For each of the student submissions below, what feedback would you give to help them improve their submission, with a goal of eventually emulating the instructor solution (provided below)? Assume that the student will be able to resubmit their assignment after reviewing this feedback.

Instructor solution:

```ruby
def combine_anagrams(words)
    words.group_by { |word| word.downcase.chars.sort }.values
end
```

Please read all the submissions before beginning to give feedback. Please don't move on to the next page until you have completed all the questions on this page.

Submission 1

```ruby
def combine_anagrams(words)
  results = Array.new
  words.each do |e|
    found = 0
    results.each_index do |i|
      if (results[i].first.downcase.split(//).sort.join == e.downcase.split(//).sort.
        results[i].push(e)
        found = 1
      end
    end
    results.push([e]) if (found == 0)
  end
  return results
end
```

(For our reference: 30.4 604)

*

It looks like found is only set to 0/1, couldn't it just be a boolean?
It looks like you make the same calls to downcase + split + sort + join multiple times. Can you think of a way to make that drier?

Submission 2

```ruby
def combine_anagrams(words)
  words.group_by { |word| word.downcase.chars.sort }.values
end
```

(For our reference: 6.8 472)

*

None; your solution is about as good as it gets

Submission 3

```ruby
def combine_anagrams(words)
  hash = {}
  words.each do |word|
    sort_word = word.downcase.chars.sort.join
    hash[sort_word] ||= []
    (hash[sort_word] << word)
  end
  return hash.values
end
```

(For our reference: 10.9 610)

*

It looks like you make a hash and mutate it over the course of your loop. Can you think of a functional way to do the same thing? (Hint: #group_by)

Submission 4

```ruby
def combine_anagrams(words)
  ar = Array.new
  words.map { |unique| unique.chars.sort.join.downcase }.uniq.map do |comp|
    ar.push(words.select { |element| (element.chars.sort.join.downcase == comp) })
  end
  print(ar)
end
```

(For our reference: 24.5 658)

*

It looks like you print(ar) when the problem asked to return ar.
It looks like you make multiple calls to chars.sort.join.downcase. Can you DRY the code up?

Submission 5

```
def combine_anagrams(words)
  words.group_by { |word| word.downcase.chars.sort }.values
end
```

(For our reference: 6.8 524)

*

None; your solution is about as good as it gets

Submission 6

```
def combine_anagrams(words)
  hash = {}
  words.each do |word|
    letters = word.downcase.chars.sort.join
    if hash.has_key?(letters) then
      hash[letters] += [word]
    else
      hash[letters] = [word]
    end
  end
  hash.values
end
```

(For our reference: 10.9 368)

*

It looks like you make a hash and mutate it over the course of your loop. Can you think of a functional way to do the same thing? (Hint: #group_by)

* Indicates Response Required

1 / 2

For each of the student submissions, rate the autogenerated hints on a scale of 1 to 5.

5 - Highly relevant: covers a key concept that should be improved.
4 - Somewhat relevant: helpful in a lesser way.
3 - Neutral: not helpful.
2 - Irrelevant: this hint is irrelevant
1 - Harmful: this hint is completely wrong and harmful

Please also comment on the degree to which these hints capture the feedback you gave in the previous step.

Submission 1

```ruby
def combine_anagrams(words)
  results = Array.new
  words.each do |e|
    found = 0
    results.each_index do |i|
      if (results[i].first.downcase.split(//).sort.join == e.downcase.split(//).sort.
        results[i].push(e)
        found = 1
      end
    end
    results.push([e]) if (found == 0)
  end
  return results
end
```

(For our reference: 30.4 604)

Here's the feedback you gave for this submission:

It looks like found is only set to 0/1, couldn't it just be a boolean? It looks like you make the same calls to downcase + split + sort + join multiple times. Can you think of a way to make that drier?

*** Here's the feedback the autograder gave:**

To improve your style, consider...

...not using a call to new.	★★★☆☆
...not using a call to first.	★★★☆☆
...not using a call to join.	★★★★☆

*** Comment here on the degree to which degree these hints capture the feedback you gave this submission.**

The autograder only made a comment on getting rid of one call to join; I made a comment on getting rid of the entire

Submission 2

```ruby
def combine_anagrams(words)
  words.group_by { |word| word.downcase.chars.sort }.values
end
```

(For our reference: 6.8 472)

Here's the feedback you gave for this submission:

None; your solution is about as good as it gets

*** Here's the feedback the autograder gave:**

No feedback generated ★★★★★

*** Comment here on the degree to which degree these hints capture the feedback you gave this submission.**

Same as mine

Submission 3

```
def combine_anagrams(words)
  hash = {}
  words.each do |word|
    sort_word = word.downcase.chars.sort.join
    hash[sort_word] ||= []
    (hash[sort_word] << word)
  end
  return hash.values
end
```

(For our reference: 10.9 610)

Here's the feedback you gave for this submission:

It looks like you make a hash and mutate it over the course of your loop. Can you think of a functional way to do the same thing? (Hint: #group_by)

*** Here's the feedback the autograder gave:**

To improve your style, consider...

...using a call to split.	★★☆☆☆
...not using a call to chars.	★☆☆☆☆

*** Comment here on the degree to which degree these hints capture the feedback you gave this submission.**

Not quite the same ; I don't see how this feedback will result in better code.

Submission 4

```
def combine_anagrams(words)
  ar = Array.new
  words.map { |unique| unique.chars.sort.join.downcase }.uniq.map do |comp|
    ar.push(words.select { |element| (element.chars.sort.join.downcase == comp) })
  end
  print(ar)
end
```

(For our reference: 24.5 658)

Here's the feedback you gave for this submission:

It looks like you print(ar) when the problem asked to return ar. It looks like you make multiple calls to chars.sort.join.downcase. Can you DRY the code up?

*** Here's the feedback the autograder gave:**

To improve your style, consider...

...using a call to each.	★★★★★
...using a call to uniq!.	★★★☆☆
...using a call to find_all.	★★★☆☆
...not using a method that produces hashes.	★★★★☆
...not using a call to print.	★★★★★

*** Comment here on the degree to which degree these hints capture the feedback you gave this submission.**

I mentioned the call to print, but none of the other feedback.

Submission 5

```
def combine_anagrams(words)
  words.group_by { |word| word.downcase.chars.sort }.values
end
```

(For our reference: 6.8 524)

Here's the feedback you gave for this submission:

None; your solution is about as good as it gets

*** Here's the feedback the autograder gave:**

No feedback generated	★★★★★

*** Comment here on the degree to which degree these hints capture the feedback you gave this submission.**

Same as mine

Submission 6

```
def combine_anagrams(words)
```

```
  hash = {}
  words.each do |word|
    letters = word.downcase.chars.sort.join
    if hash.has_key?(letters) then
      hash[letters] += [word]
    else
      hash[letters] = [word]
    end
  end
  hash.values
end
```

(For our reference: 10.9 368)

Here's the feedback you gave for this submission:

It looks like you make a hash and mutate it over the course of your loop. Can you think of a functional way to do the same thing? (Hint: #group_by)

*** Here's the feedback the autograder gave:**

To improve your style, consider...

...using a call to each_char.	★★☆☆☆
...not using a call to chars.	★★☆☆☆
...not using a call to join.	★★☆☆☆
...not using a method that produces strings.	★★☆☆☆

*** Comment here on the degree to which degree these hints capture the feedback you gave this submission.**

Not the same as my results; didn't see how this feedback would improve the code.

* Indicates Response Required

formsite Pro Trial Report Abuse

Close