Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №12

за 2 семестр

По дисциплине: «ОАиП»

Тема: «Индексные файлы»

Выполнил:

Студентка 1 курса

Группы ПО-6(1)

Мартынович Д.

Проверил:

Хацкевич М. В.

2021

Индексные файлы

Цель работы: изучить принципы программирования с использованием индексных файлов.

Вариант 24

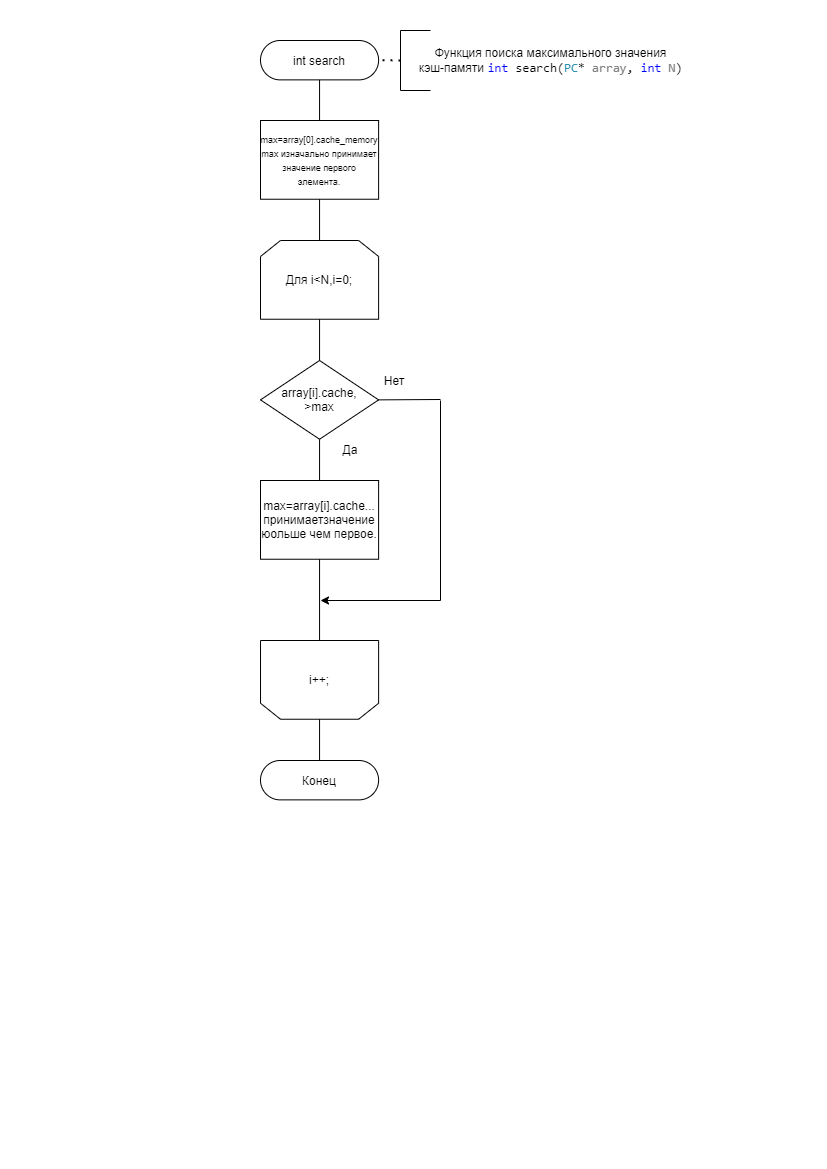
**Задание:**

В программу, разработанную в лабораторной работе 11, внести следующие изменения и дополнения:

1. При запуске программы данные читаются в массив структур из файла, при добавлении новой записи в массив структур в файл должна дописываться новая запись, без изменения остальных записей.

2. Все изменения (изменения полей записи, удаление записи) – сохраняются в файле при помощи перезаписи содержимого всего файла.

3. Сортировка должна выполняться по двум полям на выбор при помощи создания индексных файлов. Содержимое индексного файла переписывается в случае изменения значения ключевого поля (поля, по которому выполняется сортировка) или в случае удаления, добавления записей.

Блок схема:

Текст программы:

#include <iostream>

#include <windows.h>

#include <fstream>

#include <iomanip>

using namespace std;

struct PC

{

char processor[20];

float GHZ;

int Yadra;

int cache\_memory;

int num;

union

{

int map;

int crash;

};

};

int menu(); //организация меню

void input(PC\*& array, int& N); //ввод данных массива структуры

void output(PC\* array, int N,int key, char\* name); //вывод данных массива структур

void sort(PC\* array, int N, int key, char\* name); //сортировка во алфавиту по фамилии

int search(PC\* array, int N); //поиск по заданному параметру

void deleted(PC\*& array, int& N, int key, char\* name); //удаление структуры из массива структур

void write(PC\* array, int N, int key, char\* name);

void read(PC\*& array, int& N, int key, char\* name);

void add(PC\*& array, int& N, int key, char\* name);

void index\_file(int N, int\* index);

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

menu();

return 0;

}

int menu() {

int choise, N = 0;

PC\* array = new PC[N];

int key;

cout << "Работать с текстовым(1) или бинарным файлом(0)?:\n";

cin >> key;

char name[10] = "text.txt";

system("cls");

do {

cout << "Выберите действие:\n";

cout << "1) Ввод массива структур\n";

cout << "2) Вывод массива структур\n";

cout << "3) Сортировка массива структур по алфавит\n";

cout << "4) Удаление заданной структуры\n";

cout << "5) Добавление новых элементов\n";

cout << "6) Поиск максимального значения кэш-памяти\n";

cout << "7) Выход\n";

cin >> choise;

system("cls");

switch (choise)

{

case 1:

if (key == 1) {

read(array, N,key,name);

}

else if (key == 0) {

read(array, N,key,name);

}

break;

case 2:

cin.ignore();

output(array, N,key,name);

break;

case 3:

cin.ignore();

sort(array, N, key,name);

break;

case 4:

cin.ignore();

deleted(array, N, key,name);

break;

case 5:

cin.ignore();

if (key == 1) {

add(array, N,key,name);

}

else if (key == 0) {

add(array, N,key,name);

}

break;

case 6:

cin.ignore();

search(array, N);

break;

case 7: return 0;

}

cin.ignore();

system("cls");

} while (true);

delete[] array;

}

void input(PC\*& array, int& N) {

int n;

cout << "Введите количество структур: \n";

cin >> n;

N = N + n;

PC\* tempArr = new PC[N];

for (int i = 0; i < N - n; i++) {

tempArr[i] = array[i];

}

array = new PC[N];

for (int i = 0; i < N - n; i++) {

array[i] = tempArr[i];

}

delete tempArr;

system("cls");

cin.ignore();

for (int i = N - n; i < N; i++) {

cout << "1) Название процессора: ";

cin >> (array[i].processor);

cout << "2) ГГц: ";

cin >> (array[i].GHZ);

cout << "3) Кол-во ядер: ";

cin >> (array[i].Yadra);

cout << "4) Кэш-память: ";

cin >> (array[i].cache\_memory);

cin.ignore();

system("cls");

}

}

void output(PC\* array, int N, int key, char\* name)

{

cout << setw(15) << left << "№";

cout << setw(15) << left << "Процессор";

cout << setw(15) << left << "Ггц";

cout << setw(15) << left << "Ядра";

cout << setw(15) << left << "Память";

for (int i = 0; i < N; i++)

{

cout << endl;

cout << setw(15) << left << i + 1;

cout << setw(15) << left << array[i].processor;

cout << setw(15) << left << array[i].GHZ;

cout << setw(15) << left << array[i].Yadra;

cout << setw(15) << left << array[i].cache\_memory;

}

}

void sort(PC\* array, int N, int key, char\* name) {

cout << "Выберите пункт:\n";

cout << "1 - Сортировка по названию \n";

cout << "0 - Сортировка по количеству ядер\n";

int choise;

cin >> choise;

int\* index = new int[N];

for (int i = 0; i < N; i++) {

index[i] = i;

}

for (int i = 0; i < N - 1; i++) {

for (int j = i; j < N; j++) {

if (choise == 1) {

if (strcmp(array[index[j]].processor, array[index[i]].processor) < 0) {

swap(index[i], index[j]);

}

}

else if (choise == 0) {

if (array[index[j]].Yadra < array[index[i]].Yadra) {

swap(index[i], index[j]);

}

}

}

}

index\_file(N, index);

write(array, N, key, name);

output(array, N, key, name);

cin.ignore();

}

void index\_file(int N, int\* index) {

ofstream fout("Indexes.txt");

for (int i = 0; i < N; i++) {

fout << i + 1 << " " << index[i];

if (i < N - 1) {

fout << endl;

}

}

fout.close();

}

{

int max = array[0].cache\_memory, max\_ind = 0;

for (int i = 1; i < N; i++) {

if (array[i].cache\_memory > max)

max = array[i].cache\_memory;

}

return max;

}

void deleted(PC\*& array, int& N, int key, char\* name)

{

char Name[20];

cout << "Введите название страны, которую хотите удалить.\n";

gets\_s(Name);

for (int i = 0; i < N; i++) {

if (strcmp(array[i].processor, Name) == 0) {

N--;

for (int j = i; j < N; j++) {

array[j] = array[j + 1];

}

i--;

}

}

PC\* tempArr = new PC[N];

for (int i = 0; i < N; i++) {

tempArr[i] = array[i];

}

array = new PC[N];

for (int i = 0; i < N; i++) {

array[i] = tempArr[i];

}

delete[] tempArr;

system("cls");

int\* tempArray = new int[N];

for (int i = 0; i < N; i++) {

tempArray[i] = i;

}

index\_file(N, tempArray);

delete[] tempArray;

write(array, N, key, name);

output(array, N, key, name);

}

void write(PC \* array, int N, int key, char\* name)

{

ofstream file;

if (key == 1) {

file.open(name);

}

else if (key == 0) {

file.open(name, ios\_base::binary);

}

ifstream index\_file;

index\_file.open("Indexes.txt");

if (key == 1) {

for (int i = 0; i < N; i++)

{

int k;

if (index\_file.is\_open() && !index\_file.eof())

{

index\_file >> k;

index\_file >> k;

}

else k = i;

file << setw(15) << left << array[k].processor;

file << setw(15) << left << array[k].GHZ;

file << setw(15) << left << array[k].Yadra;

file << setw(15) << left << array[k].cache\_memory;

if (i < N - 1)

{

file << endl;

}

}

}

else if (key == 0) {

for (int i = 0; i < N; i++) {

int k;

if (index\_file.is\_open()) {

index\_file >> k;

index\_file >> k;

}

else k = i;

file.write((char\*) & array[k], sizeof(array[k]));

}

}

index\_file.close();

file.close();

}

void read(PC \* &array, int& N, int key, char\* name) {

ifstream file\_txt;

file\_txt.open(name);

N = 0;

if (!file\_txt.is\_open()) {

input(array, N);

write(array, N,key,name);

}

else {

array = new PC[N];

char temp[1024];

file\_txt.getline(temp, 1024, '\n');

for (int i = 0; !file\_txt.eof(); i++) {

N++;

PC\* tempArr = new PC[N - 1];

for (int j = 0; j < N - 1; j++) {

tempArr[j] = array[j];

}

array = new PC[N];

for (int j = 0; j < N - 1; j++) {

array[j] = tempArr[j];

}

delete tempArr;

file\_txt >> array[i].processor >> array[i].GHZ >> array[i].Yadra >> array[i].cache\_memory;

}

cout << "Данные считаны из файла.\n";

cin.ignore();

file\_txt.close();

}

}

void add(PC \* &array, int& N, int key, char\* name) {

ofstream file;

if (key == 1) {

file.open(name, ios\_base::app);

}

else if (key == 0) {

file.open(name, ios\_base::app | ios\_base::binary);

}

input(array, N);

file << endl;

write(array, N, key, name);

output(array, N, key, name);

file.close();

int\* tempArr = new int[N];

for (int i = 0; i < N; i++) {

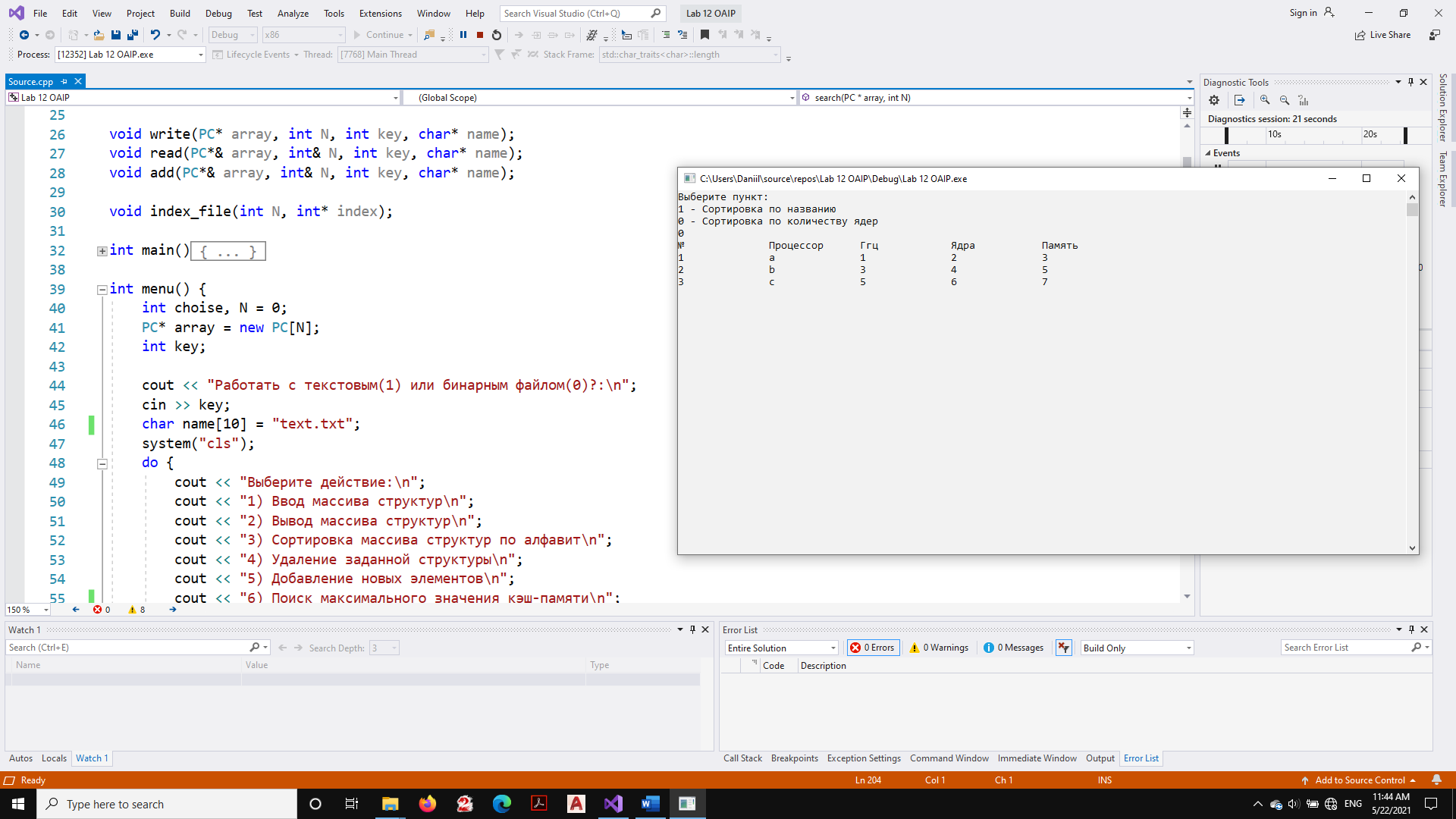
tempArr[i] = array[i].num;

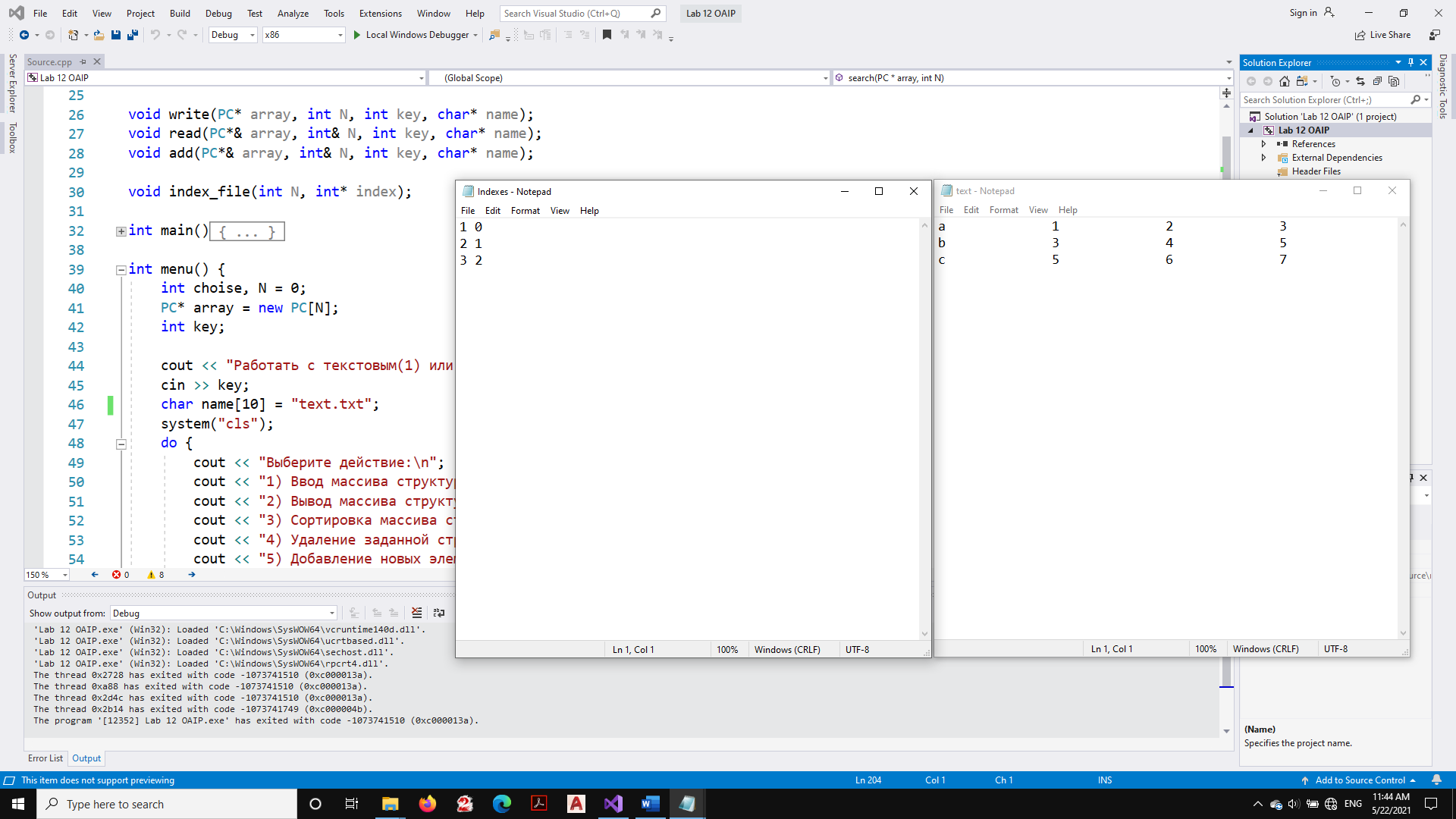
}

index\_file(N, tempArr);

delete[] tempArr;

}





Вывод: Изучил использование индексных файлов в програмировании.