НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені ІГОРЯ СІКОРСЬКОГО»

ЗВІТ

З лабораторної роботи №4

з навчальної дисципліни

«Програмцвання алгоритмічниї структур»

Варіант № 7

Тема:

**Методи пошуку у масивах**

Студентки 1-го курсу НН ІАТЕ гр. ТР-22

Грищук Анжеліка

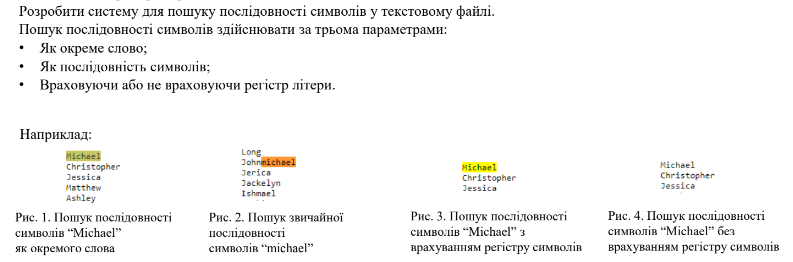
Лабораторну роботу захищено

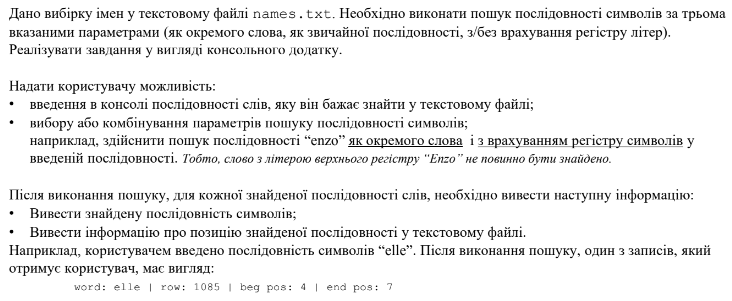
З оцінкою\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

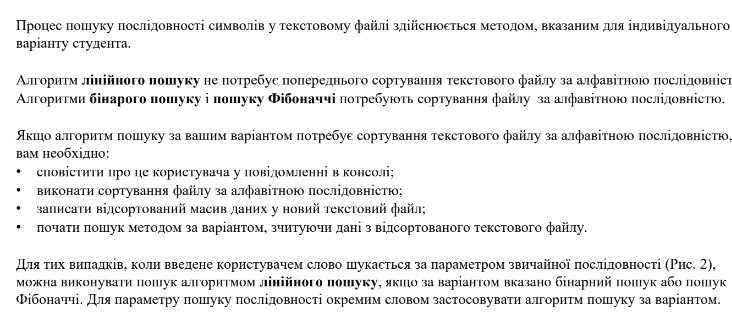
КИЇВ 2023

**Мета роботи:** Метою лабораторної роботи є отримання практичних навичок в обробці масивів, у пошуку елементів масиві різними методами. Дослідження і вивчення методів пошуку ключових елементів у масивах. Здійснення порівняння тааналізу ефективності використовуваних методів пошуку.

**Завдання лабораторної роботи:**







**Варіант №7.** 

**Код програми:**

#include <iostream>

#include <fstream>

#include <string>

#include <windows.h>

#include <vector>

#include <algorithm>

using namespace std;

//--------------- 1 -------------

// Враховуючи регістер літери

struct fibs {

int i = 0;

int p = 0;

int q = 0;

bool flag = false;

};

int getFibonacciNum(int num) {

if (num == 0) {

return 0;

}

if (num == 1) {

return 1;

}

return getFibonacciNum(num - 1) + getFibonacciNum(num - 2);

}

void startInit(std::vector<std::string> arr, fibs& f) {

f.flag = false;

int k = 0;

int size = arr.size();

while (getFibonacciNum(k + 1) < size) {

k += 1;

}

int m = getFibonacciNum(k + 1) - (size + 1);

f.i = getFibonacciNum(k) - m;

f.p = getFibonacciNum(k - 1);

f.q = getFibonacciNum(k - 2);

}

void upIndex(fibs& f) {

if (f.p == 1) {

f.flag = true;

}

f.i += f.q;

f.p -= f.q;

f.q -= f.p;

}

void downIndex(fibs& f) {

if (f.q == 0) {

f.flag = true;

}

int buff = f.q;

f.i -= f.q;

f.q = f.p - f.q;

f.p = buff;

}

int fibonacciSearch1(std::vector<std::string> arr, std::string word) {

fibs f;

startInit(arr, f);

int resIndex = -1;

while (!f.flag) {

if (f.i < 0) {

upIndex(f);

}

else if (f.i >= arr.size()) {

downIndex(f);

}

else if (arr[f.i] == word) {

resIndex = f.i;

break;

}

else if (word < arr[f.i]) {

downIndex(f);

}

else if (word > arr[f.i]) {

upIndex(f);

}

}

return resIndex;

}

//--------------- 2 -------------

// НЕ враховуючи регістер літери

int fibonacciSearch2(std::vector<std::string> arr, std::string word) {

fibs f;

startInit(arr, f);

int resIndex = -1;

while (!f.flag) {

if (f.i < 0) {

upIndex(f);

}

else if (f.i >= arr.size()) {

downIndex(f);

}

else {

std::string currentWord = arr[f.i];

std::transform(currentWord.begin(), currentWord.end(), currentWord.begin(), [](unsigned char c) { return std::tolower(c); });

std::transform(word.begin(), word.end(), word.begin(), [](unsigned char c) { return std::tolower(c); });

if (currentWord == word) {

resIndex = f.i;

break;

}

else if (word < currentWord) {

downIndex(f);

}

else if (word > currentWord) {

upIndex(f);

}

}

}

return resIndex;

}

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// СОРТУВАННЯ ЗА АЛФАВІТОМ \*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Відкриття файлу з іменами для читання

ifstream input("names.txt");

if (!input.is\_open()) {

cerr << "Помилка відкриття файлу names.txt" << endl;

return 1;

}

vector<string> names; // Вектор для зберігання імен

// Зчитування імен з файлу

string name;

while (input >> name) {

names.push\_back(name);

}

input.close(); // Закриття файлу

// Сортування вектора за алфавітом

sort(names.begin(), names.end());

// Відкриття файлу для запису відсортованих імен

ofstream output("Sortednames.txt");

// Запис відсортованих імен в файл

for (const auto& name : names) {

output << name << endl;

}

output.close(); // Закриття файлу

cout << "\* Імена відсортовано та записано у файл Sortednames.txt \*\n" << endl;

int search;

std::cout << "\* виберіть праметр пошуку послідовності символів: \n\t";

std::cout << "1. \* як окреме слово \* \n\t";

std::cout << "2. \* як послідовність символів \* \n\t\t";

cin >> search;

//----------------------------------------------------------------------------

// Як окреме слово (методом алгоритму "Пошуку Фібоначчі") |

//----------------------------------------------------------------------------

if (search == 1) {

int ans;

std::cout << "\*-- враховуючи або не враховуючи регістр літери (1 = yes / 2 = no) --\*\n\t\t";

cin >> ans;

// Враховуючи регістер літери

if (ans == 1) {

std::ifstream infile("Sortednames.txt");

std::vector<std::string> names;

std::string line;

while (std::getline(infile, line)) {

names.push\_back(line);

}

std::sort(names.begin(), names.end());

std::string searchWord;

std::cout << "Введіть слово: "; //пошук рядка, введеного користувачем, у файлі Sortednames.txt

std::cin >> searchWord;

int index = fibonacciSearch1(names, searchWord);

if (index != -1) {

std::cout << "Знайдено " << searchWord << " в рядку: " << index << std::endl;

}

else {

std::cout << "Ім'я " << searchWord << " не знайдено" << std::endl;

}

}

// Не враховуючи регістер літери

if (ans == 2) {

std::ifstream infile("Sortednames.txt");

std::vector<std::string> names;

std::string line;

while (std::getline(infile, line)) {

names.push\_back(line);

}

std::sort(names.begin(), names.end());

std::string searchWord;

std::cout << "Введіть слово: "; //пошук рядка, введеного користувачем, у файлі Sortednames.txt

std::cin >> searchWord;

int index = fibonacciSearch2(names, searchWord);

if (index != -1) {

std::cout << "Знайдено " << searchWord << " в рядку: " << index << std::endl;

}

else {

std::cout << "Ім'я " << searchWord << " не знайдено" << std::endl;

}

}

//-------------------------------------------------------------------------

// Як послідовність символів алгоритмом "Лінійного пошуку" |

//-------------------------------------------------------------------------

if (search == 2) {

int ans2;

std::cout << "\*-- враховуючи або не враховуючи регістр літери (1 = yes / 2 = no) --\*\n\t\t";

cin >> ans2;

//З урахуванням регістру літери

if (ans2 == 1) {

string search\_word2; //визначається як рядок, до якого користувач вводить слово для пошуку.

bool found2 = false; //буде встановлена як true, якщо слово буде знайдено в файлі.

ifstream file("names.txt");

cout << "Введіть слово: ";

cin >> search\_word2;

string line2; //використовується для збереження рядка, який було прочитано з файлу.

int line\_num2 = 1; // використовується для збереження номера рядка, на якому було знайдено слово.

while (getline(file, line2)) { //кожен рядок файлу по черзі читається в line2. Якщо шукане слово знаходиться в рядку

if (line2.find(search\_word2) != string::npos) {

int start\_pos = line2.find(search\_word2);

int end\_pos = start\_pos + search\_word2.length();

cout << "Знайдено ";

cout << line2.substr(0, start\_pos); // виводимо частину рядка до знайденого слова

cout << "\033[33m" << line2.substr(start\_pos, search\_word2.length()) << "\033[0m"; // виводимо знайдену частину оранжевим кольором

cout << line2.substr(end\_pos); // виводимо частину рядка після знайденого слова

cout << " в рядку ---> " << line\_num2 << ": " << endl;

found2 = true;

}

line\_num2++;

}

if (!found2) {

cout << "\* Слово не знайдено \*" << endl;

}

file.close();

}

//БЕЗ урахування регістру літери

if (ans2 == 2) {

string search\_word2; // змінна "search\_word2", яка зберігатиме введене користувачем слово

bool found2 = false; // початково встановлюється в "false" і пізніше змінюється на "true", якщо слово було знайдено у файлі.

ifstream file("names.txt"); //відкриття файлу

cout << "Введіть слово: ";

cin >> search\_word2;

string line2;

int line\_num2 = 1;

while (getline(file, line2)) { // витягується кожен рядок з файлу "Sortednames.txt"

string lowercase\_line;

for (char& c : line2) {

lowercase\_line += tolower(c); // перетворюємо усі літери на малі

}

string lowercase\_word;

for (char& c : search\_word2) {

lowercase\_word += tolower(c); // перетворюємо усі літери на малі

}

size\_t pos = 0;

while ((pos = lowercase\_line.find(lowercase\_word, pos)) != string::npos) { //Пошук слова у рядку

line2.replace(pos, search\_word2.length(), "\033[33m" + line2.substr(pos, search\_word2.length()) + "\033[0m");

pos += 13; // додаємо 13, щоб не повторювати підстрічку

}

if (lowercase\_line.find(lowercase\_word) != string::npos) {

cout << "Знайдено " << line2 << " в рядку ---> " << line\_num2 << ": " << endl;

found2 = true;

}

line\_num2++;//Збільшується лічильник номеру рядка.

}

if (!found2) {

cout << "\* Слово не знайдено \*" << endl;

}

file.close();

}

}

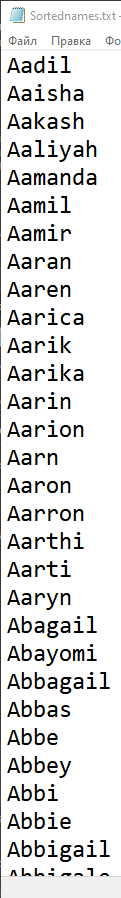
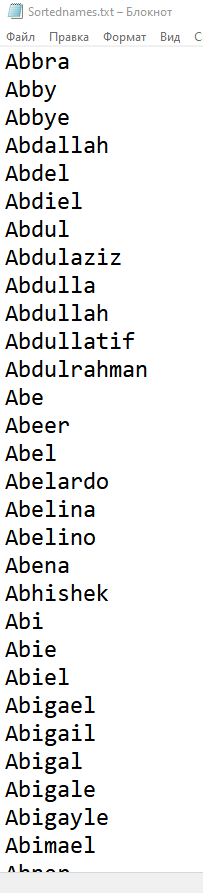
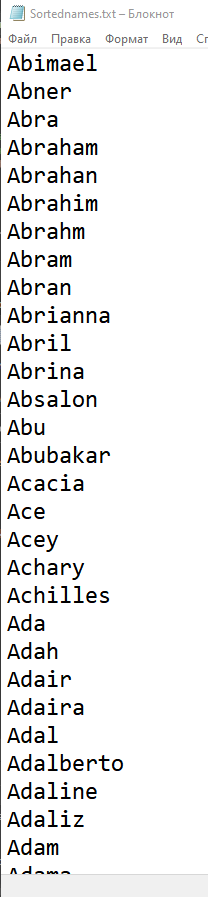
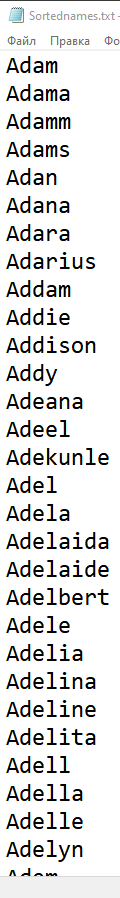
return 0;

}

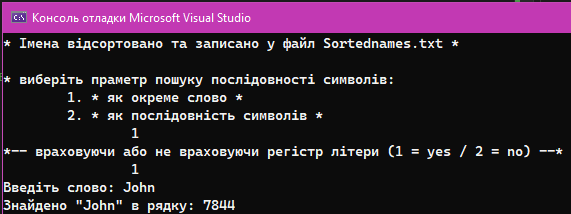
}

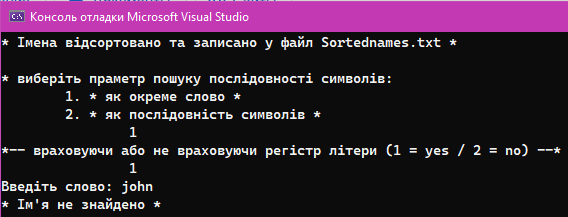
**Скріншоти, що демонструють процес виконання роботи:**

1. скріншоти створеного відсортованого за алф. порядком файлу;

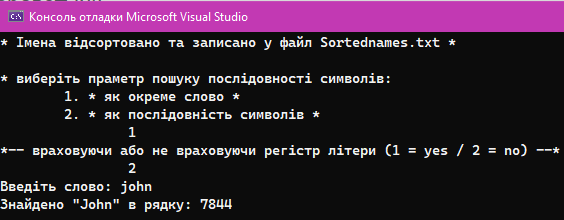
   

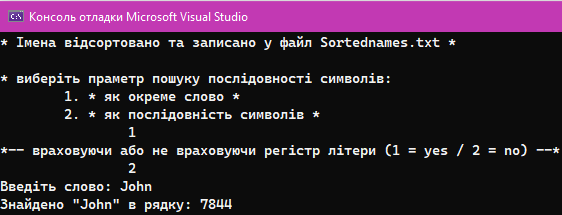
2. Скріншоти перевірки результату пошуку **«Як окремого слова»** З врахуванням регістру літери:



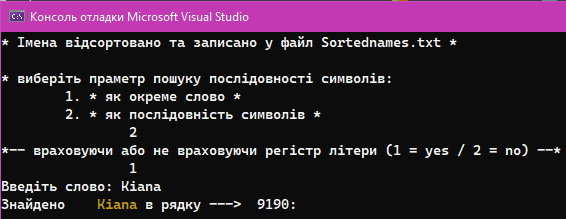


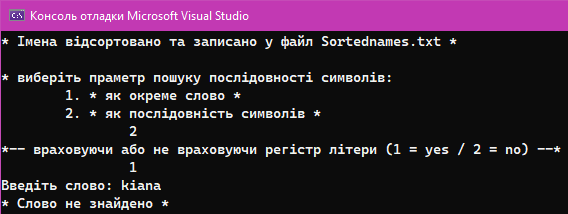
Скріншоти перевірки результату пошуку **«Як окремого слова»** БЕЗ врахуванням регістру літери:



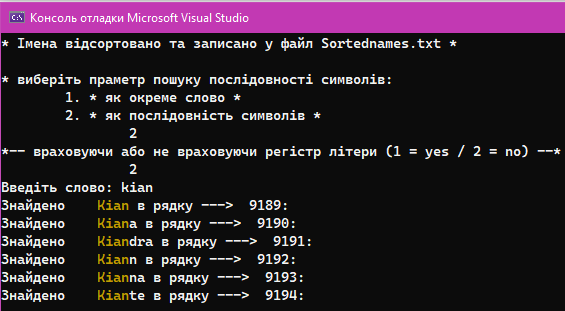


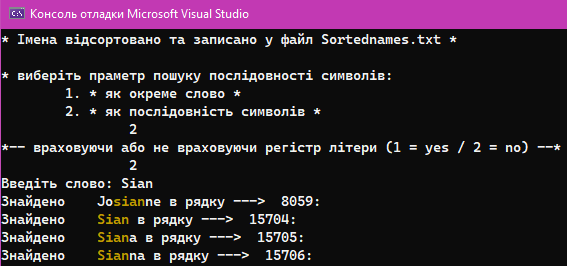
3. Скріншоти перевірки результату пошуку **«Як послідовності символів»** «З врахування регістру літери»:





Скріншоти перевірки результату пошуку **«Як послідовності символів»** «БЕЗ врахування регістру літери»:





**Висновок.** Під час виконання цієї лабораторної роботи були отримані практичні навички в обробці масивів та пошуку елементів у масивах різними методами. Досліджені і вивчені методи пошуку ключових елементів у масивах.