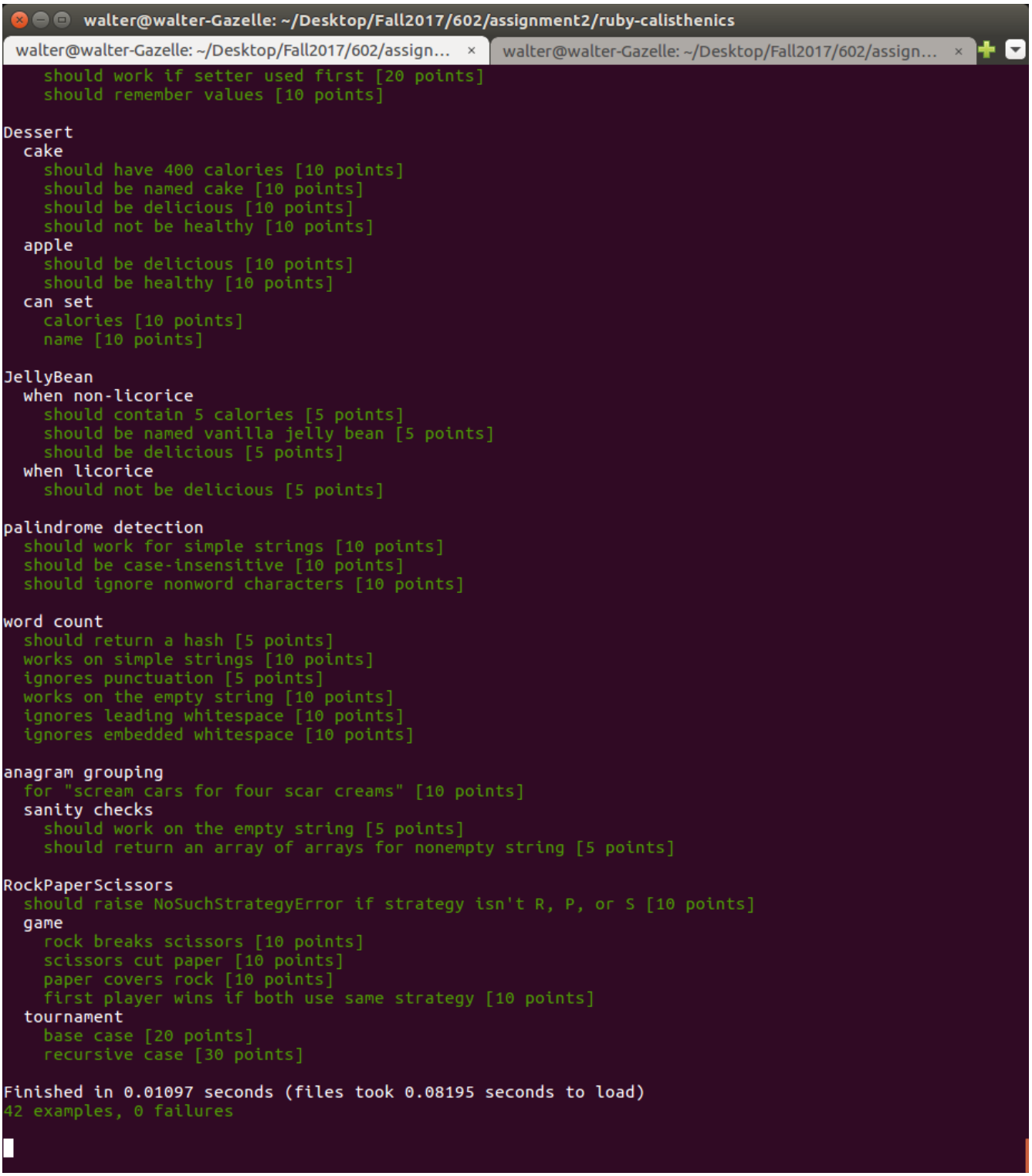
Walter Blair
Assignment 2
No partner, sources cited below



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
endmodule 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
```

rock_paper_scissors.rb

```
class RockPaperScissors

  # Exceptions this class can raise:
  class NoSuchStrategyError < StandardError ; end

  def self.winner(player1, player2)
    valid_moves = ['R', 'S', 'P']
    if valid_moves.include? player1[1] and valid_moves.include? player2[1]
       if [0, -1, 2].include? (valid_moves.index(player1[1]) - valid_moves.index(player2[1]))
          return player1
       else
          return player2
       end
    else
       # Reference - http://blog.honeybadger.io/a-beginner-s-guide-to-exceptions-in-ruby/
       raise NoSuchStrategyError.new("Strategy must be one of R,P,S")
    end
  end

  def self.tournament_winner(tournament)
    # Reference - https://stackoverflow.com/questions/18338495/iterate-every-two-elements-in-ruby-for-
loop
    tournament.each do |x, y|
       if tournament[0][0].class == String and tournament[1][0].class == String
          return self.winner(tournament[0], tournament[1])
       else
          return self.winner(self.tournament_winner(x), self.tournament_winner(y))
       end
    end
  end

end
```