Харківський національний університет імені В.Н. Каразіна

Навчально-науковий інститут комп’ютерних наук та штучного інтелекту

Спеціальність 122 Комп’ютерні науки

Спеціалізація / освітня програма Інформаційні управляючі системи та технології

Семестр 1-й

Форма навчання: денна, заочна

Рівень вищої освіти (освітньо-кваліфікаційний рівень): бакалавр

Навчальна дисципліна: «Стек технологій програмування»

ЕКЗАМЕНАЦІЙНИЙ БІЛЕТ №9

1. Що таке method\_missing?Як його можна використовувати для динамічного виклику методів? (5 балів)

2. Якпрацюєоб’єкт File у Ruby? Які методи є ключовими для роботи з файлами? (5 балів)

3. Якреалізується паттерн "Спостерігач" (Observer) у Ruby? (5 балів)

4. ЩотакеMarshalуRuby?Длячого його використовують? (5 балів)

5. (Практичне завдання) Напишіть клас PrimeChecker, який перевіряє, чи є число простим. (20 балів

1. Method\_missing — це спеціальний метод у Ruby, який викликається тоді, коли об'єкт намагається звернутися до методу, який не визначений у його класі. Це дозволяє обробляти такі виклики динамічно.

Використання method\_missing дозволяє створювати динамічну поведінку, наприклад, генерувати методи "на льоту" або обробляти виклики методів із певними правилами.

class DynamicHandler

def method\_missing(method\_name, \*args, &block)

if method\_name.to\_s.start\_with?("handle\_")

action = method\_name.to\_s.split("\_", 2).last

puts "Handling action: #{action}, with arguments: #{args.join(', ')}"

else

super

end

end

end

obj = DynamicHandler.new

obj.handle\_task("Task 1") # Handling action: task, with arguments: Task 1

obj.undefined\_method # Помилка: NoMethodError

У цьому прикладі виклики методів, що починаються з handle\_, обробляються в method\_missing, а інші викликають стандартну помилку NoMethodError.

Цей підхід зручний для обробки викликів, де імена методів можуть бути динамічно створені.

2. Об'єкт File у Ruby використовується для роботи з файлами: їх створення, читання, запису та видалення. Щоб відкрити файл, застосовуються File.open, де вказується ім'я файлу і режим роботи: "r" — читання, "w" — запис (очищує або створює новий файл), "a" — додавання в кінець.

Наприклад:

File.open("example.txt", "w") { |file| file.puts "Hello, Ruby!" }

код записує текст у файл. Для читання використовуються File.read (весь вміст) або File.foreach (построчно):

puts File.read("example.txt")

Перевірити, чи файл існує, можна через File.exist?, а видалити — File.delete.

Наприклад:

if File.exist?("example.txt") File.delete("example.txt") end

File дозволяє працювати з інформацією про файли, як-от отримання розміру (File.size) або розширення (File.extname).

3. Паттерн "Спостерігач" у Ruby використовується, щоб один об'єкт (суб'єкт) міг сповіщати інші об'єкти (спостерігачі) про зміни свого стану. Для цього Ruby має модуль Observable. Суб'єкт підключає цей модуль і додає спостерігачів через метод add\_observer. Коли стан суб'єкта змінюється, він викликає changed, щоб позначити зміну, і notify\_observers, щоб повідомити всіх спостерігачів.

Наприклад, у класі суб'єкта визначається метод update\_state, який змінює стан і повідомляє спостерігачів. Спостерігачі повинні мати метод update, куди передається новий стан. Це дозволяє динамічно реагувати на зміни. Якщо не використовувати модуль Observable, можна самостійно створити список спостерігачів і викликати їх методи вручну при зміні стану.

4. Marshal у Ruby — це вбудований модуль, який дозволяє серіалізувати та десеріалізувати об'єкти. Серіалізація — це процес перетворення об'єкта в рядок байтів, який можна зберігати у файлі, передавати через мережу або використовувати іншим чином. Десеріалізація — це зворотний процес, коли об'єкт відновлюється з рядка байтів.

Для чого використовують Marshal?

1. Збереження стану об'єктів: можна зберегти об'єкт у файл і потім відновити його для подальшого використання.
2. Передача об'єктів між програмами: серіалізований об'єкт можна передати через мережу або інші канали.
3. Кешування: серіалізація дозволяє швидко зберігати об'єкти у вигляді даних для кешування.

Основні методи Marshal

1. Marshal.dump — серіалізує об'єкт в рядок або записує його у файл.
2. Marshal.load — відновлює об'єкт з рядка байтів або файлу.

Обмеження Marshal

1. Не всі об'єкти можна серіалізувати. Наприклад, об'єкти класів IO, Proc і деякі системні об'єкти не підтримуються.
2. Серіалізовані дані специфічні для Ruby, тому не можуть бути безпосередньо використані іншими мовами.

5.

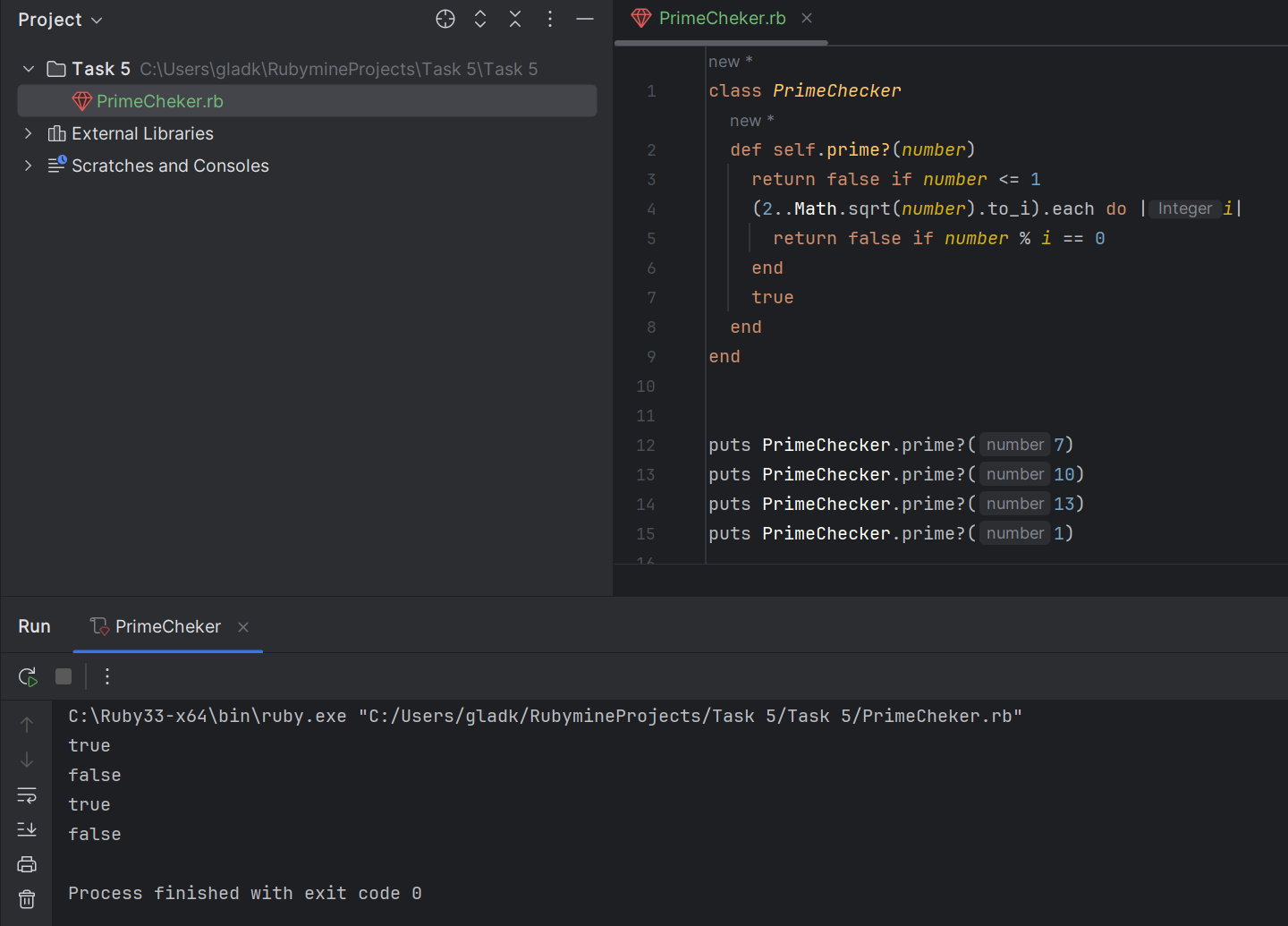


Рисунок 1 – Завдання №5