**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ВСП – Технічний коледж**

**НУ «Львівська політехніка»**

Відділення

Інформаційних технологій

Та комп`ютерної техніки

**ЗВІТ**

Про виконання лабораторної роботи №1

З предмету «Навчальна практика з основ об'єктно-орієнтованого програмування»

Виконав

Студент групи 31 ПЗ

Тисляк А.О.

Перевірив викладач:

Чистяк В.В.

Львів 2021

**Лабораторна робота № 1 «Створення класу C++»**

**Мета роботи:** порівняння об’єктно-орієнтованого та функціонального підходів; початкове знайомство з класами, об’єктами та головними елементами об’єктного підходу.

*Варіант 10*

**Завдання:** створити клас для обробки масиву даних у відповідності з наданим варіантом. Розмістити інтерфейс класу у заголовочному файлі, а визначення функцій та головну функцію програми – у двох окремих файлах. Передбачити можливість роботи з довільним числом записів, а також реалізувати окремими функціями класу: конструктори без параметрів та з параметрами ; додавання; знищення; виведення інформації на екран; пошук потрібної інформації за конкретною ознакою; редагування записів; сортування за різними полями. Використайте захищення даних для ізоляції елементів-даних класу від підпрограм, в яких цей клас використовується. Програма повинна містити меню для перевірки всіх методів класу.

**Завдання мого варіанту:** створити клас «успішність студентів», який містить такі поля: прізвище, ім’я по батькові, номер групи, оцінки з 3 предметів.

**Код програми:**

***Файл StudentProgress.h***

#pragma once

#include <string>

#include <vector>

class StudentProgress {

private:

std::string first\_name;

std::string last\_name;

std::string father\_name;

std::string group;

int math\_mark; //mathematics

int sw\_dev\_mark; //softrware development

int english\_mark; //English language

protected:

void set\_first\_name(const std::string& first\_name);

void set\_last\_name(const std::string& last\_name);

void set\_father\_name(const std::string& father\_name);

void set\_group(const std::string& group);

void set\_math\_mark(const int& math\_mark);

void set\_sw\_dev\_mark(const int& sw\_dev\_mark);

void set\_english\_mark(const int& english\_mark);

public:

StudentProgress(const std::string& first\_name = "",

const std::string& last\_name = "",

const std::string& father\_name = "",

const std::string& group = "",

const int& math\_mark = 0,

const int& sw\_dev\_mark = 0,

const int& english\_mark = 0);

StudentProgress(const StudentProgress& other);

~StudentProgress() = default;

std::string get\_first\_name() const;

std::string get\_last\_name() const;

std::string get\_father\_name() const;

std::string get\_group() const;

int get\_math\_mark() const;

int get\_sw\_dev\_mark() const;

int get\_english\_mark() const;

void print\_student\_info(const int& num = 0) const;

};

***Файл StudentProgress.cpp***

#include <iostream>

#include <string>

#include <vector>

#include <cassert>

#include <Windows.h>

#include <conio.h>

#include <algorithm>

#include "StudentProgress.h"

StudentProgress::StudentProgress(const std::string& first\_name,

const std::string& last\_name,

const std::string& father\_name,

const std::string& group,

const int& math\_mark,

const int& sw\_dev\_mark,

const int& english\_mark) {

for(auto& i : first\_name)

if (isdigit(i) || i < 'A' || i > 'z' || i > 'Z' && i < 'a') {

std::cerr << "\nFATAL ERROR: Wrong first name is given to the constructor!\n";

system("pause");

exit(-1);

}

for (auto& i : last\_name)

if (isdigit(i) || i < 'A' || i > 'z' || i > 'Z' && i < 'a') {

std::cerr << "\nFATAL ERROR: Wrong last name is given to the constructor!\n";

system("pause");

exit(-1);

}

for (auto& i : father\_name)

if (isdigit(i) || i < 'A' || i > 'z' || i > 'Z' && i < 'a') {

std::cerr << "\FATAL ERROR: Wrong father name is given to the constructor!\n";

system("pause");

exit(-1);

}

for (auto& i : group) {

if (!isdigit(i) && !(i >= 'a' && i <= 'z' || i >= 'A' && i <= 'Z' || i == '\_')) {

std::cerr << "\FATAL ERROR: Wrong math mark is given to the constructor!\n";

system("pause");

exit(-1);

}

}

if (math\_mark < 0 || math\_mark > 100) {

std::cerr << "\FATAL ERROR: Wrong math mark is given to the constructor!\n";

system("pause");

exit(-1);

}

if (sw\_dev\_mark < 0 || sw\_dev\_mark > 100) {

std::cerr << "\FATAL ERROR: Wrong software development mark is given to the constructor!\n";

system("pause");

exit(-1);

}

if (english\_mark < 0 || english\_mark > 100) {

std::cerr << "\FATAL ERROR: Wrong English language mark is given to the constructor!\n";

system("pause");

exit(-1);

}

this->first\_name = first\_name;

this->last\_name = last\_name;

this->father\_name = father\_name;

this->group = group;

this->math\_mark = math\_mark;

this->sw\_dev\_mark = sw\_dev\_mark;

this->english\_mark = english\_mark;

}

StudentProgress::StudentProgress(const StudentProgress& other) {

this->first\_name = other.first\_name;

this->last\_name = other.last\_name;

this->father\_name = other.father\_name;

this->group = other.group;

this->math\_mark = other.math\_mark;

this->sw\_dev\_mark = other.sw\_dev\_mark;

this->english\_mark = other.english\_mark;

}

std::string StudentProgress::get\_first\_name() const {

return this->first\_name;

}

std::string StudentProgress::get\_last\_name() const {

return this->last\_name;

}

std::string StudentProgress::get\_father\_name() const {

return this->father\_name;

}

std::string StudentProgress::get\_group() const {

return this->group;

}

int StudentProgress::get\_math\_mark() const {

return this->math\_mark;

}

int StudentProgress::get\_sw\_dev\_mark() const {

return this->sw\_dev\_mark;

}

int StudentProgress::get\_english\_mark() const {

return this->english\_mark;

}

void StudentProgress::set\_first\_name(const std::string& first\_name) {

this->father\_name = first\_name;

}

void StudentProgress::set\_last\_name(const std::string& last\_name) {

this->last\_name = last\_name;

}

void StudentProgress::set\_father\_name(const std::string& father\_name) {

this->father\_name = father\_name;

}

void StudentProgress::set\_group(const std::string& group) {

this->group = group;

}

void StudentProgress::set\_math\_mark(const int& math\_mark) {

this->math\_mark = math\_mark;

}

void StudentProgress::set\_sw\_dev\_mark(const int& sw\_dev\_mark) {

this->sw\_dev\_mark = sw\_dev\_mark;

}

void StudentProgress::set\_english\_mark(const int& english\_mark) {

this->english\_mark = english\_mark;

}

void StudentProgress::print\_student\_info(const int& num) const {

if (num)

std::cout << "--- Student #" << num << " -\n";

std::cout << "First name: " << this->first\_name << "\n";

std::cout << "Last name: " << this->last\_name << "\n";

std::cout << "Father name: " << this->father\_name << "\n";

std::cout << "Group: " << this->group << "\n";

std::cout << "Math mark: " << this->math\_mark << "\n";

std::cout << "Software development mark: " << this->sw\_dev\_mark << "\n";

std::cout << "English language mark: " << this->english\_mark << "\n\n";

}

***Файл Main1.cpp***

#include <iostream>

#include <Windows.h>

#include <conio.h>

#include <vector>

#include <algorithm>

#include "StudentProgress.h"

void add\_student(std::vector<StudentProgress>& arr) {

std::string fname, lname, fthname, grp;

int mmark, sdmark, emark;

bool done;

do {

std::cout << "First name: ";

std::cin >> fname;

std::cout << "Last name: ";

std::cin >> lname;

std::cout << "Father name: ";

std::cin >> fthname;

std::cout << "Group : ";

std::cin >> grp;

std::cout << "Math mark : ";

std::cin >> mmark;

std::cout << "Software development mark : ";

std::cin >> sdmark;

std::cout << "English language mark : ";

std::cin >> emark;

done = true;

for (auto& i : fname)

if (isdigit(i) || i < 'A' || i > 'z' || i > 'Z' && i < 'a') {

std::cout << "\nWrong first name entered. Try again!\n\n";

done = false;

}

for (auto& i : lname)

if (isdigit(i) || i < 'A' || i > 'z' || i > 'Z' && i < 'a') {

std::cout << "\nWrong last name entered. Try again!\n\n";

done = false;

}

for (auto& i : fthname)

if (isdigit(i) || i < 'A' || i > 'z' || i > 'Z' && i < 'a') {

std::cout << "Wrong father name entered. Try again!\n\n";

done = false;

}

for (auto& i : grp)

if (!isdigit(i) && !(i >= 'a' && i <= 'z' || i >= 'A' && i <= 'Z' || i == '\_')) {

std::cout << "\nWrong group entered. Try again!\n\n";

done = false;

}

if (mmark < 0 || mmark > 100) {

std::cout << "\nWrong math mark input. Try again!\n\n";

done = false;

}

if (sdmark < 0 || sdmark > 100) {

std::cout << "\nWrong software development mark input. Try again!\n\n";

done = false;

}

if (emark < 0 || emark > 100) {

std::cout << "\nWrong English language mark input. Try again!\n\n";

done = false;

}

} while (!done);

arr.push\_back({ fname, lname, fthname, grp, mmark, sdmark, emark });

}

void del\_student(std::vector<StudentProgress>& arr) {

int num;

std::cout << "Enter the num: ";

std::cin >> num;

if (num < 1 || num > arr.size()) {

std::cout << "\nWrong input!\n";

return;

}

arr.erase(arr.begin() + num - 1);

}

void enter\_students\_info(std::vector<StudentProgress>& arr) {

int count;

do {

std::cout << "Enter a count of students: ";

std::cin >> count;

if (count < 0)

std::cout << "Wrong input! Try again.\n";

} while (count < 0);

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

std::cout << "\n--- Student #" << i + 1 << " -\n";

add\_student(arr);

}

}

void find\_student(const std::vector<StudentProgress>& arr) {

char key;

bool valid\_pressed = false;

std::cout << "Press the appropriate key to choose a parameter to find:\n";

std::cout << "\tFirst name : 1\n\tLast name: 2\n\tFather name : 3\n\tGroup : 4\n";

std::cout << "\tMath mark : 5\n\tSoftware development mark : 6\n\tEnglish language mark : 7\n";

std::string str\_tmp;

int int\_tmp;

int found = 0;

while (!valid\_pressed) {

switch (key = \_getch()) {

case '1':

std::cout << "\nYou pressed 1\n\n";

std::cout << "Enter a first name to find: ";

std::cin >> str\_tmp;

for (auto& i : arr) {

if (i.get\_first\_name() == str\_tmp) {

i.print\_student\_info(found + 1);

found++;

}

}

valid\_pressed = true;

break;

case '2':

std::cout << "\nYou pressed 2\n\n";

std::cout << "Enter a last name to find: ";

std::cin >> str\_tmp;

for (auto& i : arr) {

if (i.get\_last\_name() == str\_tmp) {

i.print\_student\_info(found + 1);

found++;

}

}

valid\_pressed = true;

break;

case '3':

std::cout << "\nYou pressed 3\n\n";

std::cout << "Enter a father name to find: ";

std::cin >> str\_tmp;

for (auto& i : arr) {

if (i.get\_father\_name() == str\_tmp) {

i.print\_student\_info(found + 1);

found++;

}

}

valid\_pressed = true;

break;

case '4':

std::cout << "\nYou pressed 4\n\n";

std::cout << "Enter a group to find: ";

std::cin >> str\_tmp;

for (auto& i : arr) {

if (i.get\_group() == str\_tmp) {

i.print\_student\_info(found + 1);

found++;

}

}

valid\_pressed = true;

break;

case '5':

std::cout << "\nYou pressed 5\n\n";

std::cout << "Enter a math mark to find: ";

std::cin >> int\_tmp;

for (auto& i : arr) {

if (i.get\_math\_mark() == int\_tmp) {

i.print\_student\_info(found + 1);

found++;

}

}

valid\_pressed = true;

break;

case '6':

std::cout << "\nYou pressed 6\n\n";

std::cout << "Enter a software development mark to find: ";

std::cin >> int\_tmp;

for (auto& i : arr) {

if (i.get\_sw\_dev\_mark() == int\_tmp) {

i.print\_student\_info(found + 1);

found++;

}

}

valid\_pressed = true;

break;

case '7':

std::cout << "\nYou pressed 7\n\n";

std::cout << "Enter an English language mark to find: ";

std::cin >> int\_tmp;

for (auto& i : arr) {

if (i.get\_english\_mark() == int\_tmp) {

i.print\_student\_info(found + 1);

found++;

}

}

valid\_pressed = true;

break;

}

}

if (!found)

std::cout << "\nNothing is found!\n";

}

void sort\_students(const std::vector<StudentProgress>& arr) {

std::vector<StudentProgress> copy = arr;

char key;

bool sorted = false;

std::cout << "Press the appropriate key to choose a sorting parameter:\n";

std::cout << "\tFirst name : 1\n\tLast name: 2\n\tFather name : 3\n\tGroup : 4\n";

std::cout << "\tMath mark : 5\n\tSoftware development mark : 6\n\tEnglish language mark : 7\n";

while (!sorted) {

switch (key = \_getch()) {

case '1':

std::cout << "\nYou pressed 1\n\n";

std::sort(copy.begin(), copy.end(),

[](const StudentProgress& a, const StudentProgress& b)

{ return a.get\_first\_name() < b.get\_first\_name(); });

sorted = true;

break;

case '2':

std::cout << "\nYou pressed 2\n\n";

std::sort(copy.begin(), copy.end(),

[](const StudentProgress& a, const StudentProgress& b)

{ return a.get\_last\_name() < b.get\_last\_name(); });

sorted = true;

break;

case '3':

std::cout << "\nYou pressed 3\n\n";

std::sort(copy.begin(), copy.end(),

[](const StudentProgress& a, const StudentProgress& b)

{ return a.get\_father\_name() < b.get\_father\_name(); });

sorted = true;

break;

case '4':

std::cout << "\nYou pressed 4\n\n";

std::sort(copy.begin(), copy.end(),

[](const StudentProgress& a, const StudentProgress& b)

{ return a.get\_group() < b.get\_group(); });

sorted = true;

break;

case '5':

std::cout << "\nYou pressed 5\n\n";

std::sort(copy.begin(), copy.end(),

[](const StudentProgress& a, const StudentProgress& b)

{ return a.get\_math\_mark() < b.get\_math\_mark(); });

sorted = true;

break;

case '6':

std::cout << "\nYou pressed 6\n\n";

std::sort(copy.begin(), copy.end(),

[](const StudentProgress& a, const StudentProgress& b)

{ return a.get\_sw\_dev\_mark() < b.get\_sw\_dev\_mark(); });

sorted = true;

break;

case '7':

std::cout << "\nYou pressed 6\n\n";

std::sort(copy.begin(), copy.end(),

[](const StudentProgress& a, const StudentProgress& b)

{ return a.get\_english\_mark() < b.get\_english\_mark(); });

sorted = true;

break;

}

}

int copy\_size = copy.size();

for (int i = 0; i < copy\_size; i++)

copy[i].print\_student\_info(i + 1);

}

void change\_student\_info(std::vector<StudentProgress>& arr) {

int num;

std::cout << "Enter the num: ";

std::cin >> num;

if (num < 1 || num > arr.size() - 1) {

std::cout << "Wrong input!\n";

system("pause");

return;

}

std::vector<StudentProgress> tmp;

std::cout << "Enter the new information:\n\n";

add\_student(tmp);

arr[num - 1] = tmp[0];

}

void print\_all\_students(const std::vector<StudentProgress>& arr) {

std::cout << "All students list:\n\n";

int n = 1;

for (auto& i : arr) {

std::cout << "--- Student #" << n++ << "\n";

std::cout << "First name: " << i.get\_first\_name() << "\n";

std::cout << "Last name: " << i.get\_last\_name() << "\n";

std::cout << "Father name: " << i.get\_father\_name() << "\n";

std::cout << "Group : " << i.get\_group() << "\n";

std::cout << "Math mark : " << i.get\_math\_mark() << "\n";

std::cout << "Software development mark : " << i.get\_sw\_dev\_mark() << "\n";

std::cout << "English language mark : " << i.get\_english\_mark() << "\n\n";

}

}

void user\_interface() {

using namespace std;

vector<StudentProgress> arr;

bool notExit = true;

char keyHit;

while (notExit) {

cout << "Hello dear user!\n\n";

cout << "Available commands list (press the appropriate number):\n";

cout << "1) Enter students list\n";

cout << "2) Add one student\n";

cout << "3) Delete a student\n";

cout << "4) Change student data\n";

cout << "5) Print all students data\n";

cout << "6) Find students using a parameter\n";

cout << "7) Print sorted students list\n";

cout << "9) Exit the program\n\n";

keyHit = \_getch();

if (keyHit == '1') {

system("cls");

enter\_students\_info(arr);

std::cout << "\n";

system("pause");

}

else if (keyHit == '2') {

system("cls");

add\_student(arr);

std::cout << "\n";

system("pause");

}

else if (keyHit == '3') {

system("cls");

del\_student(arr);

std::cout << "\n";

system("pause");

}

else if (keyHit == '4') {

system("cls");

change\_student\_info(arr);

std::cout << "\n";

system("pause");

}

else if (keyHit == '5') {

system("cls");

print\_all\_students(arr);

system("pause");

}

else if (keyHit == '6') {

system("cls");

find\_student(arr);

system("pause");

}

else if (keyHit == '7') {

system("cls");

sort\_students(arr);

system("pause");

}

else if (keyHit == '9') {

system("cls");

cout << "Thank you for using my program!\n\n";

system("pause");

notExit = false;

}

system("cls");

}

}

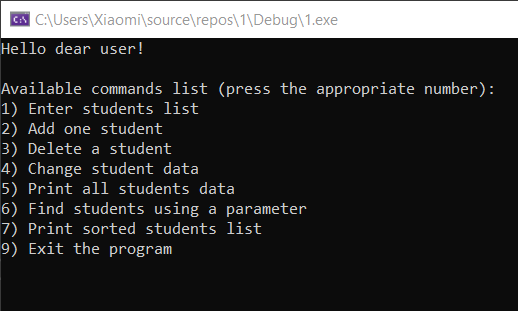
int main() {

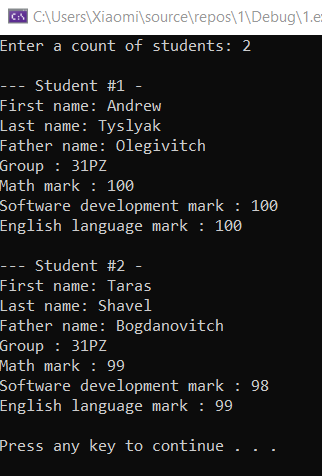
user\_interface();

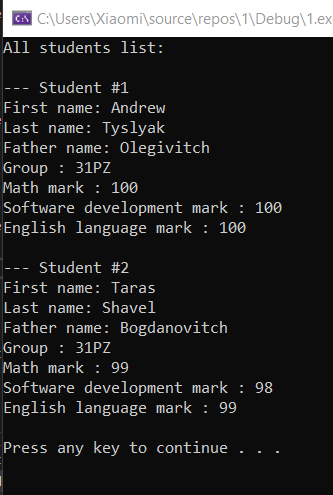
return 0;

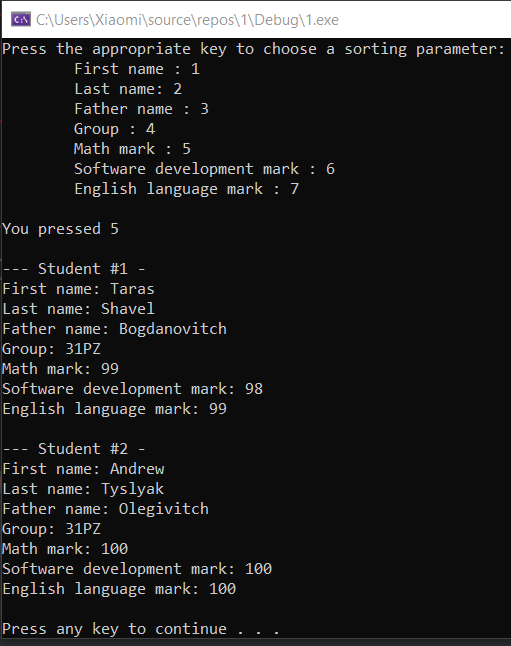
}

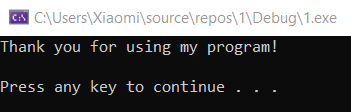
***Скрін-шоти роботи програми:***

******

******

******

******

******

**Висновок**

Отже, під час виконання лабораторної роботи №1, я чудово провів час, порівнюючи об’єктно-орієнтований та функціональний підходи; початково знайомлячись з класами, об’єктами та головними елементами об’єктного підходу.