МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЗАПОРІЗЬКА ПОЛІТЕХНІКА»

Кафедра програмних засобів

Звіт

з лабораторної роботи №3

з дисципліни: «Крос-платформне програмування»

за темою: «Робота з файлами та директоріями»

Виконав:

ст. гр. КНТ-117 М. І. Басанець

Прийняв:

старший викладач Є. М. Федорченко

2020

1 МЕТА РОБОТИ

Вивчити основні можливості та принципи роботи з файлами та директоріями у Ruby.

2 зАВДАННЯ ДО РОБОТИ

1. У файлі надано цілочисельний масив. Необхідно зчитати елементи цього масиву з файлу ,поміняти місцями мінімальний і максимальний елементи масиву, а потім записати вже новий масив у інший файл.
2. У файлі надано рядок слів. Необхідно зчитати цей рядок з файлу, перемішати усі слова цього рядка у випадковому порядку, а потім записати вже новий рядок у інший файл.
3. Створити текстовий аналізатор який буде підраховувати кількість символів у файлі, кількість пробілів у файлі, кількість рядків у файлі, кількість абзаців у файлі, середнє число слів у реченні, середнє число речень у абзаці.
4. Створити програму за допомогою класу DIR , яка буде вираховувати розмір усіх файлів у вказаній директорії.

3 Результати виконання завдання

**Завдання 1**

Код програми:

input\_file = File.new("files/input\_task1.txt", "r")

array = input\_file.read().split().map(&:to\_i)

min\_element\_index = array.find\_index(array.min())

max\_element\_index = array.find\_index(array.max())

array[min\_element\_index], array[max\_element\_index] = array[max\_element\_index], array[min\_element\_index]

output\_file = File.new("files/output\_task1.txt", "w")

output\_file.write(array.join(" "))

Результат роботи програми зображено на рисунку 3.1

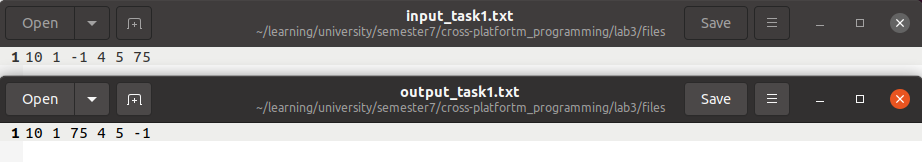


Рисунок 3.1 – Результати роботи програми №1

**Завдання 2**

Код програми:

input\_file = File.new("files/input\_task2.txt", "r")

words = input\_file.read().split()

words = words.shuffle()

output\_file = File.new("files/output\_task2.txt", "w")

output\_file.write(words.join(" "))

Результати роботи програми зображено на рисунку 3.2

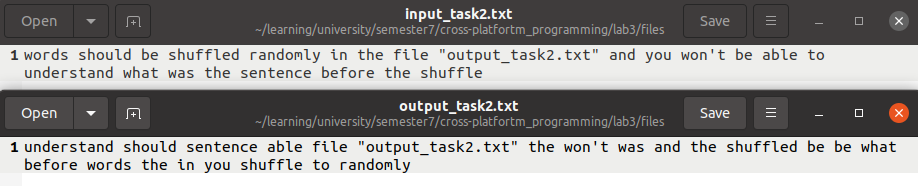


Рисунок 3.2 – Результати роботи програми 2

**Завдання 3**

Код програми:

input\_file = File.new("files/input\_task3.txt", "r")

text = input\_file.read()

paragraphes = text.lines().select{|line| line.strip().chomp().size() > 0}

sentences = text.split(/[.!?]+/).map(&:strip)

total\_characters = text.size()

total\_spaces = text.scan(/\ /).size()

total\_lines = text.lines().size()

total\_paragraphes = paragraphes.size()

avg\_words\_in\_sentence = sentences.map{|s| s.scan(/\w+/).size()}.sum() / sentences.size().to\_f()

avg\_sentences\_in\_paragraphes = paragraphes.map{|par| par.scan(/[.!?]+/).size()}.sum() / paragraphes.size().to\_f()

puts "Total characters in file: #{total\_characters}"

puts "Total spaces in file: #{total\_spaces}"

puts "Total lines in file: #{total\_lines}"

puts "Total paragraphes in file: #{total\_paragraphes}"

puts "Avg number of words in sentence: #{avg\_words\_in\_sentence}"

puts "Avg number of sentences in paragraphes: #{avg\_sentences\_in\_paragraphes}"

Результати роботи програми зображено на рисунку 3.3

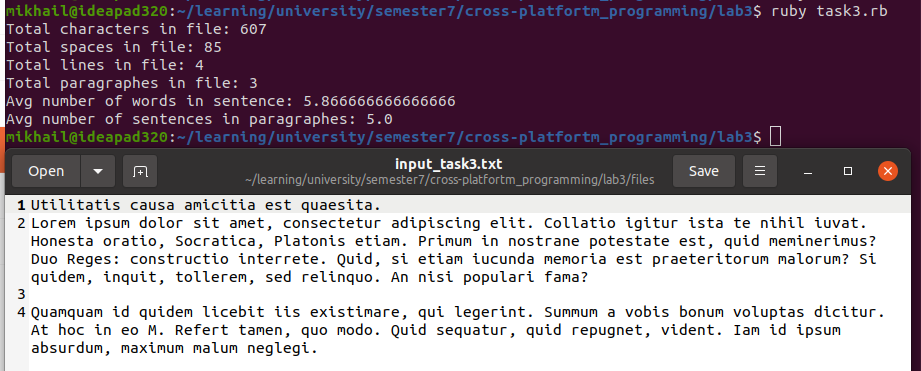


Рисунок 3.3 – Результати роботи програми 3

**Завдання 4**

Код програми:

def traverse\_entry (path)

if File.file?(path)

puts(path)

return File.size(path)

elsif File.directory?(path)

entries = Dir.glob(Dir.children(path).map() {|f| File.join(path, f)})

return entries.map() {|entry| traverse\_entry(entry)} .sum()

end

end

total\_size = traverse\_entry("files")

puts "Total size of the folder is #{total\_size}"

Результати роботи програми зображено на рисунку 3.4.

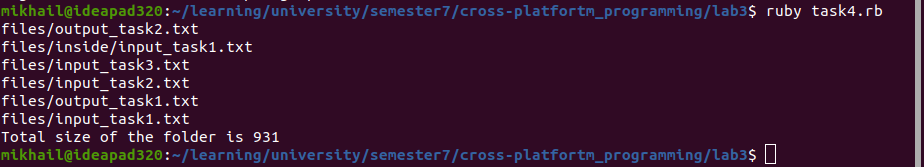


Рисунок 3.4 – Результати роботи програми 4

4 висновки

При виконанні даної лабораторної роботи на практиці засвоїв основні принципи та засади роботи з файлами та директоріями у Ruby.

1. Навіщо потрібен метод read?

Метод read зчитує файл у один рядок.

2. Який метод використовується для посимвольного зчитування?

Для посимвольного зчитування файла у Ruby використовується метод getc.

3. Як виконується запис у файл в Ruby?

Запис здійснюється за допомогою методів puts, print, write якщо файл відкритий і знаходиться в режимі запису або додавання.

4. Навіщо потрібен клас DIR?

DIR використовується для роботи з директоріями, також як і для роботи з файлами. Основна спрямованість класу Dir - надати користувачеві можливість запиту, обходу і фільтрації по директоріях файлової системи.

5. Як створити каталог за допомогою класу DIR?

mkrid

Dir.mkdir("/tmp/test")