1 testas - factorialo skaiciavimui

```ruby
# frozen_string_literal: true
#Klase kuri apskaiciuoja skaicisu faktoriala
class Integer
  def fact
    #nuo 1 iki pasirinkto skaiciaus
    #Kiekviena iteracija skaicius yra padauginamas
    (1..self).reduce(:*) || 1
  end
end
#Nuskaitomas skaicius is consolines eilutes
require 'readline'
puts 'Input number:'
first_nr = gets
#Parasomas atsakymas skaicium kreipiantis i klases Integer Fact metoda.
puts first_nr.to_i.fact

```

Testo class

```ruby
require 'test/unit'
#importuoja testuojama faila
require './fourth'
class PalindromeTest < Test::Unit::TestCase


  def setup
  end
#test method
  def test_p_match
  #nurodom reiksme pradine
        first_nr = 5
        #apskaiciuoja factoriala
        first_nr.to_i.fact
        #daro lyginima ar skaiciaus 5 faktorialas lygus 120, jeigu lygus viskas gerai, jei nelygus paraso
        #po kablelio "Expected 120 but was {Tikra reiksme kokia buvo gauta is klases}
        assert( first_nr.to_i.fact.eql?(120), "Expected 120 but was #{first_nr.to_i.fact}" )
  end
end
```

Paleidus testa

```
C:\Users\audri\Desktop\ruby\RoR\2pd>ruby test.rb
Input number:
5
120
Loaded suite test
Started
.
Finished in 0.0017069 seconds.
---------------------------------------------------------------------------------------------------------------------

1 tests, 2 assertions, 0 failures, 0 errors, 0 pendings, 0 omissions, 0 notifications
100% passed
---------------------------------------------------------------------------------------------------------------------

585.86 tests/s, 1171.71 assertions/s

C:\Users\audri\Desktop\ruby\RoR\2pd>
```

2 testas “

Klases kuri testuojama kodas

```ruby
# frozen_string_literal: true
#Sukuriama nauja klase
class ArrayOfMultiples
#Aprasomas naujas self multiply metodas priimantis du skaicius
  def self.multiply(first_nr, second_nr)
  #atsakymo masyvas, kuris laiko skaiciu sandaugas
    answer = []
    sum = 0
    i = 0
      #While ciklas yra vykdomas kol i yra mazesnis uz pirma skaiciu, kiekvienos iteracijos metu i yra padidinamas
    while i < first_nr.to_i
    #Prie sum pridedamas antras skaicius kiekviena iteracija.
      sum += second_nr.to_i
      #kiekvienas sum pushinamas i array tokiu budu, gaunama kiekvieno skaiciaus sandauga
      answer.push(sum)
      i += 1
    end
     #atspausdinamas sandaugu masyvas
   # puts 'Array of multiples:'
    return answer
  end
end

#Nuskaitomi skaiciai is consolines eilutes
require 'readline'
puts 'Input first number:'
first_nr = gets
puts 'Input second number:'
second_nr = gets
#Kreipinys i klases multiply metoda perduodant pirma ir antra skaiciu
#Pirmas skaicius - Kiek kartu bus antras skaicius padaugintas
#Antras skaicius - Skaicius, kuri daugins.
puts ArrayOfMultiples.multiply(first_nr, second_nr)

```

Testo kodas

main.rb | second.rb | third.rb | test2.rb | test.rb

```ruby
require 'test/unit'
#importuoja testuojama faila
require './second'
class PalindromeTest < Test::Unit::TestCase


  def setup
  end
#test method
  def test_p_match
  #nurodom reiksme pradine
        first_nr = 2
        #nurodom reiksme pradine
        second_nr = 3
        #Kviecia metoda kitoj klasej
        answer = ArrayOfMultiples.multiply(first_nr, second_nr)
        #bnurodom kokio atsakymo tikimasi
        expected=[3,6]
        #Assert sulygina ar atsakymas ir tiketinas atsakymas vienodas, jei ne parodo kas skiriasi.
        assert( answer==expected, "Expected 3 and 6, but was #{answer}" )
  end

end
```

Paleidus testo.rb

```
C:\Users\audri\Desktop\ruby\RoR\2pd>ruby test2.rb
Input first number:
2
Input second number:
3
3
6
Loaded suite test2
Started
.
Finished in 0.001529 seconds.
---------------------------------------------------------------------------------------------------------------

1 tests, 1 assertions, 0 failures, 0 errors, 0 pendings, 0 omissions, 0 notifications
100% passed
---------------------------------------------------------------------------------------------------------------

654.02 tests/s, 654.02 assertions/s

C:\Users\audri\Desktop\ruby\RoR\2pd>
```