Міністерство освіти і науки України

Державний університет ,,Житомирська політехніка”

Кафедра ІПЗ

Група: ІПЗ-23-1

Програмування мовою Python

Лабораторна робота №6

«РОБОТА З ФАЙЛАМИ»

Виконав: Риженко Я.В.

Прийняв: Желізко В. В.

**Мета роботи:** ознайомитися з засобами роботи з файлами

в мові Python, діями над ними.

**Хід роботи**

**Завдання 1.** Створіть новий файл numbers.txt у текстовому редакторі і запишіть

у нього 10 чисел, кожне з нового рядка. Напишіть програму, яка зчитує ці

числа з файла і обчислює їх суму, виводить цю суму на екран і, водночас,

записує цю суму у інший файл з назвою sum\_numbers.txt.

**Листинг програми:**

with open("numbers.txt", "w") as file:  
 for number in [5, 356, 12, -120, -25, -37, 35, 40, 45, 50]:  
 file.write(f"{number}\n")  
  
with open("numbers.txt", "r") as file:  
 numbers = [int(line.strip()) for line in file]  
 total\_sum = sum(numbers)  
  
print(f"Сума чисел: {total\_sum}")  
  
with open("sum\_numbers.txt", "w") as file:  
 file.write(f"{total\_sum}")

**Результат виконання:**

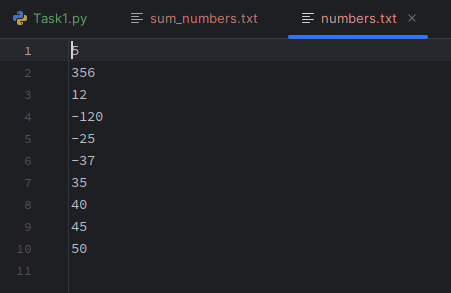
****

Рис. 1. Числа.

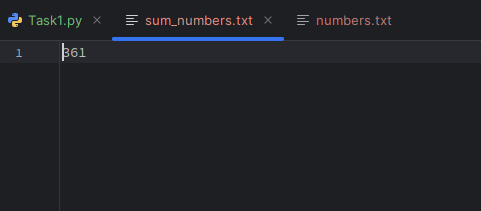


Рис. 2. Текстовий документ, створений після виконання.

**Завдання 2.**

Реалізуйте програму, яка зчитує довільну кількість цілих чисел, що

вводяться з командного рядка, і записує у текстовий файл інформацію, щодо

парності або непарності чисел.

**Листинг програми:**

nums = []  
print("Введіть довільну кількість цілих чисел (або введіть 'end' для завершення):")  
  
while True:  
 user\_input = input()  
 if user\_input.lower() == "end":  
 break  
 try:  
 num = int(user\_input)  
 nums.append(num)  
 except ValueError:  
 print("Будь ласка, введіть ціле число або 'end'.")  
  
print("Введені числа:", nums)  
  
with open("oddOrEven.txt", "w") as file:  
 for num in nums:  
 if num % 2 == 0: file.write(str(num)+" - Even" + "\n")  
 else: file.write(str(num)+" - Odd" + "\n")

**Результат виконання:**

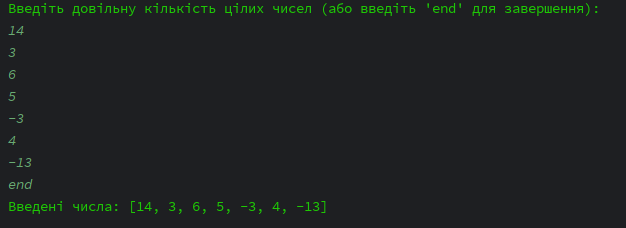
****

Рис. 3. Результат.

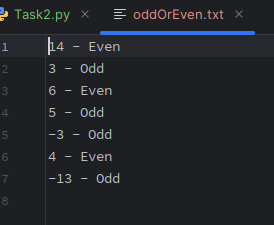
****

Рис. 4. Текстовий документ, створений після виконання.

**Завдання 3.**

Створіть новий файл у текстовому редакторі і напишіть кілька

рядків тексту у ньому про можливості Python. Кожен рядок повинен

починатися з фрази: «Python можна використати для ...» . Збережіть файл з

ім’ям learning\_python.txt. Напишіть програму, яка зчитує файл і виводить текст

з перебором рядків файла і зі збереженням рядків у списку з подальшим

сортуванням списку за довжиною рядків в ньому від найбільшого до

найменшого.

**Листинг програми:**

with open("learning\_python.txt", "w", encoding="utf-8") as file:  
 file.write("Python можна використати для аналізу даних.\n")  
 file.write("Python можна використати для створення веб-додатків.\n")  
 file.write("Python можна використати для машинного навчання.\n")  
 file.write("Python можна використати для автоматизації задач.\n")  
  
filename = "learning\_python.txt"  
  
try:  
 with open(filename, "r", encoding="utf-8") as file:  
 lines = [line.strip() for line in file.readlines()]  
  
 sorted\_lines = sorted(lines, key=len, reverse=True)  
  
 print("Вміст файлу, відсортований за довжиною рядків:")  
 for line in sorted\_lines:  
 print(line)  
  
except FileNotFoundError:  
 print(f"Файл '{filename}' не знайдено. Будь ласка, створіть його спочатку.")  
  
print(f"Файл '{filename}' створено та заповнено.")

**Результат виконання:**

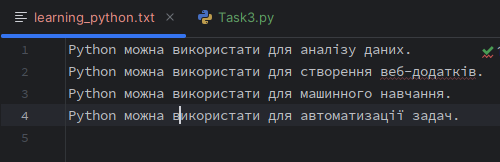
****

Рис. 5. Текстовий документ, створений після виконання.

**Завдання 4.**

Прочитайте кожен рядок зі створеного у попередньому завданні

файла learning\_python.txt і замініть слово Python назвою іншої мови, наприклад

C при виведенні на екран. Отриманий файл має бути створений в новому

каталозі, що розміщується в поточному. Відкрийте файл пострічково і дайте

можливість користувачеві визначити які змінені фрази є актуальними,

наприклад для мови С, а які ні. Всі хибні твердження запишіть в інший файл, а істинні – в поточний.

**Листинг програми:**

import os  
  
original\_file = "learning\_python.txt"  
  
new\_directory = "new\_directory"  
os.makedirs(new\_directory, exist\_ok=True)  
  
true\_file = os.path.join(new\_directory, "true\_statements.txt")  
false\_file = os.path.join(new\_directory, "false\_statements.txt")  
  
modified\_file = os.path.join(new\_directory, "learning\_c.txt")  
  
with open(original\_file, "r", encoding="utf-8") as infile, \  
 open(modified\_file, "w", encoding="utf-8") as outfile:  
 for line in infile:  
 modified\_line = line.replace("Python", "C")  
 outfile.write(modified\_line)  
  
print(f"Файл зі зміненими рядками створено: {modified\_file}")  
  
with open(modified\_file, "r", encoding="utf-8") as infile, \  
 open(true\_file, "w", encoding="utf-8") as true\_outfile, \  
 open(false\_file, "w", encoding="utf-8") as false\_outfile:  
  
 for line in infile:  
 print(f"Чи актуальне це твердження? {line.strip()}")  
 user\_response = input("Введіть 'y' для так або 'n' для ні: ").strip().lower()  
  
 if user\_response == "y":  
 true\_outfile.write(line)  
 elif user\_response == "n":  
 false\_outfile.write(line)  
 else:  
 print("Некоректна відповідь. Пропускаємо це твердження.")

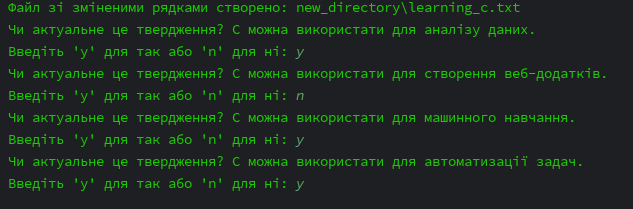
**Результат виконання:** ****

Рис. 6. Результат.

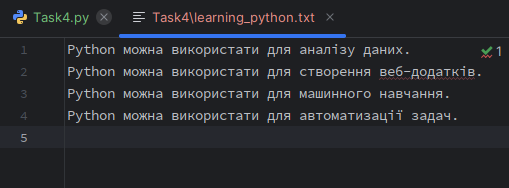
****

Рис. 7. Текстовий документ, створений після виконання.

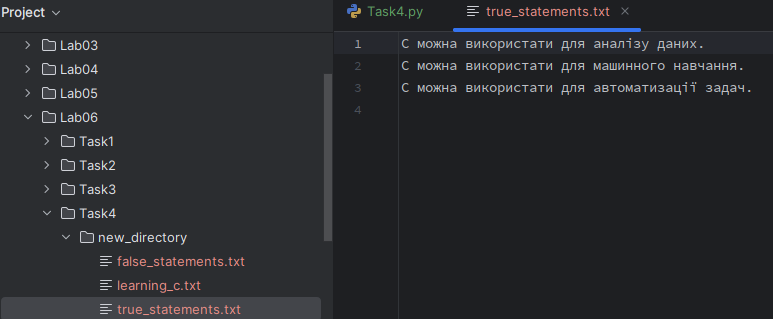
****

Рис. 8. Текстовий документ, створений після виконання.

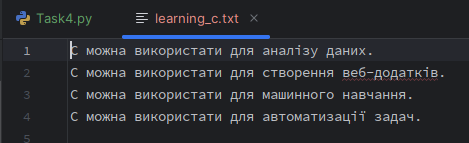
****

Рис. 9. Текстовий документ, створений після виконання.

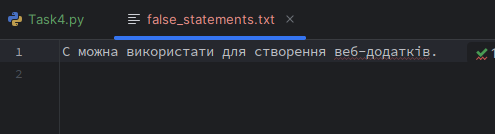
****

Рис. 10. Текстовий документ, створений після виконання.

**Завдання 5.**

Створіть порожній файл guest\_book.txt у текстовому редакторі.

Напишіть програму, яка запитує у користувачів імена. При введенні кожного

імені виведіть на екран рядок з вітанням для користувача і запишіть рядок

вітання у файл з ім’ям guest\_book.txt. Простежте за тим, щоб кожне

повідомлення розміщувалося в окремому рядку файла з зазначенням часу

внесення цього повідомлення. Передбачте зазначення в файлі часу його

створення і вказання в ньому часу останніх внесених змін.

**Листинг програми:**

import time  
import os  
  
def add\_guest\_to\_book(guest\_name, file\_name="guest\_book.txt"):  
 current\_time = time.strftime("%Y-%m-%d %H:%M:%S")  
 greeting = f"Привіт, {guest\_name}! (Додано: {current\_time})"  
 print(greeting)  
 with open(file\_name, "a", encoding="utf-8") as file:  
 file.write(f"{greeting}\n")  
 update\_last\_modified\_time(file\_name)  
  
def update\_last\_modified\_time(file\_name):  
 if os.path.exists(file\_name):  
 with open(file\_name, "r", encoding="utf-8") as file:  
 lines = file.readlines()  
 last\_modified = f"Останні зміни: {time.strftime('%Y-%m-%d %H:%M:%S')}\n"  
 if len(lines) > 0 and "Останні зміни:" in lines[-1]:  
 lines[-1] = last\_modified  
 else:  
 lines.append(last\_modified)  
 lines.append("-" \* 50 + "\n")  
 with open(file\_name, "w", encoding="utf-8") as file:  
 file.writelines(lines)  
  
print("Введіть імена гостей. Для завершення введіть 'вихід'.")  
while True:  
 guest\_name = input("Введіть ім'я: ").strip()  
 if guest\_name.lower() == 'вихід':  
 print("Роботу завершено.")  
 break  
 elif guest\_name:  
 add\_guest\_to\_book(guest\_name)

**Результат виконання:**

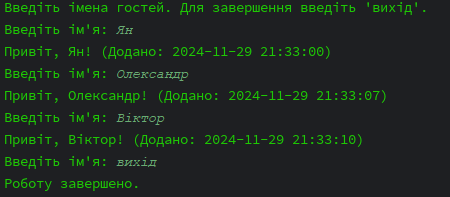
****

Рис. 11. Результат.

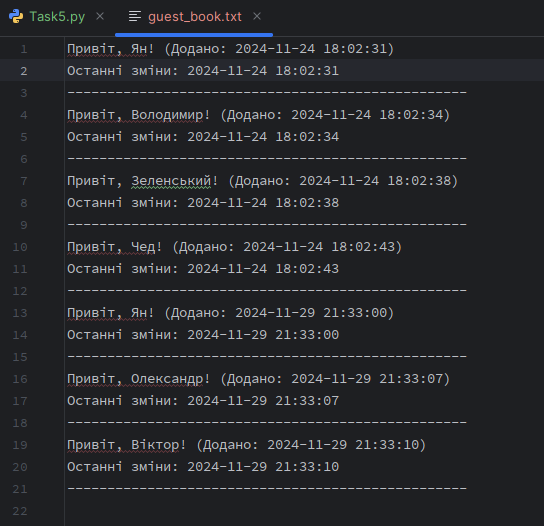


Рис. 12. Текстовий документ, створений після виконання.

**Завдання 6.**

Збережіть в тектовому файлі публікацію про Python на 3000 слів

англійською мовою. Напишіть програму, що аналізуватиме частоту з якою в

тексті зустрічастимуться окремі літери чи слова незалежно від їх регістру.

Результат робот програми має виводитись в консоль і зберігатись в окремому

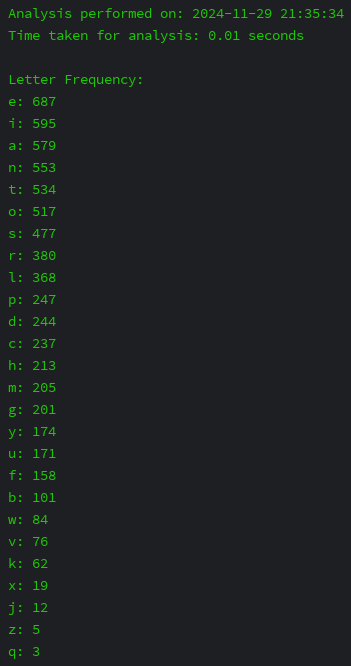
файлі з зазначенням часу його стоврення, часу виконнання окремих змін,

результатів пошкуку і часу, що знадовся на виконнання цього пошуку.

**Листинг програми:**

import os  
import time  
from collections import Counter  
  
input\_file = "python.txt" #Публікація на 3000 слів англійською мовою  
  
output\_file = f"analysis\_results\_{time.strftime('%Y%m%d\_%H%M%S')}.txt"  
  
def analyze\_text(file\_path):  
 start\_time = time.time()  
  
 with open(file\_path, "r", encoding="utf-8") as file:  
 text = file.read()  
  
 letters = Counter(c.lower() for c in text if c.isalpha())  
  
 words = Counter(word.lower() for word in text.split())  
  
 end\_time = time.time()  
 elapsed\_time = end\_time - start\_time  
  
 results = [  
 f"Analysis performed on: {time.strftime('%Y-%m-%d %H:%M:%S')}",  
 f"Time taken for analysis: {elapsed\_time:.2f} seconds",  
 "\nLetter Frequency:",  
 \*[f"{letter}: {count}" for letter, count in letters.most\_common()],  
 "\nWord Frequency:",  
 \*[f"{word}: {count}" for word, count in words.most\_common(20)] # Top 20 words  
 ]  
  
 with open(output\_file, "w", encoding="utf-8") as out\_file:  
 out\_file.write("\n".join(results))  
  
 return results, output\_file  
  
analysis\_results, result\_file\_path = analyze\_text(input\_file)  
  
print("\n".join(analysis\_results))  
  
  
print(f"\nResults saved to: {result\_file\_path}")

**Результат виконання:**

****

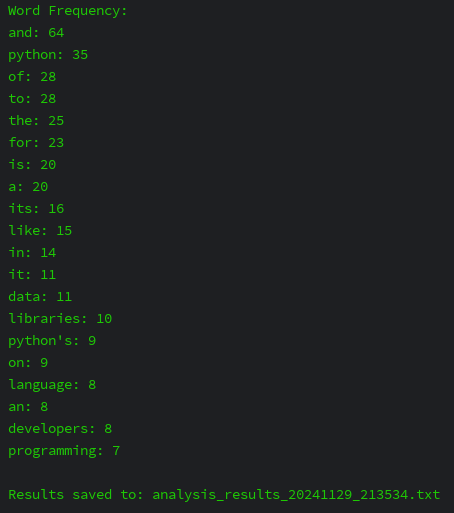
****

Рис. 13. Результат.

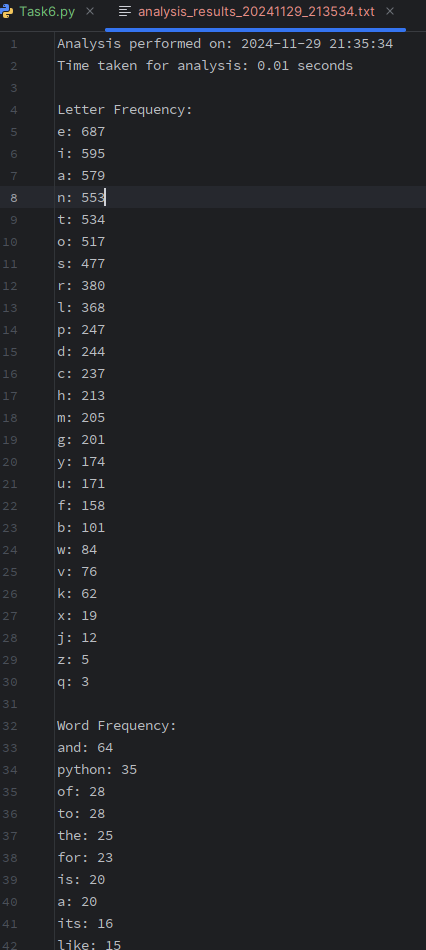


Рис. 14. Текстовий документ, створений після виконання.

**Завдання 7.**

Завантажте файл marks.csv і визначте кількість студентів, що

проходили тестування. Виведіть інформацію про те яку оцінку набрали

відповідна кільксть студентів. Виведіть інформацію яку середню оцінку

отримував студент за певний час виконання КМР (крок – 1 хв). Створіть

текстовий файл і запишіть в нього статистику по правильним відповідям для

кожного окремого питання (який відсоток правильних і неправильних

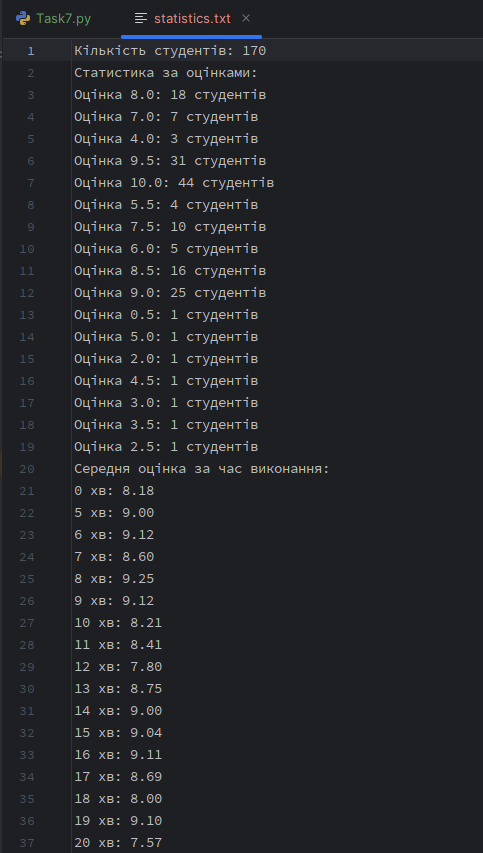
відповідей на питання дали студенти). В цей же файл внесіть інформацію про 5

найкращих оцінок в співвідношенні оцінка/час витрачений складання КМР.

**Листинг програми:**

import csv  
  
def parse\_duration(duration\_str):  
 if "хв" in duration\_str and "сек" in duration\_str:  
 parts = duration\_str.split(" ")  
 minutes = int(parts[0].replace("хв", ""))  
 seconds = int(parts[2].replace("сек", ""))  
 return minutes + seconds / 60  
 return 0  
  
def safe\_float\_conversion(value):  
 try:  
 return float(value.replace(",", "."))  
 except ValueError:  
 return None  
  
file\_name = "marks.csv"  
students\_data = []  
with open(file\_name, "r", encoding="utf-8") as file:  
 reader = csv.reader(file)  
 for row in reader:  
 students\_data.append(row)  
  
student\_ids = {row[0] for row in students\_data}  
students\_count = len(student\_ids)  
  
grades\_count = {}  
for row in students\_data:  
 grade = safe\_float\_conversion(row[4])  
 if grade is not None:  
 grades\_count[grade] = grades\_count.get(grade, 0) + 1  
  
time\_to\_grades = {}  
for row in students\_data:  
 duration = parse\_duration(row[3])  
 grade = safe\_float\_conversion(row[4])  
 if grade is not None:  
 rounded\_time = round(duration)  
 if rounded\_time not in time\_to\_grades:  
 time\_to\_grades[rounded\_time] = []  
 time\_to\_grades[rounded\_time].append(grade)  
  
average\_grades\_per\_time = {time: sum(grades) / len(grades) for time, grades in time\_to\_grades.items()}  
  
questions\_stats = {}  
questions = students\_data[0][5:]  
for i, question in enumerate(questions):  
 correct = sum(1 for row in students\_data if safe\_float\_conversion(row[5 + i]) and safe\_float\_conversion(row[5 + i]) > 0)  
 total = len(students\_data)  
 questions\_stats[f"Question {i + 1}"] = {  
 "correct\_percent": (correct / total) \* 100,  
 "incorrect\_percent": 100 - (correct / total) \* 100,  
 }  
  
ratios = []  
for row in students\_data:  
 grade = safe\_float\_conversion(row[4])  
 duration = parse\_duration(row[3])  
 if grade is not None and duration > 0:  
 ratio = grade / duration  
 ratios.append((ratio, row[0]))  
  
top\_5\_ratios = sorted(ratios, reverse=True)[:5]  
  
output\_file = "statistics.txt"  
with open(output\_file, "w", encoding="utf-8") as file:  
 file.write(f"Кількість студентів: {students\_count}\n")  
 file.write("Статистика за оцінками:\n")  
 for grade, count in grades\_count.items():  
 file.write(f"Оцінка {grade}: {count} студентів\n")  
 file.write("Середня оцінка за час виконання:\n")  
 for time, avg\_grade in sorted(average\_grades\_per\_time.items()):  
 file.write(f"{time} хв: {avg\_grade:.2f}\n")  
 file.write("Аналіз правильності відповідей:\n")  
 for question, stats in questions\_stats.items():  
 file.write(f"{question}: Правильно: {stats['correct\_percent']:.2f}%, Неправильно: {stats['incorrect\_percent']:.2f}%\n")  
 file.write("Топ-5 співвідношень оцінка/час:\n")  
 for ratio, student\_id in top\_5\_ratios:  
 file.write(f"ID {student\_id}: {ratio:.2f}\n")

**Результат виконання:**



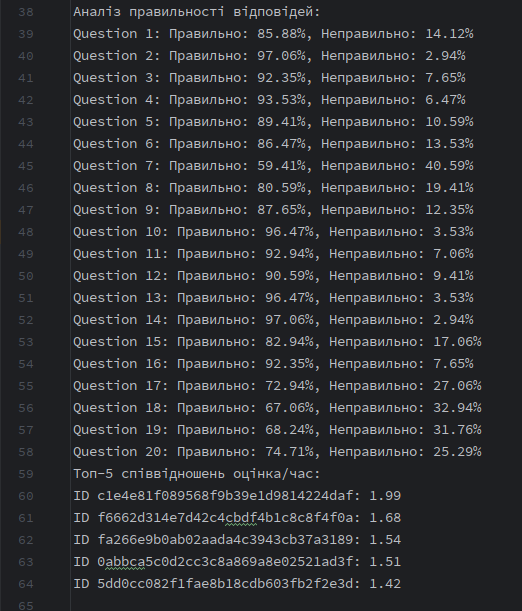


Рис. 15. Текстовий документ, створений після виконання.

**Висновок:** Я навчився працювати з файлами в Python: відкривати їх у різних режимах, читати та записувати дані, використовувати контекстний менеджер with для автоматичного закриття файлів, а також обробляти текстові файли. Це дозволяє ефективно керувати файлами та забезпечувати коректну роботу програм із зовнішніми даними.

***Посилання на Git****: :* *<https://github.com/JanRizhenko/Python_Labs>*