Міністерство освіти і науки України

Вінницький національний технічний університет

Факультет комп’ютерних систем та автоматики

Кафедра КСУ

Лабораторна робота №1

з дисципліни: «Компютерні технології та програмування»

Тема: «Структури»

Виконав

студент групи 2АКІТ-20б

Н. Бондар

Перевірив

к.т.н., доц.кафедри КСУ

Севастьянов В. М.

м.Вінниця, ВНТУ 2021

**Лабораторна робота №1**

**Тема: «Структури»**

**Варіант 1.**

**Завдання :**

1. Описати структуру з ім'ям STUDENT, що містить наступні поля:

- прізвище і ініціали;

- номер групи;

- успішність (масив з п’яти елементів).

2. Написати програму, що виконує наступні дії:

- введення з клавіатури даних у масив, що складається з десяти структур типу STUDENT;

- записи повинні бути упорядковані по зростанню номера групи;

- вивід на дисплей прізвищ і номерів груп для всіх студентів, включених у масив, якщо

середній бал студента більше 4,0;

- якщо таких студентів немає, вивести відповідне повідомлення.

**Хід роботи :**

**1. Ретельно вивчаю поставлене передімною завдання.**

**2. Відкриваю середовище та пишу початковий код програми, супутньо коментуючи фрагменти коду.**

**3. Описую структуру з відповідними до завдання полями.**

// Defining struct prototipe

struct STUDENT {

char name[100] = {};

char group\_number[100] = {};

int marks[marks\_number] = {};

float middle\_mark = 0;

};

**4. Розписую цикл введення необхідної іншормаціїї з клавіатури.**

// Circle to input all student's information

for (int num = 0; num < student\_number; num++) {

// Entering strudent's name

std::cout << "Enter student's name : ";

input(students\_list[num].name);

// Entering strudent's group number

std::cout << "Enter student's group number : ";

input(students\_list[num].group\_number);

// Entering strudent's marks

std::cout << "Enter " << marks\_number << " student's marks :\n";

for (int num\_1 = 0; num\_1 < marks\_number; num\_1++) {

std::cout << " Mark number " << num\_1 + 1 << " : ";

std::cin >> students\_list[num].marks[num\_1];

students\_list[num].middle\_mark += students\_list[num].marks[num\_1];

}

std::cin.get();

std::cout << '\n';

}

**5. До вищеописаного шиклу додаю розрахунок середньої арифметичної оцінки для подальшого використання у вибірковому відображенні студентів.**

// Entering strudent's marks

std::cout << "Enter " << marks\_number << " student's marks :\n";

for (int num\_1 = 0; num\_1 < marks\_number; num\_1++) {

std::cout << " Mark number " << num\_1 + 1 << " : ";

std::cin >> students\_list[num].marks[num\_1];

students\_list[num].middle\_mark += students\_list[num].marks[num\_1];

}

// Entering strudent's midle mark

students\_list[num].middle\_mark /= marks\_number;

**6. Використовуючи метод сортування «Бульбашкою» відсортовую елементи структури за збільшенням номеру групи студента.**

// Creating temp struct copy

STUDENT temp\_student;

int num\_3 = 0;

// Bouble sort circle

for (int num\_1 = 0; num\_1 < student\_number - 1; num\_1++) {

for (int num\_2 = 0; num\_2 < student\_number - 1; num\_2++) {

for (num\_3 = 0; students\_list[num\_2].group\_number[num\_3] == students\_list[num\_2 + 1].group\_number[num\_3]/\* && students\_list[num\_2].group\_number[num\_3] != '\0' && students\_list[num\_2 + 1].group\_number[num\_3] != '\0'\*/; num\_3++) {}

if (students\_list[num\_2].group\_number[num\_3] > students\_list[num\_2 + 1].group\_number[num\_3]) {

// Comented text was used for debbuging

// std::cout << '\n' << students\_list[num\_2].group\_number[0] << " > " << students\_list[num\_2 + 1].group\_number[0] << '\n';;

// Swithing students' name

move(students\_list[num\_2].name, temp\_student.name);

move(students\_list[num\_2 + 1].name, students\_list[num\_2].name);

move(temp\_student.name, students\_list[num\_2 + 1].name);

// Swithing students' group number

move(students\_list[num\_2].group\_number, temp\_student.group\_number);

move(students\_list[num\_2 + 1].group\_number, students\_list[num\_2].group\_number);

move(temp\_student.group\_number, students\_list[num\_2 + 1].group\_number);

// Swithing students' marks

for (int num\_3 = 0; num\_3 < marks\_number; num\_3++) {

temp\_student.marks[num\_3] = students\_list[num\_2].marks[num\_3];

students\_list[num\_2].marks[num\_3] = students\_list[num\_2 + 1].marks[num\_3];

students\_list[num\_2 + 1].marks[num\_3] = temp\_student.marks[num\_3];

}

// Swithing students' middle mark

temp\_student.middle\_mark = students\_list[num\_2].middle\_mark;

students\_list[num\_2].middle\_mark = students\_list[num\_2 + 1].middle\_mark;

students\_list[num\_2 + 1].middle\_mark = temp\_student.middle\_mark;

}

}

}

**7. Створюю цикл відображення студентів та додаткової іншормації про них.**

// Showing students list

std::cout << "\nStudents list (sorted by group number from big to small) :\n";

for (int num\_1 = 0; num\_1 < student\_number; num\_1++) {

if (students\_list[num\_1].middle\_mark > needed\_mark) {

std::cout << "\n Name : " << students\_list[num\_1].name

<< "\n Group : " << students\_list[num\_1].group\_number

<< "\n Makrs : ";

for (int num\_2 = 0; num\_2 < marks\_number; num\_2++) {

std::cout << students\_list[num\_1].marks[num\_2] << ' ';

}

std::cout << "\n Middle mark : " << students\_list[num\_1].middle\_mark;

std::cout << '\n';

student\_found = true;

}

}

**8. Додаю параметр виведення інформації про студента, якщо його середнє арифметичне більше чітвірки.**

bool student\_found = false;

// Showing students list

std::cout << "\nStudents list :\n";

for (int num\_1 = 0; num\_1 < student\_number; num\_1++) {

if (students\_list[num\_1].middle\_mark > needed\_mark) {

std::cout << "\n Name : " << students\_list[num\_1].name

<< "\n Group : " << students\_list[num\_1].group\_number

<< "\n Makrs : ";

for (int num\_2 = 0; num\_2 < marks\_number; num\_2++) {

std::cout << students\_list[num\_1].marks[num\_2] << ' ';

}

std::cout << "\n Middle mark : " << students\_list[num\_1].middle\_mark;

std::cout << '\n';

student\_found = true;

}

}

**9. Якщо таких студентів не було виявлено — виводже відповідне повідомлення.**

// If nobody was worst to be shown print such message

if (!student\_found) {

std::cout << "Such student was not found !!!\n";

}

**10. Розписуюю необхідні функціїї введення та перемішення стрічки.**

// Input function body

void input(char \*string) {

for (int num\_1 = 0; num\_1 < 100; num\_1++) {

string[num\_1] = std::cin.get();

if (string[num\_1] == '\n') {

string[num\_1] = '\0';

break;

}

}

}

// Move function body

void move(char \*string\_1, char \*string\_2) {

std::cout << "\nmove " << string\_1 << " to " << string\_2;

for (int num = 0; string\_2[num]; num++) {

string\_2[num] = '\0';

}

for (int num = 0; string\_1[num] != '\0'; num++) {

string\_2[num] = string\_1[num];

}

}

**11. Знаходжу та виправлею помилки, які були допищені при написанні коду програми.**

**12. Оптимізовую роботу коду.**

**Готовий код програми :**

/\*

Варіант 1.

1.Описати структуру з ім'ям STUDENT, що містить наступні поля:

- прізвище і ініціали;

- номер групи;

- успішність (масив з п’яти елементів).

2. Написати програму, що виконує наступні дії:

- введення з клавіатури даних у масив, що складається з десяти структур типу STUDENT;

- записи повинні бути упорядковані по зростанню номера групи;

- вивід на дисплей прізвищ і номерів груп для всіх студентів, включених у масив, якщо

середній бал студента більше 4,0;

- якщо таких студентів немає, вивести відповідне повідомлення.

\*/

#include<iostream>

void input(char \*string); // Function for inputing string

void move(char \*string\_1, char \*string\_2); // Function to move one string into other one

// Three main control points

#define student\_number 3 // should be 10

#define marks\_number 3

#define needed\_mark 2.0

// Defining struct prototipe

struct STUDENT {

char name[100] = {};

char group\_number[100] = {};

int marks[marks\_number] = {};

float middle\_mark = 0;

};

int main() {

// Initialisingarray of structs

STUDENT students\_list[student\_number];

// Circle to input all student's information

for (int num = 0; num < student\_number; num++) {

// Entering strudent's name

std::cout << "Enter student's name : ";

input(students\_list[num].name);

// Entering strudent's group number

std::cout << "Enter student's group number : ";

input(students\_list[num].group\_number);

// Entering strudent's marks

std::cout << "Enter " << marks\_number << " student's marks :\n";

for (int num\_1 = 0; num\_1 < marks\_number; num\_1++) {

std::cout << " Mark number " << num\_1 + 1 << " : ";

std::cin >> students\_list[num].marks[num\_1];

students\_list[num].middle\_mark += students\_list[num].marks[num\_1];

}

// Entering strudent's midle mark

students\_list[num].middle\_mark /= marks\_number;

std::cin.get();

std::cout << '\n';

}

// Creating temp struct copy

STUDENT temp\_student;

int num\_3 = 0;

// Bouble sort circle

for (int num\_1 = 0; num\_1 < student\_number - 1; num\_1++) {

for (int num\_2 = 0; num\_2 < student\_number - 1; num\_2++) {

for (num\_3 = 0; students\_list[num\_2].group\_number[num\_3] == students\_list[num\_2 + 1].group\_number[num\_3]/\* && students\_list[num\_2].group\_number[num\_3] != '\0' && students\_list[num\_2 + 1].group\_number[num\_3] != '\0'\*/; num\_3++) {}

if (students\_list[num\_2].group\_number[num\_3] > students\_list[num\_2 + 1].group\_number[num\_3]) {

// Comented text was used for debbuging

// std::cout << '\n' << students\_list[num\_2].group\_number[0] << " > " << students\_list[num\_2 + 1].group\_number[0] << '\n';;

// Swithing students' name

move(students\_list[num\_2].name, temp\_student.name);

move(students\_list[num\_2 + 1].name, students\_list[num\_2].name);

move(temp\_student.name, students\_list[num\_2 + 1].name);

// Swithing students' group number

move(students\_list[num\_2].group\_number, temp\_student.group\_number);

move(students\_list[num\_2 + 1].group\_number, students\_list[num\_2].group\_number);

move(temp\_student.group\_number, students\_list[num\_2 + 1].group\_number);

// Swithing students' marks

for (int num\_3 = 0; num\_3 < marks\_number; num\_3++) {

temp\_student.marks[num\_3] = students\_list[num\_2].marks[num\_3];

students\_list[num\_2].marks[num\_3] = students\_list[num\_2 + 1].marks[num\_3];

students\_list[num\_2 + 1].marks[num\_3] = temp\_student.marks[num\_3];

}

// Swithing students' middle mark

temp\_student.middle\_mark = students\_list[num\_2].middle\_mark;

students\_list[num\_2].middle\_mark = students\_list[num\_2 + 1].middle\_mark;

students\_list[num\_2 + 1].middle\_mark = temp\_student.middle\_mark;

}

}

}

bool student\_found = false;

// Showing students list

std::cout << "\nStudents list (sorted by group number from big to small) :\n";

for (int num\_1 = 0; num\_1 < student\_number; num\_1++) {

if (students\_list[num\_1].middle\_mark > needed\_mark) {

std::cout << "\n Name : " << students\_list[num\_1].name

<< "\n Group : " << students\_list[num\_1].group\_number

<< "\n Makrs : ";

for (int num\_2 = 0; num\_2 < marks\_number; num\_2++) {

std::cout << students\_list[num\_1].marks[num\_2] << ' ';

}

std::cout << "\n Middle mark : " << students\_list[num\_1].middle\_mark;

std::cout << '\n';

student\_found = true;

}

}

// If nobody was worst to be shown print such message

if (!student\_found) {

std::cout << "Such student was not found !!!\n";

}

return 0;

}

// Input function body

void input(char \*string) {

for (int num\_1 = 0; num\_1 < 100; num\_1++) {

string[num\_1] = std::cin.get();

if (string[num\_1] == '\n') {

string[num\_1] = '\0';

break;

}

}

}

// Move function body

void move(char \*string\_1, char \*string\_2) {

// std::cout << "\nmove " << string\_1 << " to " << string\_2;

for (int num = 0; string\_2[num]; num++) {

string\_2[num] = '\0';

}

for (int num = 0; string\_1[num] != '\0'; num++) {

string\_2[num] = string\_1[num];

}

}

**Висновок :** виконуючи дану лабораторно роботу, я навчився використовувати структири, для обєднань інформації різних типів. Такою використав на практиці сортування методом більбашки.