daftcode

RUBY ON RAILS 4 BEGINNERS

2. Obiektowość i organizacja

Praca domowa

```ruby
def sort_letters(string)
  string.chars.sort.join
end

def count_vowels(string)
  string.count('aeiouy')
end

def filter_odd(array)
  array.map(&:odd?)
end
```

```ruby
sort_letters('dcba') #=> 'abcd'
sort_letters('zyx') #=> 'xyz'

count_vowels('mmmm') #=> 0
count_vowels('super') #=> 2
count_vowels('super extra') #=> 4

filter_odd([1, 2, 3, 4, 5]) #=> [1, 3, 5]
filter_odd([6, 7, 9, 2, 6, 5]) #=> [7, 9, 5]
```

RUBY ON RAILS
BEGINNERS

Inject - wyjaśnienie

```
[1, 2, 3].inject { |acc, e| puts acc; acc * e }
# 1
# 2
# => 6

[1, 2, 3].inject(1) { |acc, e| puts acc; acc * e }
# 1
# 1
# 2
#=> 6
```

Konwencje

Case

```
variable_name = 9 # zmienna

method_name() # metoda

CONSTANT = 9 # stała

StandardError # klasa / moduł
```

! oraz ?

```ruby
object.save! # rzuca błędem
array.map! # modyfikuje array

object.save if object.valid?
# true / false
```

Wyrażenia

```ruby
if object == nil || object == false
  false
else
  true
end
```

```ruby
!!object
```

```ruby
if variable == nil || variable == false
  variable = 3
else
  variable
end
```

```ruby
variable ||= 3
```

RUBY ON RAILS
BEGINNERS

Klasy

RUBY ON RAILS
BEGINNERS

Definicja

```ruby
class Person

end
```

```ruby
person = Person.new
#=> #<Person:0x007fb24981ce70>

person.is_a? Person
#=> true
```

RUBY ON RAILS
BEGINNERS

Konstruktor

```ruby
class Person

  def initialize name
    @name = name
  end
end
```

```ruby
person = Person.new('Tomek')
#=> #<Person:0x007ff88b0c1580 @name="Tomek">

person.name
#=> NoMethodError
```

RUBY ON RAILS
BEGINNERS

Getters & setters

```ruby
class Person
  # attr_reader :name # getter
  # attr_writer :name # setter
  attr_accessor :name # oba

  def initialize name
    self.name = name
  end
end
```

```ruby
person = Person.new('Tomek')
#=> #<Person:0x007ff88b0c1580 @name="Tomek">

person.name #=> "Tomek"
person.name = 'Tomasz' #=> "Tomasz"
person.name #=> "Tomasz"
```

Metody instancyjne

```ruby
class Person
  attr_accessor :name

  def initialize name
    self.name = name
  end

  def greet
    "Hello, my name is #{name}"
  end
end
```

```ruby
person = Person.new('Tomek')
#=> #<Person:0x007ff88b0c1580 @name="Tomek">

person.greet
#=> "Hello, my name is Tomek"
```

Modyfikatory dostępu

```ruby
class Person
  attr_accessor :name

  def initialize name
    self.name = name
  end

  def greet
    "Hello, my name is #{name} #{smile}"
  end

  private

  def smile
    ':)'
  end
end
```

```ruby
person = Person.new('Tomek')
#=> #<Person:0x007ff88b0c1580 @name="Tomek">

person.greet
#=> "Hello, my name is Tomek :)"

person.smile
#=> NoMethodError
```

Metody i zmienne klasowe

```ruby
class Person
  attr_accessor :name
  @@count = 0

  def initialize name
    self.name = name
    @@count += 1
  end

  def self.count
    @@count
  end
end
```

```ruby
Person.count
#=> 0

person = Person.new('Tomek')
#=> #<Person:0x007ff88b0c1580 @name="Tomek">

Person.count
#=> 1
```

Dziedziczenie

```ruby
class Student < Person

end
```

```ruby
student = Student.new('Bartek')
#=> #<Student:0x007ff88b051d20 @name="Bartek">

student.name
#=> "Bartek"

student.is_a? Person
#=> true
```

Nadpisywanie

```ruby
class Student < Person
  attr_accessor :index

  def initialize name, index
    super(name)
    self.index = index
  end
end
```

```ruby
student = Student.new('Bartek', 123)
#=> #<Student:0x007ff88b051d20 @name="Bartek" @index=123>

student.index
#=> 123
```

Poszerzanie

```ruby
class Integer

  def factorial
    (1..self).inject(:*)
  end
end

3.factorial #=> 6

[1, 2, 3].map(&:factorial) #=> [1, 2, 6]
```

Przykład

```ruby
class CustomError < StandardError

  def message
    'OHMYGODITSONFIRE'
  end
end

begin
  raise CustomError
rescue CustomError => e
  e.message #=> "OHMYGODITSONFIRE"
end
```

RUBY ON RAILS
BEGINNERS

Moduły

Include

```ruby
module Talkable

  def welcome
    "Hello, I am #{signature}"
  end

  private

  def signature
    defined?(index) ? "#{name}##{index}" : name
  end
end
```

```ruby
class Person
  include Talkable
end

Person.new('Tom').welcome
#=> "Hello, I am Tom"

Student.new('Tom', 123).welcome
#=> "Hello, I am Tom#123"
```

Extend

```ruby
module Descriptable

  def description
    "This is a #{name.downcase}"
  end
end

class Person
  extend Descriptable
end
```

```ruby
Person.description
#=> "This is a person"

Student.description
#=> "This is a student"
```

Hooks

```ruby
module MyModule

  def self.included base
    puts "I must go, #{base} needs me!"
  end
end
```

```ruby
class Someone
  include MyModule
end
#I must go, Someone needs me!
#=> Someone
```

Nesting

```ruby
module MyModule
  class MyClass
  end
end

class MyModule::MyClass

end

::String # outer scope
```

Gemy

RUBY ON RAILS
BEGINNERS

Instalacja

```
$ gem install faker
Fetching: faker-1.9.1.gem (100%)
Successfully installed faker-1.9.1
Parsing documentation for faker-1.9.1
Installing ri documentation for faker-1.9.1
Done installing documentation for faker after 2 seconds
1 gem installed
```

Użycie

```ruby
require 'faker'

Faker::Name.name
#=> "Carl Toy IV"

Faker::Internet.email
#=> "edward@schultz.name"
```

Thanks!