**Ejercicio:** Ejercicio 7   
def combinar\_anagramas   
  
end   
# input: ['cars', 'for', 'potatoes', 'racs', 'four', 'scar', 'creams', 'scream']   
# output: [ ["cars", "racs", "scar"],   
# ["four"],   
# ["for"],   
# ["potatoes"],   
# ["creams", "scream"] ]

**Puntos:**

**10.0**

**Resultado:**

anagrams  
should handle an empty array [2 points]  
should handle the most simple, single-character case with no repeats [2 points] (FAILED)  
should handle the most simple, single-character case with capital letters but no repeats [2 points] (FAILED)  
should handle a set with no repeated anagrams and words with distinct letters [10 points] (FAILED)  
should handle a set with no repeated anagrams, but words with similar letters [2 points]  
should handle non-anagrams that have all but one letters the same [2 points]  
should handle a complicated case of some repeated anagrams, but no duplicates [25 points] (FAILED)  
should handle a complicated case of some repeated anagrams, but no duplicates [25 points] (FAILED)  
should handle a simple case of some repeated single-character inputs [2 points] (FAILED)  
should handle a complicated case of some repeated anagrams, with duplicate inputs [7 points] (FAILED)  
should handle a complicated case of some repeated anagrams, with duplicate inputs [7 points] (FAILED)  
should treat single-character capital letters as equal [2 points]  
should handle repeated single-character inputs, and treat captial letters as anagrams, while preserving case in the output [2 points] (FAILED)  
should treat two identical words with different cases as the same, and preserve case in output [2 points]  
should treat many identical words with different cases as the same, and preserve case in output [8 points] (FAILED)  
  
Failures:  
  
1) anagrams should handle the most simple, single-character case with no repeats [2 points]  
Failure/Error: result.length.should == 3  
expected: 3  
got: 2 (using ==)  
# /var/www/tallerucb/lib/ejercicios/17.rb:11:in `block (2 levels) in '  
# ./lib/rspec\_runner.rb:36:in `block in run\_rspec'  
# ./lib/rspec\_runner.rb:32:in `run\_rspec'  
# ./lib/rspec\_runner.rb:23:in `run'  
# ./lib/graders/weighted\_rspec\_grader.rb:6:in `grade!'  
# grade:31:in `

'  
  
2) anagrams should handle the most simple, single-character case with capital letters but no repeats [2 points]  
Failure/Error: result.length.should == 3  
expected: 3  
got: 2 (using ==)  
# /var/www/tallerucb/lib/ejercicios/17.rb:18:in `block (2 levels) in '  
# ./lib/rspec\_runner.rb:36:in `block in run\_rspec'  
# ./lib/rspec\_runner.rb:32:in `run\_rspec'  
# ./lib/rspec\_runner.rb:23:in `run'  
# ./lib/graders/weighted\_rspec\_grader.rb:6:in `grade!'  
# grade:31:in `'  
  
3) anagrams should handle a set with no repeated anagrams and words with distinct letters [10 points]  
Failure/Error: result.length.should == 4  
expected: 4  
got: 2 (using ==)  
# /var/www/tallerucb/lib/ejercicios/17.rb:25:in `block (2 levels) in '  
# ./lib/rspec\_runner.rb:36:in `block in run\_rspec'  
# ./lib/rspec\_runner.rb:32:in `run\_rspec'  
# ./lib/rspec\_runner.rb:23:in `run'  
# ./lib/graders/weighted\_rspec\_grader.rb:6:in `grade!'  
# grade:31:in `'  
  
4) anagrams should handle a complicated case of some repeated anagrams, but no duplicates [25 points]  
Failure/Error: result.length.should == 5  
expected: 5  
got: 2 (using ==)  
# /var/www/tallerucb/lib/ejercicios/17.rb:51:in `block (2 levels) in '  
# ./lib/rspec\_runner.rb:36:in `block in run\_rspec'  
# ./lib/rspec\_runner.rb:32:in `run\_rspec'  
# ./lib/rspec\_runner.rb:23:in `run'  
# ./lib/graders/weighted\_rspec\_grader.rb:6:in `grade!'  
# grade:31:in `'  
  
5) anagrams should handle a complicated case of some repeated anagrams, but no duplicates [25 points]  
Failure/Error: result.length.should == 4  
expected: 4  
got: 2 (using ==)  
# /var/www/tallerucb/lib/ejercicios/17.rb:58:in `block (2 levels) in '  
# ./lib/rspec\_runner.rb:36:in `block in run\_rspec'  
# ./lib/rspec\_runner.rb:32:in `run\_rspec'  
# ./lib/rspec\_runner.rb:23:in `run'  
# ./lib/graders/weighted\_rspec\_grader.rb:6:in `grade!'  
# grade:31:in `'  
  
6) anagrams should handle a simple case of some repeated single-character inputs [2 points]  
Failure/Error: result.length.should == 4  
expected: 4  
got: 2 (using ==)  
# /var/www/tallerucb/lib/ejercicios/17.rb:65:in `block (2 levels) in '  
# ./lib/rspec\_runner.rb:36:in `block in run\_rspec'  
# ./lib/rspec\_runner.rb:32:in `run\_rspec'  
# ./lib/rspec\_runner.rb:23:in `run'  
# ./lib/graders/weighted\_rspec\_grader.rb:6:in `grade!'  
# grade:31:in `'  
  
7) anagrams should handle a complicated case of some repeated anagrams, with duplicate inputs [7 points]  
Failure/Error: result.length.should == 5  
expected: 5  
got: 2 (using ==)  
# /var/www/tallerucb/lib/ejercicios/17.rb:72:in `block (2 levels) in '  
# ./lib/rspec\_runner.rb:36:in `block in run\_rspec'  
# ./lib/rspec\_runner.rb:32:in `run\_rspec'  
# ./lib/rspec\_runner.rb:23:in `run'  
# ./lib/graders/weighted\_rspec\_grader.rb:6:in `grade!'  
# grade:31:in `'  
  
8) anagrams should handle a complicated case of some repeated anagrams, with duplicate inputs [7 points]  
Failure/Error: result.length.should == 3  
expected: 3  
got: 2 (using ==)  
# /var/www/tallerucb/lib/ejercicios/17.rb:79:in `block (2 levels) in '  
# ./lib/rspec\_runner.rb:36:in `block in run\_rspec'  
# ./lib/rspec\_runner.rb:32:in `run\_rspec'  
# ./lib/rspec\_runner.rb:23:in `run'  
# ./lib/graders/weighted\_rspec\_grader.rb:6:in `grade!'  
# grade:31:in `'  
  
9) anagrams should handle repeated single-character inputs, and treat captial letters as anagrams, while preserving case in the output [2 points]  
Failure/Error: result.length.should == 4  
expected: 4  
got: 2 (using ==)  
# /var/www/tallerucb/lib/ejercicios/17.rb:92:in `block (2 levels) in '  
# ./lib/rspec\_runner.rb:36:in `block in run\_rspec'  
# ./lib/rspec\_runner.rb:32:in `run\_rspec'  
# ./lib/rspec\_runner.rb:23:in `run'  
# ./lib/graders/weighted\_rspec\_grader.rb:6:in `grade!'  
# grade:31:in `'  
  
10) anagrams should treat many identical words with different cases as the same, and preserve case in output [8 points]  
Failure/Error: result.length.should == 3  
expected: 3  
got: 2 (using ==)  
# /var/www/tallerucb/lib/ejercicios/17.rb:105:in `block (2 levels) in '  
# ./lib/rspec\_runner.rb:36:in `block in run\_rspec'  
# ./lib/rspec\_runner.rb:32:in `run\_rspec'  
# ./lib/rspec\_runner.rb:23:in `run'  
# ./lib/graders/weighted\_rspec\_grader.rb:6:in `grade!'  
# grade:31:in `'  
  
Finished in 0.02073 seconds  
15 examples, 10 failures  
  
Failed examples:  
  
rspec /var/www/tallerucb/lib/ejercicios/17.rb:8 # anagrams should handle the most simple, single-character case with no repeats [2 points]  
rspec /var/www/tallerucb/lib/ejercicios/17.rb:15 # anagrams should handle the most simple, single-character case with capital letters but no repeats [2 points]  
rspec /var/www/tallerucb/lib/ejercicios/17.rb:22 # anagrams should handle a set with no repeated anagrams and words with distinct letters [10 points]  
rspec /var/www/tallerucb/lib/ejercicios/17.rb:48 # anagrams should handle a complicated case of some repeated anagrams, but no duplicates [25 points]  
rspec /var/www/tallerucb/lib/ejercicios/17.rb:55 # anagrams should handle a complicated case of some repeated anagrams, but no duplicates [25 points]  
rspec /var/www/tallerucb/lib/ejercicios/17.rb:62 # anagrams should handle a simple case of some repeated single-character inputs [2 points]  
rspec /var/www/tallerucb/lib/ejercicios/17.rb:69 # anagrams should handle a complicated case of some repeated anagrams, with duplicate inputs [7 points]  
rspec /var/www/tallerucb/lib/ejercicios/17.rb:76 # anagrams should handle a complicated case of some repeated anagrams, with duplicate inputs [7 points]  
rspec /var/www/tallerucb/lib/ejercicios/17.rb:89 # anagrams should handle repeated single-character inputs, and treat captial letters as anagrams, while preserving case in the output [2 points]  
rspec /var/www/tallerucb/lib/ejercicios/17.rb:102 # anagrams should treat many identical words with different cases as the same, and preserve case in output [8 points]