

Міністерство освіти і науки України  
Національний технічний університет України «Київський політехнічний  
інститут імені Ігоря Сікорського»  
Факультет інформатики та обчислювальної техніки  
Кафедра інформатики та програмної інженерії

Звіт

З лабораторної роботи №1 з дисципліни  
«Основи програмування 2.  
Модульне програмування»

«Файли даних»

Варіант 27

Виконав студент ІП-11 Савенко Олексій Андрійович

(шифр, прізвище, ім'я, по батькові)

Перевірів Вітковська І.І.

(прізвище, ім'я, по батькові)

Київ 2022

# Лабораторна робота 1

## Файли даних

**Мета роботи** – вивчити особливості створення і обробки текстових файлів даних.

## Варіант 27

### Індивідуальне завдання

27. Створити текстовий файл. Кожен парний рядок вихідного файлу переписати в перший новий текстовий файл, кожен непарний - у другий. У файлі з парними рядками змінити рядки таким чином, щоб слова кожного рядка були лексично впорядковані за алфавітом. Вивести вміст вихідного і створених файлів.

### Постановка задачі

Потрібно створити текстовий файл: його створення та заповнення потрібно автоматизувати програмно, його дозаповнення буде доступно для користувача протягом виконання програми. Перепишемо в новостворені файли парні та непарні рядки відповідно. Для виконання останнього завдання необхідно поділити текст на рядки, далі на слова та відсортувати їх за алфавітом.

Програмний код на мові **Python**:

```
import os

def inputText(): #заповнення стартового файлу
    print("Input your text for initial file(to end the entering double press Enter):")
    initial = 'initial.txt'
    file = open(initial, 'w'); #відкриття файлу для запису
    text = " " #temp
    c = 0
    while(text): #isEmpty  когда в строке не пусто
        text=input()
        if c == 0: file.write(text)
        else: file.write('\n'+text)
        c+=1
    file.close()
    return initial

def chooseFiles(a,b,c): #перепис рядків по файлах
```

```

initial = open(a, 'r') #стартовий файл, для зчитування
even = open(b, 'w') #для запису парних рядків
odd = open(c, 'w') #для запису непарних рядків
c = 1 #фіксування відповідного номеру рядка
text = " " #рядок для запису введенного користувачем тексту
while(text):
    text = initial.readline() #зчитуємо порядково
    if text:
        if(c%2==0):even.write(text)
        else: odd.write(text) #Перенесение на нов строку автомтаом
        c+=1
initial.close()
even.close()
odd.close()

def addText(): #дозаповнення файлу
    print("Input your text:")
    a = 'text.txt'
    f = open(a, 'a'); #відкриваємо файл для дозапису
    text = " "
    c = 0
    while(text):
        text=input()
        if c == 0: f.write(text)
        else: f.write("\n"+text)
        c+=1
    f.close()

def outputFile(a): #виведення вмісту файлів
    f = open(a, 'r')
    text = f.read()
    print(text)
    f.close()

def alphabet(address): #сортування за алфавітом
    file = open(address, 'r')
    text = file.read() #записуємо весь текст файлу
    arr = text.split('\n') #розділяємо текст по рядках в список
    arr.pop(len(arr)-1) #видалення зайвого рядка (пустого)
    file.close()
    numb = 0 #Індекс відповідного рядка файлу
    for i in arr:
        tArr = i.split(' ') #розділяємо рядок на слова
        for j in range(0, len(tArr)-1): #сортування алфавітом(за основу взято бульбашку)
            for j1 in range(0, len(tArr)-1):
                if len(tArr[j1])>len(tArr[j1+1]):
                    for k in range(0, len(tArr[j1+1])):
                        if tArr[j1][k] > tArr[j1+1][k]:
                            t = tArr[j1+1]
                            tArr[j1+1] = tArr[j1]
                            tArr[j1] = t
                            break
                        elif tArr[j1][k] < tArr[j1+1][k]:
                            break
            else:
                for k in range(0, len(tArr[j1])):
                    if tArr[j1][k] > tArr[j1+1][k]:
                        t = tArr[j1+1]

```

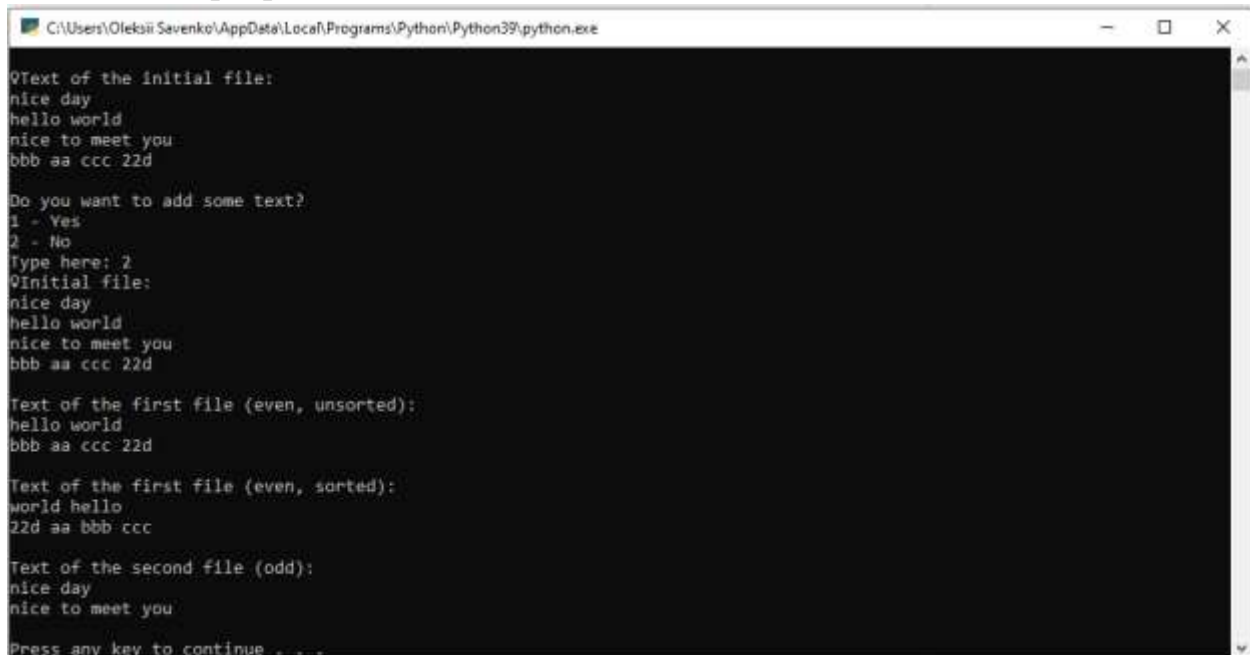
```

        tArr[j1+1] = tArr[j1]
        tArr[j1] = t
        break
    elif tArr[j1][k] > tArr[j1+1][k]:
        break
    arr[numb] = tArr #запис у масив рядків відсортований ряд слів
    numb+=1
file = open(adress, 'w') #запис зміненого тексту в файл
for i in arr:
    file.write(' '.join(i)+'\n')
file.close()

if __name__=="__main__":
    intialFile = inputText() #стартовий файл, ф-ція повертає назву файлу
    os.system('cls||clear') #очищення консолі
    print("Text of the initial file:")
    outputFile(intialFile)
    choice = int(input("Do you want to add some text?\n1 - Yes\n2 - No\nType here: "))
    if choice == 1:
        addText()
        os.system('cls||clear')
        print("Initial file:")
        outputFile(intialFile)
        even = 'even.txt' #перший файл (для парних)
        odd = 'odd.txt' #другий файл (для непарних)
        chooseFiles(intialFile, even, odd)
        print("Text of the first file (even, unsorted):")
        outputFile(even)
        alphabet(even)
        print("Text of the first file (even, sorted):")
        outputFile(even)
        print("Text of the second file (odd):")
        outputFile(odd)

```

## Виконання програми на мові Python:



```

C:\Users\Oleksii Savenko\AppData\Local\Programs\Python\Python39\python.exe

QText of the initial file:
nice day
hello world
nice to meet you
bbb aa ccc 22d

Do you want to add some text?
1 - Yes
2 - No
Type here: 2
QInitial file:
nice day
hello world
nice to meet you
bbb aa ccc 22d

Text of the first file (even, unsorted):
hello world
bbb aa ccc 22d

Text of the first file (even, sorted):
world hello
22d aa bbb ccc

Text of the second file (odd):
nice day
nice to meet you

Press any key to continue . . .

```

## Програмний код на мові C++:

```
#include <iostream>
#include <fstream>
#include <string>
#include "Functions.h"
using namespace std;

int main() {
    string initial = inputText(); //Створення початкового файла з поверненням його адреси
з функції
    system("CLS"); //очищення консолі
    string even = "even.txt"; //Файл для парних рядків
    string odd = "odd.txt"; //Файл для непарних рядків
    cout << "Initial file:\n";
    outputFile(initial);
    cout << "If you want add some information to your file, press:\n1 - Agree\n2 -
Deny\nType your answer here: ";
    int choice;
    cin >> choice;
    if (choice == 1) addText(initial);
    system("CLS");
    chooseFile(initial, even, odd);
    cout << "Initial file:\n";
    outputFile(initial);
    cout << "File with even lines (unalphabet):\n";
    outputFile(even);
    alphabetSort(even);
    cout << "File with even lines (alphabet):\n";
    outputFile(even);
    cout << "File with odd lines:\n";
    outputFile(odd);
}
```

### Functions.cpp

```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;
```

```

string inputText() {
    string initial = "Initial.txt";
    cout << "Input your text (press Ctrl-Q combination to stop entering your
text):\n";
    ofstream fout(initial); //Створення та відкриття об'єкту класу для запису даних у
нього
    string str; //Рядок для заповнення начального файлу
    int count = 0; //Номер відповідного рядка, який вводиться користувачем
    while (str.find(17) == string::npos) { //перевірка на наявність комбінації(Ctrl-Q) в
поточці
        getline(cin, str);
        if (count == 0 && str.find(17) == string::npos)
            fout << str;
        else if (str.find(17) == string::npos)
            fout << endl << str;
        count++;
    }
    fout.close();
    return initial;
}

void chooseFile(string init, string even, string odd) {
    ifstream filein(init); //Створення та відкриття об'єкту класу для читання даних з
нього
    ofstream even1(even); //..для запису в перший файл (для парних)
    ofstream odd2(odd); //..для запису в другий файл(для непарних)
    string str;
    int count = 1; // Лічильник номеру відповідної строки
    while (!filein.eof()) { //перевірка на кінець текстового файла
        getline(filein, str);
        if (count % 2 == 0) {
            if (count == 2) even1 << str;
            else even1 << endl << str;
        }
        else {
            if (count == 1) odd2 << str;
            else odd2 << endl << str;
        }
        count++;
    }
    filein.close();
    even1.close();
    odd2.close();
}

void addText(string address) {
    ofstream fileout(address, ios::app); //app (append) використовується для дозапису
даних у кінець файлу
    cout << "You can add some text,to end entering the text press Ctrl-Q:\n";
    string str;
    int count = 0;
    while (str.find(17) == string::npos) {
        getline(cin, str);
        if (count == 0 && str.find(17) == string::npos)
            fileout << str;
        else if (str.find(17) == string::npos)
            fileout << endl << str;
    }
}

```

```

        count++;
    }
    fileout.close();
}

void outputFile(string address) {
    ifstream filein(address); //Об'єкт класу файла для читання інформації з нього
    string str;
    while (!filein.eof()) {
        getline(filein, str);
        cout << str << endl;
    }
    cout << endl;
}

void alphabetSort(string address) {
    fstream filein(address); //об'єкт універсального класу (i/o)(як читання, так і запис)
    int cntLines = 0;
    string str;
    while (!filein.eof()) { //Обрахування кількості рядків у файлі
        getline(filein, str);
        cntLines++;
    }
    filein.close();
    filein.open(address, ios::in); //відкриття файла у режимі зчитування інформації
    string* lines = new string[cntLines]; //масив для запису рядків тексту з файла
    for (int i = 0; i < cntLines; i++) {
        getline(filein, lines[i]);
    }
    filein.close();
    for (int i = 0; i < cntLines; i++) {
        int cntSpace = 0;
        string nStr = "";
        for (int j = 0; j < lines[i].length(); j++) { //Видалення надлишкових пробілів
            між словами відповідного рядка
            if (lines[i][j] == ' ') {
                cntSpace++;
            }
            else {
                if (cntSpace != 0) {
                    cntSpace = 0;
                    nStr += ' ';
                }
                nStr += lines[i][j];
            }
        }
        if (nStr[0] == ' ') nStr.erase(0, 1); //Видалення пробіла у випадку першого
        символу рядка
        if (nStr[nStr.length() - 1] != ' ') nStr.push_back(' ');
        int cntWords = 0;
        lines[i] = nStr; //Переназначення вже відредактованої строки замість
        відповідного елементу у масиві строк
        while (nStr.length() > 0) { //обрахування к-сті слів у рядку
            cntWords++;
            nStr.erase(0, nStr.find(' ') + 1);
        }
        string* words = new string[cntWords]; //перезапис рядка в масив слів
        nStr = lines[i];
    }
}

```

```

    for (int j = 0; j < cntWords; j++) {
        words[j] = nStr.substr(0, nStr.find(' '));
        nStr.erase(0, nStr.find(' ') + 1);
    }
    for (int j = 0; j < cntWords - 1; j++) { //сортування слів за алфавітом
        for (int g = 1; g < cntWords; g++) {
            if (words[g - 1].length() > words[g].length()) {
                for (int k = 0; k < words[g].length(); k++) {
                    if (words[g - 1][k] > words[g][k]) { //перевірка букв сусідніх
                        string t = words[g];
                        words[g] = words[g - 1];
                        words[g - 1] = t;
                        break;
                    }
                    else if (words[g - 1][k] < words[g][k]) break;
                }
            }
            else {
                for (int k = 0; k < words[g - 1].length(); k++) {
                    if (words[g - 1][k] > words[g][k]) {
                        string t = words[g];
                        words[g] = words[g - 1];
                        words[g - 1] = t;
                        break;
                    }
                    else if (words[g - 1][k] < words[g][k]) break;
                }
            }
        }
    }
    lines[i] = "";
    for (int j = 0; j < cntWords; j++) { //запис відсортованих слів у відповідний
рядок
        lines[i] += words[j] + ' ';
    }
    lines[i].pop_back();
}
filein.open(adress, fstream::in | fstream::out | fstream::trunc); //запис
відповідного рядка масиву до файлу (trunc - для очистки файлу)
for (int i = 0; i < cntLines; i++) {
    filein << lines[i] << endl;
}
filein.close();
}

```

## Functions.h

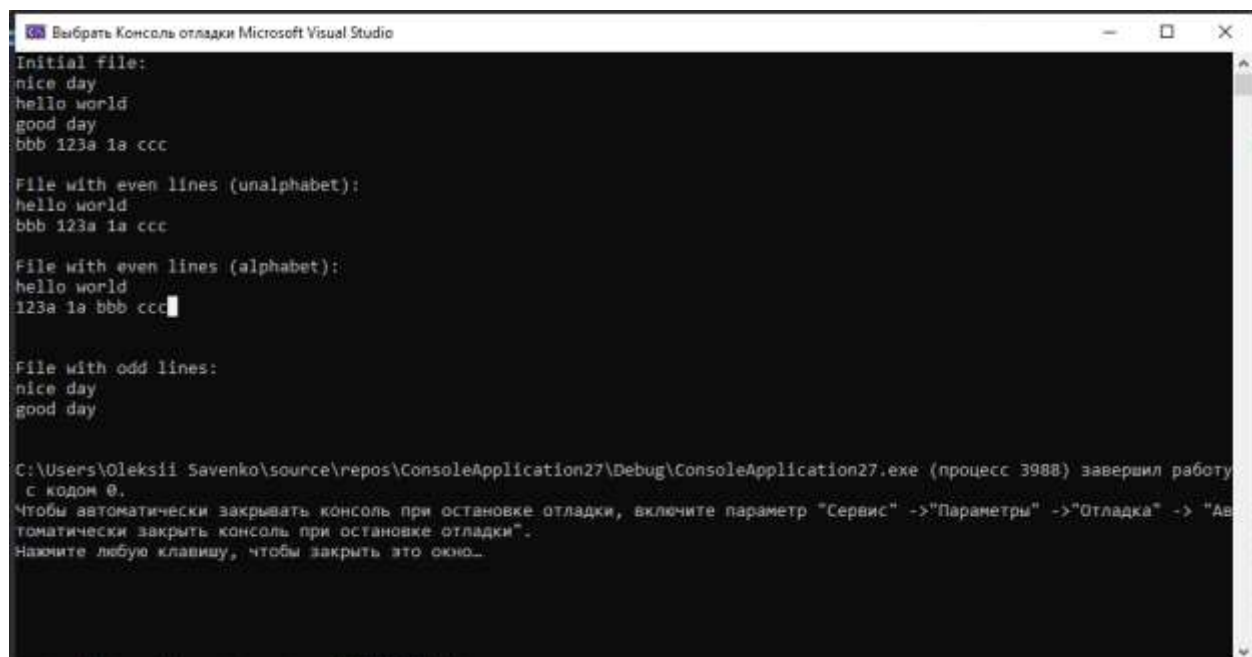
```

#pragma once
#include <string>
using namespace std;
string inputText(); //заповнення текстом стартового файлу
void addText(string); //Доповнення файла текстом у разі потреби користувача
void chooseFile(string, string, string); //Розподілення тексту по відповідним файлам
void outputFile(string); //Виведення інформації з відповідного файла
void alphabetSort(string); //Відсортування слів у рядках файла за алфавітом

```



## Виконання програмного коду на мові C++:



```
Выбрать Консоль отладки Microsoft Visual Studio
Initial file:
nice day
hello world
good day
bbb 123a 1a ccc

File with even lines (unalphabet):
hello world
bbb 123a 1a ccc

File with even lines (alphabet):
hello world
123a 1a bbb ccc

File with odd lines:
nice day
good day

C:\Users\Oleksii Savenko\source\repos\ConsoleApplication27\Debug\ConsoleApplication27.exe (процесс 3988) завершил работу
с кодом 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" -> "Ав-
томатически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно...
```

### **Висновок**

Отже я вивчив особливості створення і обробки текстових файлів даних. Вивчений мною матеріал був застосований для виконання поданого завдання, було реалізовано автоматизоване створення файлів, запис та перезапис у інші файли, для позначення кінця інформації, яка буде записано було використано послідовність клавіш (CTRL-Q – C++, а також Enter - Python). Для виконання відповідної дії були застосовані відомості, віднайдені у таблиці ASCII. Для сортування за алфавітом за основу було взято Bubble-sort, з метою посимвольного порівняння сусідніх слів та відповідного результату у вигляді слів відсортованих за алфавітом.