ЗВІТ

Про виконання лабораторної роботи №10.2A

*«Впорядкування та бінарний пошук в масиві структур»*

з дисципліни

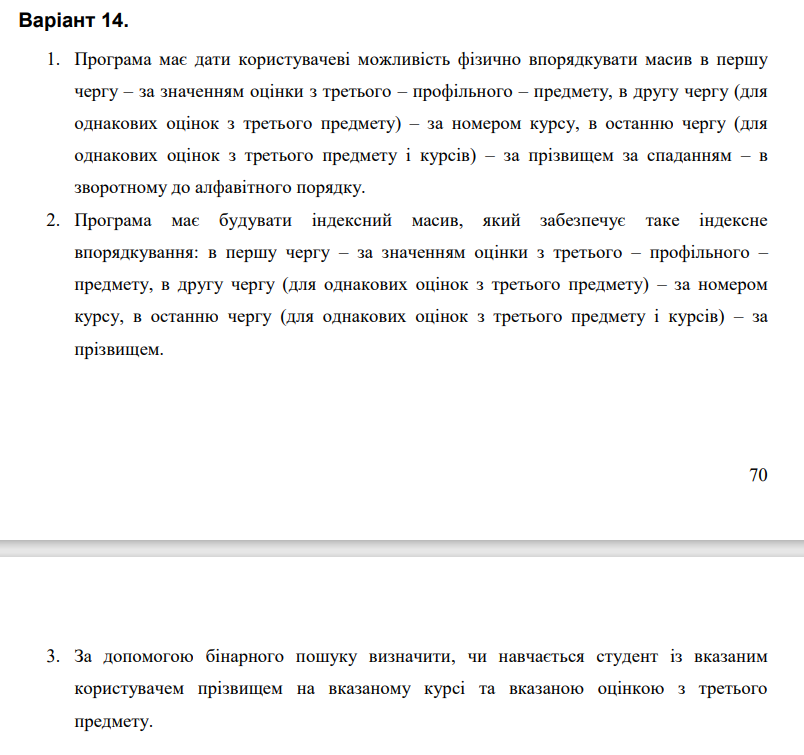
«Алгоритмізація та програмування»

студента групи РІ-12

*Синчук Іван Романович*

Зображення, що містить текст, Шрифт, знімок екрана, ряд

Автоматично згенерований опис



#include <iostream>

#include <iomanip>

#include <string>

#include <algorithm>

#include <Windows.h>

using namespace std;

enum Specialization { COMPUTER\_SCIENCE, MATHEMATICS, PHYSICS, ENGINEERING, BIOLOGY };

string specializationStr[] = { "Комп'ютерні науки", "Математика", "Фізика", "Інженерія", "Біологія" };

struct Student {

string surname;

unsigned course;

Specialization specialization;

int physics;

int mathematics;

int informatics;

};

void InputStudents(Student\* students, int N);

void PrintStudents(const Student\* students, int N);

double CalculateAverage(const Student& student);

void SortStudents(Student\* students, int N);

void IndexSortStudents(Student\* students, int N, int\* indexArray);

int BinarySearch(const Student\* students, int\* indexArray, int N, const string& surname, int course, int informaticsGrade);

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int numStudents;

cout << "Введіть кількість студентів: ";

cin >> numStudents;

Student\* students = new Student[numStudents];

int\* indexArray = new int[numStudents];

int menuItem;

do {

cout << "\nМеню:\n";

cout << "1 - Введення даних студентів\n";

cout << "2 - Вивід даних студентів\n";

cout << "3 - Фізичне сортування студентів\n";

cout << "4 - Індексне сортування студентів\n";

cout << "5 - Пошук студента\n";

cout << "0 - Вихід\n";

cout << "Виберіть дію: ";

cin >> menuItem;

switch (menuItem) {

case 1:

InputStudents(students, numStudents);

break;

case 2:

PrintStudents(students, numStudents);

break;

case 3:

SortStudents(students, numStudents);

cout << "Список студентів фізично відсортовано.\n";

break;

case 4:

IndexSortStudents(students, numStudents, indexArray);

cout << "Список студентів індексно відсортовано.\n";

break;

case 5: {

string surname;

int course, informaticsGrade;

cout << "Введіть прізвище для пошуку: ";

cin >> surname;

cout << "Введіть курс для пошуку: ";

cin >> course;

cout << "Введіть оцінку з інформатики для пошуку: ";

cin >> informaticsGrade;

int result = BinarySearch(students, indexArray, numStudents, surname, course, informaticsGrade);

if (result != -1)

cout << "Студент знайдений за індексом: " << result << "\n";

else

cout << "Студент не знайдений.\n";

break;

}

case 0:

break;

default:

cout << "Неправильний вибір. Спробуйте ще раз.\n";

}

} while (menuItem != 0);

delete[] students;

delete[] indexArray;

return 0;

}

void InputStudents(Student\* students, int N) {

int spec;

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

cout << "Студент № " << i + 1 << ":\n";

cin.ignore();

cout << " Прізвище: "; getline(cin, students[i].surname);

cout << " Курс: "; cin >> students[i].course;

cout << " Спеціальність (0 - Комп'ютерні науки, 1 - Математика, 2 - Фізика, 3 - Інженерія, 4 - Біологія): ";

cin >> spec;

students[i].specialization = static\_cast<Specialization>(spec);

cout << " Оцінка з фізики: "; cin >> students[i].physics;

cout << " Оцінка з математики: "; cin >> students[i].mathematics;

cout << " Оцінка з інформатики: "; cin >> students[i].informatics;

}

}

void PrintStudents(const Student\* students, int N) {

cout << "=============================================================================================\n";

cout << "| № | Прізвище | Курс | Спеціальність | Фізика | Математика | Інформатика |\n";

cout << "---------------------------------------------------------------------------------------------\n";

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

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

cout << setw(15) << left << students[i].surname;

cout << " | " << setw(4) << right << students[i].course;

cout << " | " << setw(21) << left << specializationStr[students[i].specialization];

cout << " | " << setw(6) << right << students[i].physics;

cout << " | " << setw(10) << right << students[i].mathematics;

cout << " | " << setw(11) << right << students[i].informatics << " |\n";

}

cout << "=============================================================================================\n";

}

double CalculateAverage(const Student& student) {

return (student.physics + student.mathematics + student.informatics) / 3.0;

}

void SortStudents(Student\* students, int N) {

sort(students, students + N, [](const Student& a, const Student& b) {

if (a.informatics != b.informatics)

return a.informatics > b.informatics;

if (a.course != b.course)

return a.course > b.course;

return a.surname > b.surname;

});

}

void IndexSortStudents(Student\* students, int N, int\* indexArray) {

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

indexArray[i] = i;

}

sort(indexArray, indexArray + N, [&](int a, int b) {

if (students[a].informatics != students[b].informatics)

return students[a].informatics > students[b].informatics;

if (students[a].course != students[b].course)

return students[a].course > students[b].course;

return students[a].surname > students[b].surname;

});

}

int BinarySearch(const Student\* students, int\* indexArray, int N, const string& surname, int course, int informaticsGrade) {

int left = 0, right = N - 1;

while (left <= right) {

int mid = left + (right - left) / 2;

int idx = indexArray[mid];

if (students[idx].informatics == informaticsGrade && students[idx].course == course && students[idx].surname == surname)

return idx;

if (students[idx].informatics > informaticsGrade ||

(students[idx].informatics == informaticsGrade && students[idx].course > course) ||

(students[idx].informatics == informaticsGrade && students[idx].course == course && students[idx].surname > surname)) {

left = mid + 1;

}

else {

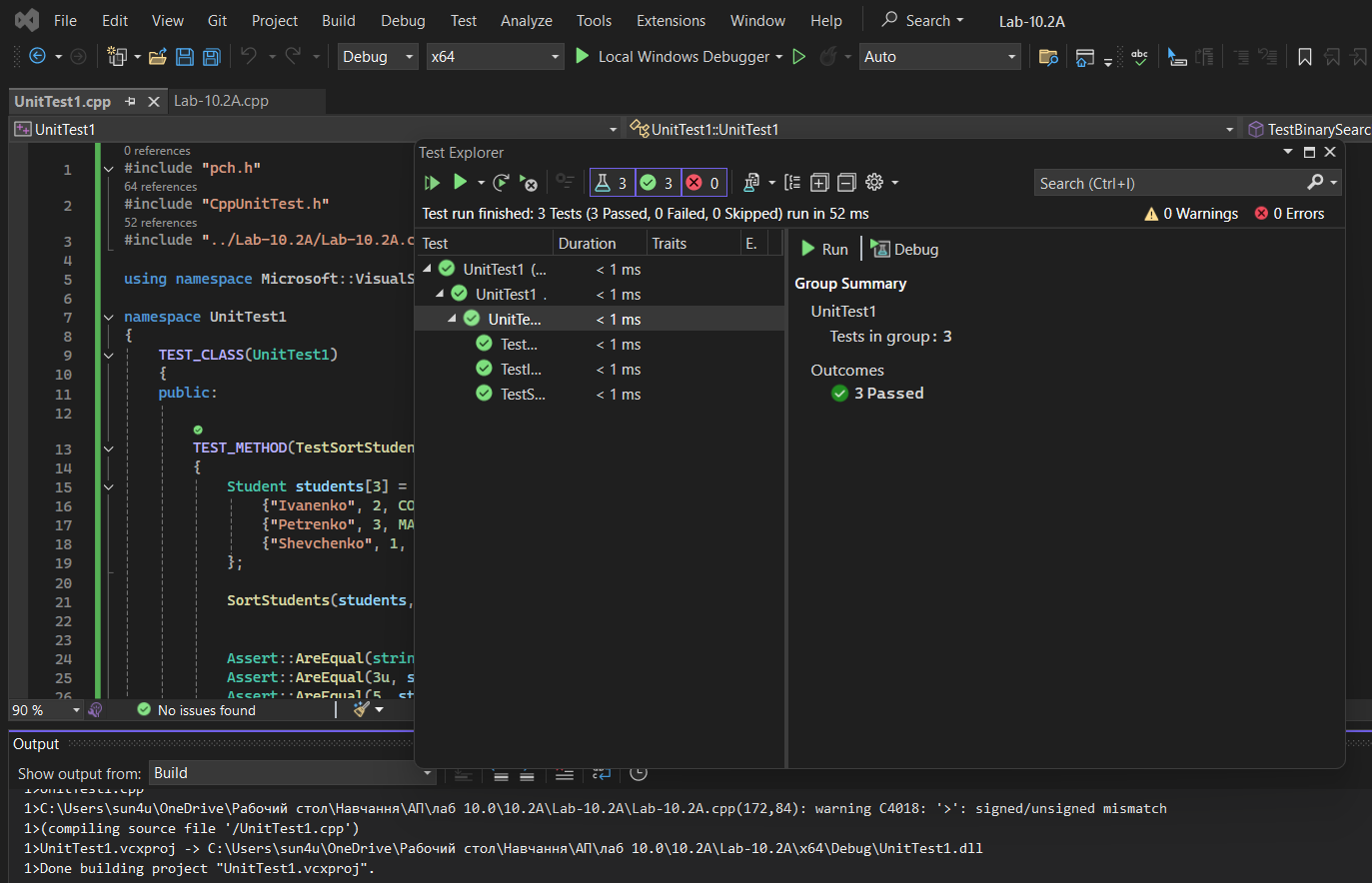
right = mid - 1;

}

}

return -1;

}



GitHub:

Висновки: На цій лабораторній я навчився впорядковувати масив структур з об’єднаннями. Навчитися здійснювати фізичне та індексне впорядкування. Навчився здійснювати бінарний пошук у фізично чи індексно впорядкованому масиві.