Гладкоскок Максим КС31

Варіант 16

Теоретичні питання

1. Як оголосити рядок у Ruby?
2. Як у Ruby реалізувати власний виняток?
3. Як працює метод super у Ruby для виклику методів батьківського класу?
4. Як створити і використовувати класи і модулі в Ruby?

Практичні завдання

1. Напишіть метод, який обчислює кількість елементів у масиві, більших за певне значення.
2. Напишіть метод, який рахує кількість входжень символу в рядку

1. У Ruby можно оголосити рядок, використовуючи одинарні лапки (') або подвійні лапки (").
   Example = “Hello World!”
   Example = ’Hello World’
2.

   1. Створюємо свій клас винятку, який успадковує StandardError.
   2. Викликаємо (raise) виняток у разі виникнення помилки.
   3. Обробляємо (rescue) виняток, щоб не допустити аварійного завершення програми.

```
class AgeError < StandardError
end

def check_age(age)
  raise AgeError, "Вік має бути не менше 18!" if age < 18
  puts "Вік прийнятний: #{age}"
end

begin
  check_age(16)
rescue AgeError => e
  puts "Перехоплено: #{e.message}"
end
```

   Result:
   Перехоплено: Вік має бути не менше 18!

   Якщо змінити check_age(16) на check_age(20):
   Вік прийнятний: 20

3. Метод super дозволяє нам викликати метод із батьківського класу, який має таку ж назву, як і в дочірньому класі. Це корисно, коли ми хочемо використовувати поведінку батьківського методу, але додати щось своє.

Уявімо, що у нас є два класи: Тварина (батьківський клас) і Собака (дочірній клас).

```
class Animal
  def speak
    puts "Тварина пронаунс: звук"
  end
end
class Dog < Animal
  def speak
    super
    puts "Собака гавкає: Гав гав"
  end
end
dog = Dog.new
dog.speak
```

Result:

Тварина каже: звук

Собака гавкає: Гав-гав

super бере частину поведінки з батьківського класу , потім додаємо нову частину в дочірньому класі.

4. Приклад 1

```
class Car
  def initialize(make)
    @make = make
  end

  def info
    puts "Авто марки: #{@make}"
  end
end

car = Car.new("BMW")
car.info
```

Car — це клас.
initialize — задає марку авто, коли створюється новий об'єкт.
info — метод, який виводить інформацію.

Приклад 2

```
module Greetable
  def greet
    puts "Привіт!"
  end
end

class Person
  include Greetable
end

person = Person.new
person.greet
```

Greetable — це модуль з методом greet.
include Greetable — додає метод greet до класу Person.

Клас: створює об'єкти (наприклад, авто).

Модуль: додає методи до класів (наприклад, "Привіт!" для людей).

1. Порахувати кількість елементів у масиві, більших за значення:
   Лістинг коду 1

```
def count_greater(arr, val)
    arr.select { |x| x > val }.size
end
puts count_greater([1, 5, 10, 15, 20], 10)
```

   Result: 2

2. Порахувати кількість входжень символу в рядку:
   Лістинг коду 1

```
def char_count(str, ch)
  str.chars.count(ch)
end

puts char_count("hello world", 'o')
```

   Result: 2