**мІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**нАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «лЬВІВСЬКА ПОЛІТЕХНІКА»**

**ІНСТИТУТ КОМП’ЮТЕРНИХ НАУК ТА ІНОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

**Кафедра ІСМ**

|  |
| --- |
| **2020** |



**ЗВІТ**

**Про виконання лабораторної роботи № 9.2 А**

**« Впорядкування та бінарний пошук в масиві структур»**

**З дисципліни**

**«Алгоритмізація та програмування»**

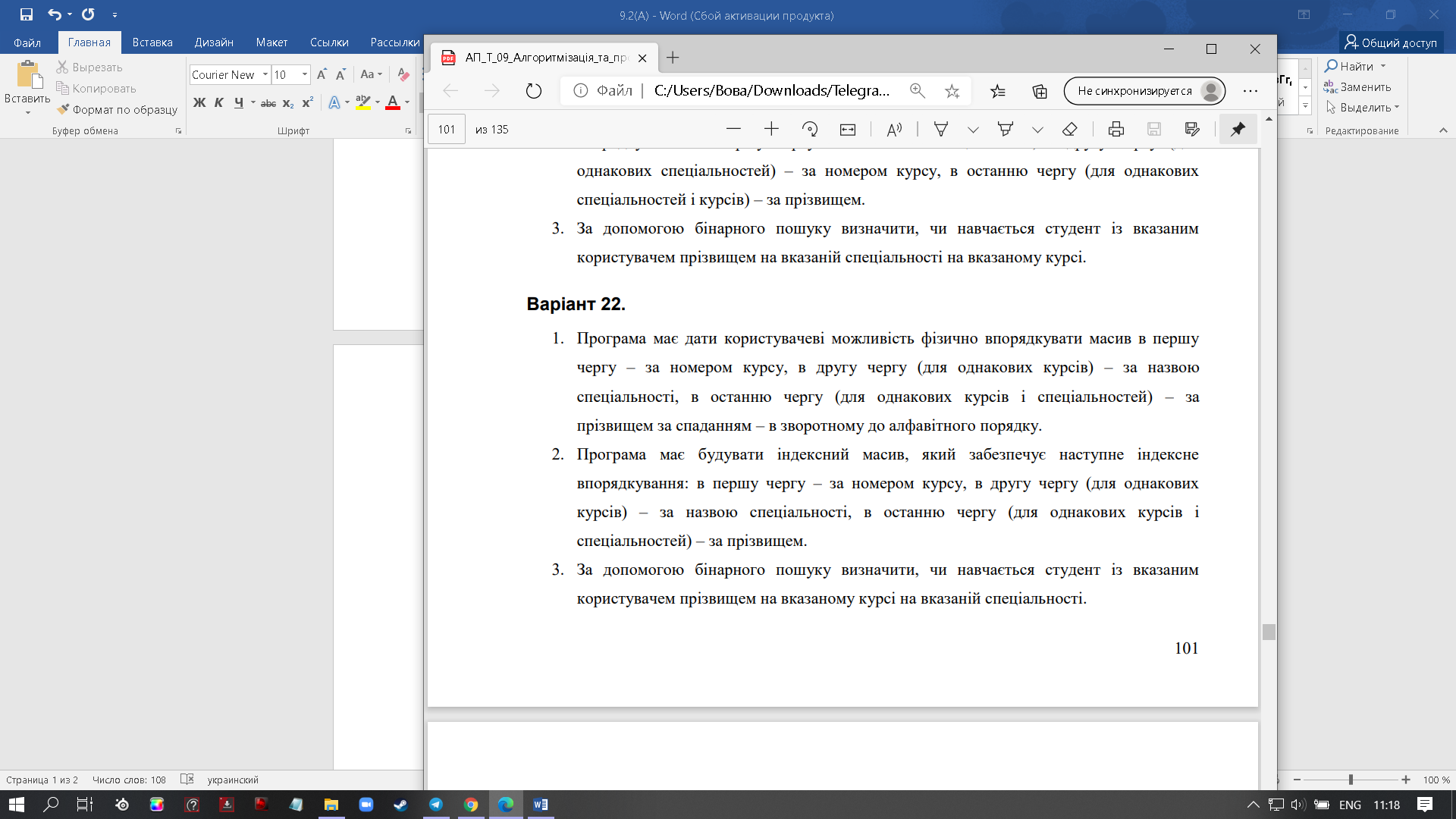
**Студента групи ІТ-12**

**Шкринди Володимира**

**Мета роботи**

Навчитися впорядковувати масив структур з об’єднаннями. Навчитися здійснювати фізичне та індексне впорядкування. Навчитися здійснювати бінарний пошук у фізично чи індексно впорядкованому масиві.

**Умова завдання**



**Текст програми**

// A

#include <iostream>

#include <iomanip>

#include <string>

#include <Windows.h>

using namespace std;

enum Spec { KN, INF, ME, FI, TN };

string SpecStr[] = { "Комп.наука", "Інформатика", "Математика та Економіка", "Фізика та Інформатика", "Трудове навчання" };

struct Student

{

string prizv;

int kurs;

Spec spec;

int physics;

int math;

int inform;

};

void Create(Student\* p, const int N);

void Print(Student\* p, const int N);

void Sort(Student\* p, const int N);

int\* IndexSort(Student\* p, const int N);

void PrintIndexSorted(Student\* p, int\* I, const int N);

int BinSearch(Student\* p, const int N, const string prizv, const Spec spec,const int kurs);

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int N;

cout << "Введіть кількість студентів N: "; cin >> N;

Student\* p = new Student[N];

int ispec;

Spec spec;

string prizv;

int found;

int kurs;

int menuItem;

do {

cout << endl << endl << endl;

cout << "Виберіть дію:" << endl << endl;

cout << " [1] - введення даних з клавіатури" << endl;

cout << " [2] - вивід даних на екран" << endl;

cout << "[3] - індекстне впорядкування даних" << endl;

cout << " [4] - фізичне впорядкування даних" << endl;

cout << " [5] - бінарний пошук студента за спеціальністю та прізвищем і курсом"

<< endl << endl;

cout << " [0] - вихід та завершення роботи програми" << endl << endl;

cout << "Введіть значення: "; cin >> menuItem;

cout << endl << endl << endl;

switch (menuItem)

{

case 1:

Create(p, N);

break;

case 2:

Print(p, N);

break;

case 3:

PrintIndexSorted(p, IndexSort(p, N), N);

break;

case 4:

Sort(p, N);

Print(p, N);

break;

case 5:

cout << "Введіть ключі пошуку:" << endl;

cout << " Спеціальність: (0 - Комп.науки, 1 -Інформатика , 2 - Математика та економіка, 3 - Фізика та Інформатика,4 - Трудове навчання): ";

cin >> ispec;

spec = (Spec)ispec;

cin.get();

cin.sync();

cout << endl;

cout << " прізвище: "; getline(cin, prizv);

cout << endl;

cout << "Курс:";cin >> kurs;

if ((found = BinSearch(p, N, prizv, spec, kurs)) != -1)

cout << "Знайдено студента в позиції: " << found + 1 << endl;

else

cout << "Шуканого студента не знайдено" << endl;

break;

case 0:

break;

default:

cout << "Ви ввели помилкове значення! "

"Слід ввести число - номер вибраного пункту меню" << endl;

}

} while (menuItem != 0);

return 0;

}

void Create(Student\* p, const int N)

{

int spec;

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

{

cout << "Студент № " << i + 1 << ":" << endl;

cin.get();

cin.sync();

cout << " прізвище: "; getline(cin, p[i].prizv);

cout << " курс: "; cin >> p[i].kurs;

cout << " Спеціальність: (0 - Комп.науки, 1 -Інформатика , 2 - Математика та економіка, 3 - Фізика та Інформатика,4 - Трудове навчання): ";

cin >> spec;

p[i].spec = (Spec)spec;

cout << " Оцінка по фізиці: ";cin >> p[i].physics;

cout << " Оцінка по математиці: ";cin >> p[i].math;

cout << " Оцінка по інформатиці: ";cin >> p[i].inform;

}

}

void Print(Student\* p, const int N)

{

cout << "========================================================================="

<< endl;

cout << "| № | Прізвище | Курс | Спеціальність | Фізика | Математика | Інформатика|"

<< endl;

cout << "-------------------------------------------------------------------------"

<< endl;

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

{

cout << "|" << setw(3) << right << i + 1 << " ";

cout << "| " << setw(6) << left << p[i].prizv << "| " << setw(4) << right << p[i].kurs << " "

<< "| " << setw(6) << left << SpecStr[p[i].spec] << setw(4);

cout << " |" << setw(4) << right << p[i].physics << setw(4) << right << "| ";

cout << setw(4) << right << p[i].math << setw(4) << right << " | ";

cout << setw(4) << right << p[i].inform << setw(1) << right << " |" << endl;

cout << "========================================================================="

<< endl;

cout << endl;

}

}

void Sort(Student\* p, const int N)

{

Student tmp;

for (int i0 = 0; i0 < N - 1; i0++)

for (int i1 = 0; i1 < N - i0 - 1; i1++)

if ((p[i1].kurs > p[i1 + 1].kurs)

||

(p[i1].kurs == p[i1 + 1].kurs &&

p[i1].spec > p[i1 + 1].spec)

||

(p[i1].kurs == p[i1 + 1].kurs &&

p[i1].spec == p[i1 + 1].spec) && p[i1].prizv < p[i1 + 1].prizv)

{

tmp = p[i1];

p[i1] = p[i1 + 1];

p[i1 + 1] = tmp;

}

}

// INDEX

int\* IndexSort(Student\* p, const int N)

{

int\* I = new int[N];

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

I[i] = i;

int i, j, value;

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

{

value = I[i];

for (j = i - 1;

j >= 0 && ((p[I[j]].kurs > p[value].kurs) ||

(p[I[j]].kurs == p[value].kurs &&

p[I[j]].spec > p[value].spec)

||

(p[I[j]].kurs == p[value].kurs &&

p[I[j]].spec == p[value].spec && p[I[j]].prizv > p[value].prizv));

j--)

{

I[j + 1] = I[j];

}

I[j + 1] = value;

}

return I;

}

void PrintIndexSorted(Student \* p, int\* I, const int N)

{

cout << "=============================================================================="

<< endl;

cout << "| № | Прізвище | Курс | Спеціальність | Фізика | Математика | Інформатика |"

<< endl;

cout << "------------------------------------------------------------------------------"

<< endl;

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

{

cout << "|" << setw(3) << right << i + 1 << " ";

cout << "| " << setw(6) << left << p[I[i]].prizv << "| " << setw(4) << right << p[I[i]].kurs << " "

<< "| " << setw(6) << left << SpecStr[p[I[i]].spec] << setw(4);

cout << " |" << setw(4) << right << p[I[i]].physics << setw(4) << right << "| ";

cout << setw(4) << right << p[I[i]].math << setw(4) << right << " | ";

cout << setw(4) << right << p[I[i]].inform << setw(1) << right << " |" << endl;

cout << "========================================================================="

<< endl;

cout << endl;

}

}

int BinSearch(Student\* p, const int N, const string prizv, const Spec spec, const int kurs)

{

int L = 0, R = N - 1, m;

do {

m = (L + R) / 2;

if (p[m].prizv == prizv && p[m].spec == spec && p[m].kurs == kurs)

return m;

if ((p[m].spec < spec)

||

(p[m].spec == spec &&

p[m].prizv < prizv)

||

(p[m].spec == spec &&

p[m].prizv == prizv && p[m].kurs < kurs))

{

L = m + 1;

}

else

{

R = m - 1;

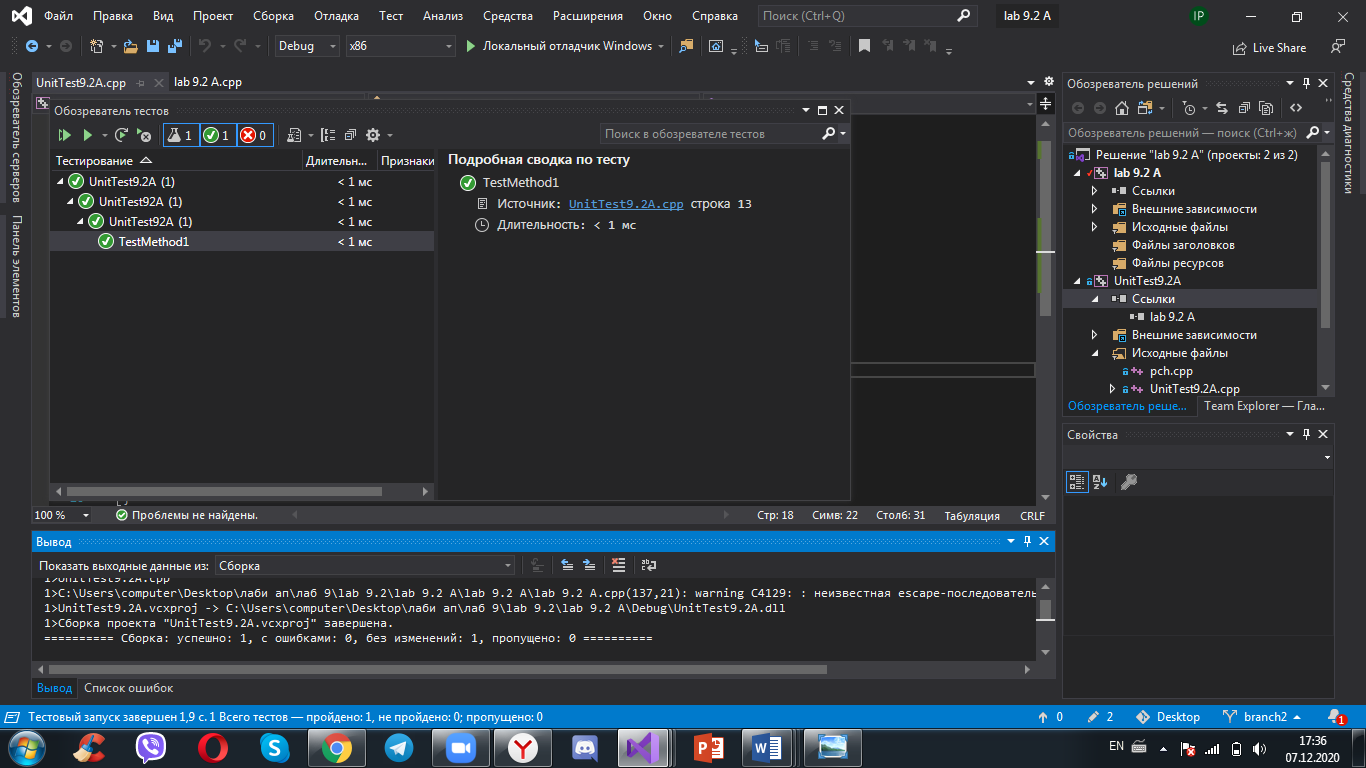
}

} while (L <= R);

return -1;

}

**Unit-test**



#include "pch.h"

#include "CppUnitTest.h"

#include "../lab 9.2(A)/lab 9.2(A).cpp"

using namespace Microsoft::VisualStudio::CppUnitTestFramework;

namespace UnitTest92A

{

TEST\_CLASS(UnitTest92A)

{

public:

TEST\_METHOD(TestMethod1)

{

Student\* p = new Student[1];

p->prizv = "Matus";

p->inform = 5;

p->math = 4;

p->physics = 3;

bool functionResult = BinSearch(p, 1, "Matus", 3, 4);

Assert::AreEqual(false, functionResult);

}

};

}

**Cтрукутурна схема**

**Посилання на git-репозиторій з проектом:**

**Висновок:** на цій лабораторній роботі я навчився впорядковувати масив структур з об’єднаннями. Навчитися здійснювати фізичне та індексне впорядкування. Навчитися здійснювати бінарний пошук у фізично чи індексно впорядкованому масиві.