**мІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**нАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «лЬВІВСЬКА ПОЛІТЕХНІКА»**

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

**Кафедра ІСМ**

|  |
| --- |
| **2022** |



**ЗВІТ**

**Про виконання лабораторної роботи №3.2**

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

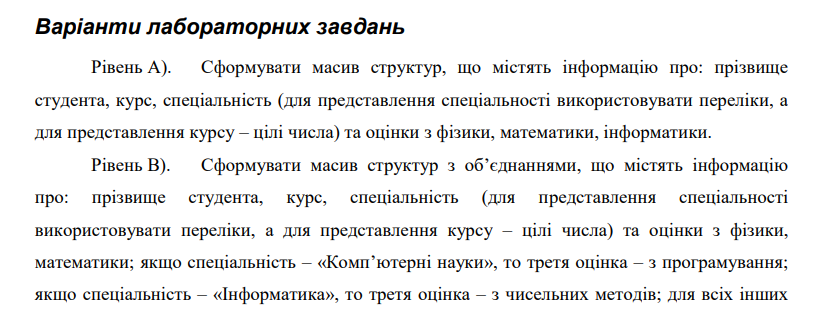
**З дисципліни**

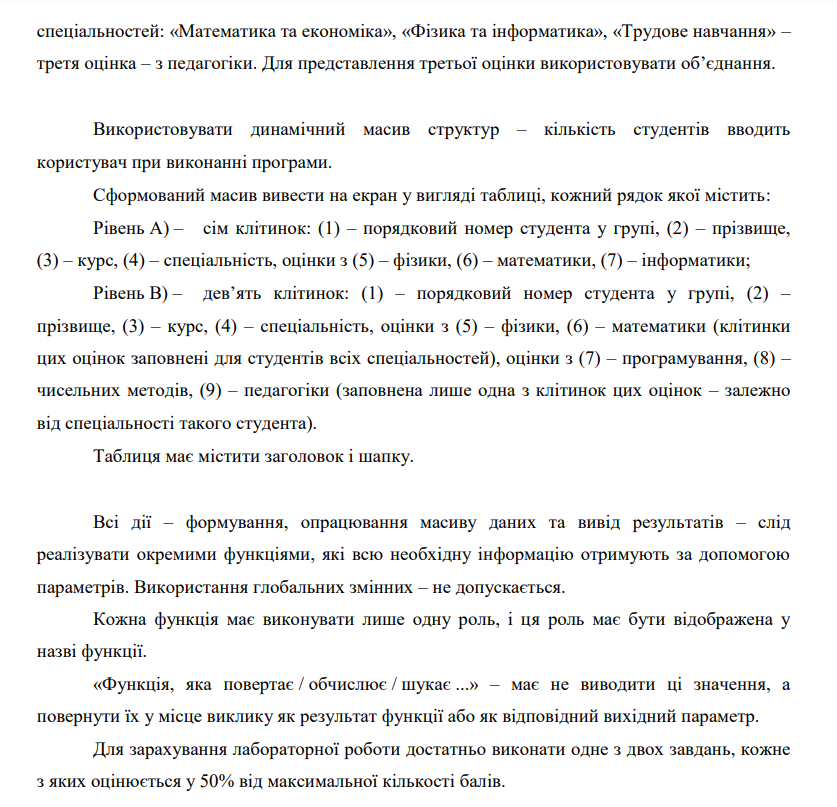
**«Програмування та командна робота»**

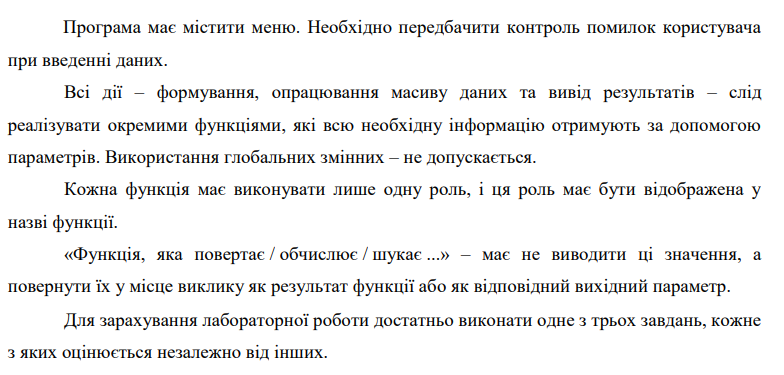
**Студента групи ІТ-11**

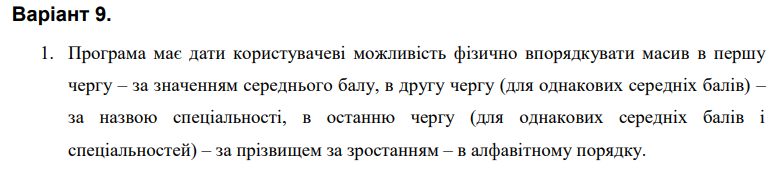
**Лисецького Володимира Любомировича**

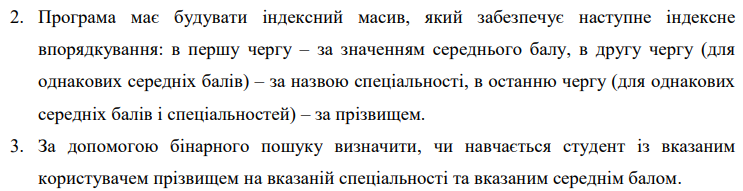
**Умова завдання:**

****

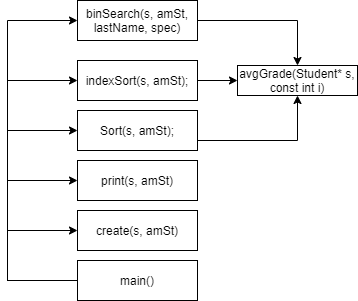


****

****

****

**Структурна схема програми:**

****

**Текст програми:**

#include <iostream>

#include <string>

#include <Windows.h>

#include <iomanip>

using namespace std;

enum Spec { CS, IT, ME, PhIT, TR };

string strSpec[] = { "Комп'ютернi науки", "Iнформатика", "Математика та економiка", "Фiзика та iнформатика", "Трудове навчання" };

struct Student {

string lname;

int kurs;

Spec spec;

int physics;

int math;

union {

int progr;

int numb;

int ped;

};

};

void create(Student\* s, int amSt) {

int sp;

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

cout << "Прiзвище: ";

cin >> s[i].lname;

cout << "Курс: ";

cin >> s[i].kurs;

cout << "Спецiальнiсть (0 - Комп'ютернi науки, 1 - Iнформатика, 2 - Математика та економiка, 3 - Фiзика та iнформатика, 4 - Трудове навчання) ";

cin >> sp;

s[i].spec = (Spec)sp;

cout << "Оцiнка з фiзики: ";

cin >> s[i].physics;

cout << "Оцiнка з математики: ";

cin >> s[i].math;

if (s[i].spec == 0) {

cout << "Оцiнка з програмування: ";

cin >> s[i].progr;

}

else if (s[i].spec == 1) {

cout << "Оцiнка з чисельних методiв: ";

cin >> s[i].numb;

}

else {

cout << "Оцiнка з педагогiки: ";

cin >> s[i].ped;

}

}

}

void print(Student\* s, int amSt)

{

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

<< endl;

cout << "| № | Прiзвище | Курс | Спецiальнiсть | Фiзика | Математика | Програмування | Чисельнi методи | Педагогiка |"

<< endl;

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

<< endl;

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

cout << "| " << setw(2) << right << i + 1 << "";

cout << "| " << setw(14) << left << s[i].lname

<< "| " << setw(5) << right << s[i].kurs

<< "| " << setw(23) << left << strSpec[s[i].spec]

<< "| " << setw(7) << s[i].physics

<< "| " << setw(11) << right << s[i].math;

if (s[i].spec == 0) {

cout << "| " << setw(14) << right << s[i].progr

<< "| " << setw(16) << right << " "

<< "| " << setw(11) << right << " " << "| ";

}

else if (s[i].spec == 1) {

cout << "| " << setw(14) << right << " "

<< "| " << setw(16) << right << s[i].numb

<< "| " << setw(11) << right << " " << "| ";

}

else {

cout << "| " << setw(14) << right << " "

<< "| " << setw(16) << right << " "

<< "| " << setw(11) << right << s[i].ped << "| ";

}

cout << endl;

}

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

<< endl;

}

double avgGrade(Student\* s, const int i) {

double avg = 0;

if (s[i].spec == 0)

avg = (1.0 \* (s[i].physics + s[i].math + s[i].progr)) / 3;

else if (s[i].spec == 1)

avg = (1.0 \* (s[i].physics + s[i].math + s[i].numb)) / 3;

else

avg = (1.0 \* (s[i].physics + s[i].math + s[i].ped)) / 3;

return avg;

}

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

{

Student tmp;

for (int i0 = 0; i0 < N - 1; i0++) // метод "бульбашки"

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

if ((avgGrade(p,i1) > avgGrade(p, i1+1))

||

(avgGrade(p, i1) == avgGrade(p, i1+1) &&

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

||

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

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

{

tmp = p[i1];

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

p[i1 + 1] = tmp;

}

}

int\* indexSort(Student \*s, 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 && ((avgGrade(s, I[j]) > avgGrade(s, value))

||

(avgGrade(s, I[j]) == avgGrade(s, value) &&

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

||

(s[I[j]].spec == s[value].spec &&

s[I[j]].lname > s[value].lname));

j--)

{

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

}

I[j + 1] = value;

}

return I;

}

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

{ // повертає індекс знайденого елемента або -1, якщо шуканий елемент відсутній

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

do {

m = (L + R) / 2;

if (p[m].lname == prizv && p[m].spec == spec && avgGrade(p,m) == avgGrade(p, m))

return m;

if ((p[m].lname < prizv)

||

(p[m].lname == prizv &&

p[m].spec < spec)

||

p[m].spec < spec &&

avgGrade(p, m) < avgGrade(p, m))

{

L = m + 1;

}

else

{

R = m - 1;

}

} while (L <= R);

return -1;

}

void printIndexSorted(Student\* s, int\* I, int amSt)

{

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

<< endl;

cout << "| № | Прiзвище | Курс | Спецiальнiсть | Фiзика | Математика | Програмування | Чисельнi методи | Педагогiка |"

<< endl;

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

<< endl;

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

cout << "| " << setw(2) << right << i + 1 << "";

cout << "| " << setw(14) << left << s[I[i]].lname

<< "| " << setw(5) << right << s[I[i]].kurs

<< "| " << setw(23) << left << strSpec[s[I[i]].spec]

<< "| " << setw(7) << s[I[i]].physics

<< "| " << setw(11) << right << s[I[i]].math;

if (s[I[i]].spec == 0) {

cout << "| " << setw(14) << right << s[I[i]].progr

<< "| " << setw(16) << right << " "

<< "| " << setw(11) << right << " " << "| ";

}

else if (s[I[i]].spec == 1) {

cout << "| " << setw(14) << right << " "

<< "| " << setw(16) << right << s[I[i]].numb

<< "| " << setw(11) << right << " " << "| ";

}

else {

cout << "| " << setw(14) << right << " "

<< "| " << setw(16) << right << " "

<< "| " << setw(11) << right << s[I[i]].ped << "| ";

}

cout << endl;

}

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

<< endl;

}

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int amSt;

string lastName;

Spec spec;

int specNo, avgGrade, found;

cout << "Кількість студентів: ";

cin >> amSt;

Student\* s = new Student[amSt];

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] - фізичне впорядкування даних " << endl;

cout << " [4] - індексне впорядкування та вивід даних " << endl;

cout << " [5] - бінарний пошук для визначення, чи навчається студент " << endl;

cout << " із вказаним користувачем прізвищем на вказаній спеціальності та вказаним середнім балом." << endl;

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

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

cout << endl << endl << endl;

switch (menuItem) {

case 1:

create(s, amSt);

break;

case 2:

print(s, amSt);

break;

case 3:

Sort(s, amSt);

break;

case 4:

printIndexSorted(s, indexSort(s, amSt), amSt);

break;

case 5:

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

cout << " Прізвище: "; cin >> lastName;

cout << endl;

cout << " Спецiальнiсть (0 - Комп'ютернi науки, 1 - Iнформатика, 2 - Математика та економiка, 3 - Фiзика та iнформатика, 4 - Трудове навчання) ";

cin >> specNo;

spec = (Spec)specNo;

cout << " Середній бал: " << endl;

cin >> avgGrade;

cin.get(); // очищуємо буфер клавіатури – бо залишаються символи

cin.sync(); // "кінець рядка", які не дають ввести наступний літерний рядок

if ((found = binSearch(s, amSt, lastName, spec)) != -1)

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

else

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

break;

case 0:

break;

}

} while (menuItem != 0);

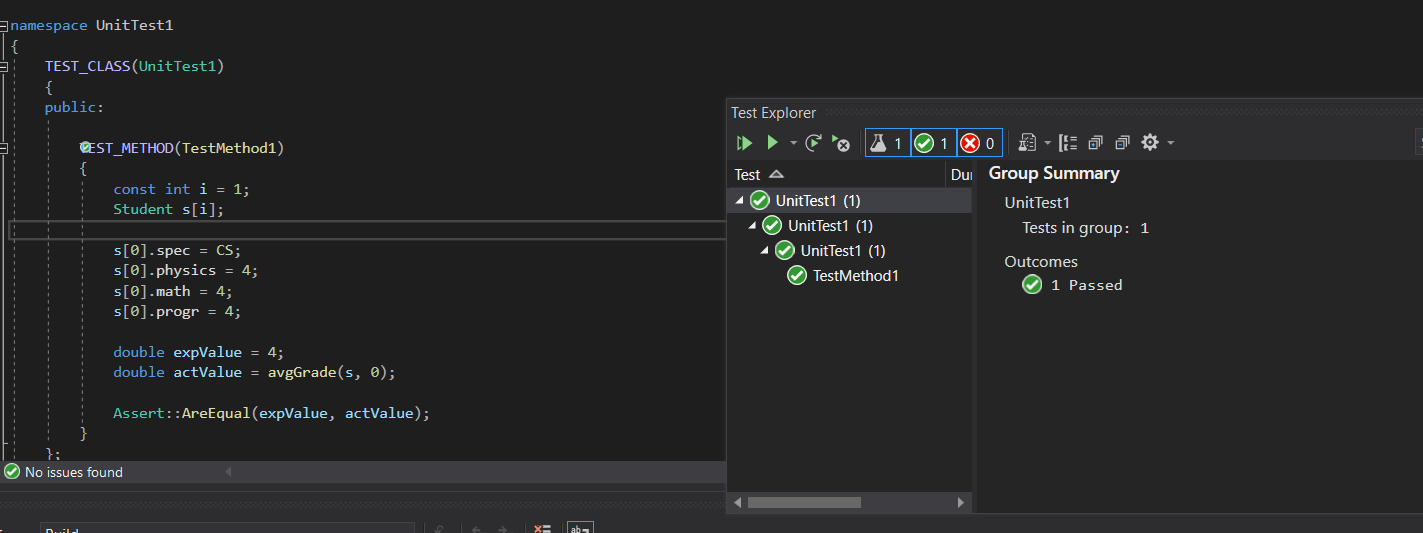
return 0;

}

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

**https://github.com/MarshmallowSoup/ptw-lab-3-2**

**Результати unit-тесту:**

****

**Висновки:** ми навчилися впорядковувати масив структур з об’єднаннями, а також здійснювати фізичне та індексне впорядкування. Навчитися здійснювати бінарний пошук у фізично чи індексно впорядкованому масиві.