

Міністерство освіти і науки України
Донецький національний університет імені Василя Стуса Факультет
інформаційних і прикладних технологій
Кафедра інформаційних технологій

ЗВІТ

з лабораторної роботи № 12
з дисципліни «Основи програмування»
на тему:
«Робота з текстовими файлами»

Виконав: студент гр. Б25_д/Ф3

Кручківський Ю.О.

Перевірив: доц. Бабаков Р. М.

Код до завдання:

```
import sys # імпортуємо модуль sys для керування стандартним виводом
import LabWork_9.src.solution1 as lab1 # імпортуємо першу частину
рішення 9-ї лабораторної роботи
import LabWork_9.src.solution2 as lab2 # імпортуємо другу частину
рішення 9-ї лабораторної роботи

def chek_number(number): # оголошуємо функцію для перевірки діапазону
числа
    return True if -1000 <= number <= 1000 else False # повертаємо
істину якщо число в межах від -1000 до 1000

def parse_array_text(raw_text): # оголошуємо функцію для парсингу одного
масиву з тексту
    array_chars = [0,0] # ініціалізуємо список для збереження індексів
дужок
    for i ,char in enumerate(raw_text): # проходимо по кожному символу
тексту з його індексом
        if char == "[": # якщо символ - відкриваюча дужка
            array_chars[0] = i # зберігаємо її індекс
        if char == "]": # якщо символ - закриваюча дужка
            array_chars[1] = i # зберігаємо її індекс
            break # виходимо з циклу після знаходження першої
закриваючої дужки

    array_text = raw_text[array_chars[0]+1:array_chars[1]:] # виділяємо
текст всередині дужок
    chars_array = array_text.split(",") # розділяємо текст на окремі
елементи по комі
    array = [] # ініціалізуємо порожній список для чисел
    for num in chars_array: # проходимо по кожному отриманому рядку з
числом
        num = int(num) # конвертуємо рядок у ціле число
        if not chek_number(num): # перевіряємо чи число входить у
допустимий діапазон
            raise ValueError("number not in range -1000 <= number <=
1000") # викидаємо помилку якщо число поза діапазоном
        array.append(num) # додаємо число до списку
    return array # повертаємо сформований список чисел

def parse_matrix_text(raw_text): # оголошуємо функцію для парсингу
матриць з тексту
    array_chars = [0,0] # ініціалізуємо список для індексів дужок
    array_2 = [] # ініціалізуємо список для збереження вкладених масивів
    for i ,char in enumerate(raw_text): # проходимо по тексту з
індексами
        if char == "[" and raw_text[i+1] != "[": # якщо знайдено початок
вкладеного масиву
            array_chars[0] = i # зберігаємо індекс початку
            if char == "]" and raw_text[i+1] != "]": # якщо знайдено кінець
вкладеного масиву
                array_chars[1] = i # зберігаємо індекс кінця

    array_2.append(parse_array_text(raw_text[array_chars[0]:array_chars[1]+1:
])) # парсимо вкладений масив та додаємо до списку
    return array_2 # повертаємо отриману матрицю

f = open("./config/output.txt", "w", encoding="utf-8") # відкриваємо
файл для запису результату
```

```
sys.stdout = f # перенаправляємо стандартний вивід у файл
sys.stderr = f # перенаправляємо вивід помилок у файл

with open("./config/array.txt", "r") as file: # відкриваємо файл з
вхідними даними для читання
    text = file.read() # зчитуємо весь вміст файлу у змінну

matrix = parse_matrix_text(text) # парсимо отриманий текст у матрицю
чисел

lab1.run_solution(matrix) # запускаємо перше рішення 9-ї лабораторної з
отриманою матрицею
lab2.run_solution(matrix) # запускаємо друге рішення 9-ї лабораторної з
отриманою матрицею
```

Висновки:

Під час виконання лабораторної роботи було створено рішення що обгортає код для запису в файл.