Харківський національний університет ім. В.Н. Каразіна

Факультет комп’ютерних наук

Кафедра електроніки та управляючих систем

ЗВІТ

З КОНТРОЛЬНОЇ РОБОТИ №2

дисципліни: «Мови прикладного програмування»

Виконав: студент групи ЗКС31

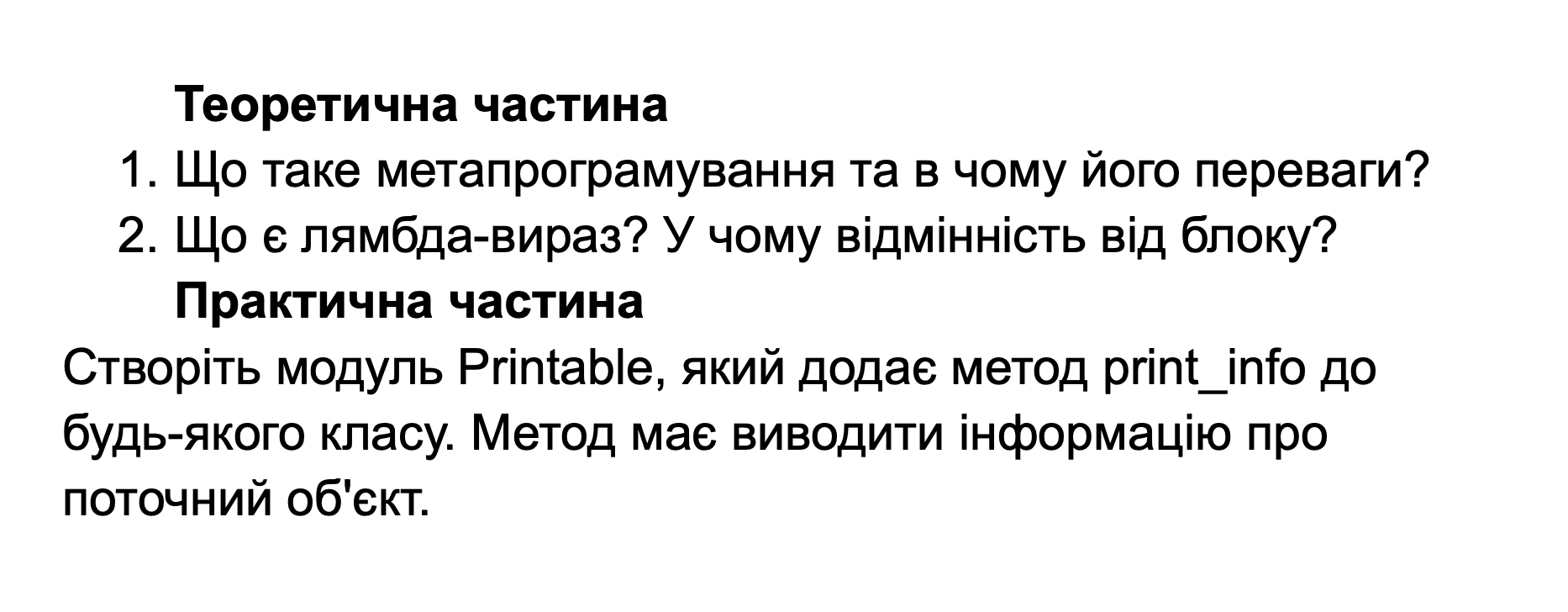
Маїло В.В.

Перевірив:

Паршенцев Б.В.

Харків

2023



**Теоретична частина**

1. **Метапрограмування** - це практика написання комп'ютерних програм, які можуть обробляти або генерувати інші програми як свої дані, або змінювати себе під час виконання. У Ruby метапрограмування часто асоціюється з динамічним визначенням методів, використанням **eval**, а також з використанням хуків класу, таких як **included**, **extended** і **method\_missing**. Переваги метапрограмування включають здатність писати більш гнучкі та адаптивні програми, зменшення дублювання коду, а також можливість створення потужних бібліотек і DSL (domain-specific languages).
2. **Лямбда-вираз** у Ruby - це анонімна функція, яка може бути збережена у змінній і передана як аргумент у інші функції. Лямбда в Ruby створюється за допомогою ключового слова **lambda** або через синтаксис **->**. Лямбда-вирази перевіряють кількість аргументів і повертають значення з останнього виразу всередині. Відмінності лямбди від блоку в Ruby:
   * **Контроль аргументів**: Лямбда строго перевіряє кількість аргументів, в той час як блоки більш гнучкі.
   * **Поведінка return**: В лямбді **return** повертає значення з лямбди, а в блоку **return** виходить з обгортаючого методу.

**Практична частина**

У цьому прикладі модуль Printable додає метод print\_info до класу User. Цей метод виводить інформацію про об'єкт, включаючи його клас та атрибути.

Лістинг 1:

module Printable

def print\_info

puts "Об'єкт класу #{self.class}, дані: #{self.inspect}"

end

end

class User

include Printable

attr\_accessor :name, :age

def initialize(name, age)

@name = name

@age = age

end

end

user = User.new("Volodymyr", 19)

user.print\_info