
    should be delicious [10 points]
    should be healthy [10 points]
  can set
    calories [10 points]
    name [10 points]

JellyBean
  when non-licorice
    should contain 5 calories [5 points]
    should be named vanilla jelly bean [5 points]
    should be delicious [5 points]
  when licorice
    should not be delicious [5 points]

palindrome detection
  should work for simple strings [10 points]
  should be case-insensitive [10 points]
  should ignore nonword characters [10 points]

word count
  should return a hash [5 points]
  works on simple strings [10 points]
  ignores punctuation [5 points]
  works on the empty string [10 points]
  ignores leading whitespace [10 points]
  ignores embedded whitespace [10 points]

anagram grouping
  for "scream cars for four scar creams" [10 points]
  sanity checks
    should work on the empty string [5 points]
    should return an array of arrays for nonempty string [5 points]

RockPaperScissors
  should raise NoSuchStrategyError if strategy isn't R, P, or S [10 points] (PENDING: No reason given)
```

fun_with_strings

```
module FunWithStrings
  # Reference - https://stackoverflow.com/questions/1634750/ruby-function-to-remove-all-white-spaces
  def palindrome?
    return self.downcase.gsub(/[^a-zA-z]/, "") ==
      self.downcase.reverse.gsub(/[^a-zA-z]/, "")
  end
  def count_words
    wordCount = Hash.new
    wordList = self.downcase.gsub(/[^a-zA-z\s+]/, "").split()
    wordList.each do |word|
      if wordCount[word].nil?
        wordCount[word] = 1
      else
        wordCount[word] += 1
      end
    end
    return wordCount
  end
  def anagram_groups
    wordList = self.split()
    rtnList = Array.new
    wordList.each do |word|
      group = Array.new
      wordList.each do |w|
        if word.chars.sort == w.chars.sort
          group.push(w)
        end
      end
      rtnList.push(group)
    end
    return rtnList
  end
end

# make all the above functions available as instance methods on Strings:

class String
  include FunWithStrings
end
module FunWithStrings
  # Reference - https://stackoverflow.com/questions/1634750/ruby-function-to-remove-all-white-spaces
  def palindrome?
    return self.downcase.gsub(/[^a-zA-z]/, "") ==
      self.downcase.reverse.gsub(/[^a-zA-z]/, "")
  end
  def count_words
```

```

wordCount = Hash.new
wordList = self.downcase.gsub(/[^a-zA-z\s+]/, "").split()
wordList.each do |word|
  if wordCount[word].nil?
    wordCount[word] = 1
  else
    wordCount[word] += 1
  end
end
return wordCount

```

```

end
def anagram_groups
  wordList = self.split()
  rtnList = Array.new
  wordList.each do |word|
    group = Array.new
    wordList.each do |w|
      if word.chars.sort == w.chars.sort
        group.push(w)
      end
    end
    rtnList.push(group)
  end
  return rtnList
end
end

```

make all the above functions available as instance methods on Strings:

```

class String
  include FunWithStrings
end

```

dessert

```
class Dessert
```

```
  attr_accessor :name  
  attr_accessor :calories
```

```
  def initialize(name, calories)  
    @name = name  
    @calories = calories  
  end  
  def healthy?  
    return @calories < 200  
  end  
  def delicious?  
    return true  
  end  
end
```

```
class JellyBean < Dessert
```

```
  attr_accessor :flavor  
  
  def initialize(flavor)  
    @flavor = flavor  
    @calories = 5  
    @name = flavor + " jelly bean"  
  end  
  def delicious?  
    return @flavor != "licorice"  
  end  
end
```

attr_accessor_with_history

Reference - <https://web.stanford.edu/~ouster/cgi-bin/cs142-winter15/classEval.php>

Old Reference - <https://stackoverflow.com/questions/9311347/using-instance-variables-in-class-methods-ruby>

Old Reference - <https://stackoverflow.com/questions/19898574/getting-nameerror-format-is-not-allowed-as-an-instance-variable-name-when-tes>

Old Reference - <https://stackoverflow.com/questions/7693877/rails-variable-as-part-of-method-name?rq=1>

Solution Reference - https://mauricio.github.io/2009/06/04/understanding-class_eval-module_eval-and-instance_eval.html

```
class Class
```

```
  def attr_accessor_with_history(attr_name)
    attr_name = attr_name.to_s # make sure it's a string
    attr_reader attr_name # create the attribute's getter
    attr_reader attr_name+"_history" # create bar_history getter
    class_eval %Q{
```

```
      # This isn't setting the attr_name declared above
```

```
      def #{attr_name}=( val )
        if #{attr_name}_history.nil?
          @#{attr_name}_history = Array.new
        end
        @#{attr_name}_history.push(@attr_name)
        @attr_name = val
      end
```

```
      # This is getting instance @bar - is it overriding class @bar?
```

```
      def #{attr_name}
        @attr_name
      end
```

```
      # This is getting instance @bar - is it overriding class @bar?
```

```
      def #{attr_name}_history
        @#{attr_name}_history
      end
```

```
    }
```

```
  end
```

```
end
```