Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

Механіко-технологічний факультет

ЗВІТ

ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 11

з навчальної дисципліни

“Базові методології та технології програмування”

РЕАЛІЗАЦІЯ ПРОГРАМНИХ ЗАСОБІВ ОБРОБЛЕННЯ ДИНАМІЧНИХ СТРУКТУР ДАНИХ ТА БІНАРНИХ ФАЙЛІВ

ЗАВДАННЯ ВИДАВ

доцент кафедри кібербезпеки та програмного забезпечення

Доренський О. П.

<https://github.com/odorenskyi/>

ВИКОНАЛИ

студенти академічної групи ТК-23-1

Шавлєнков П.О

Макодзеба П.П

студент академічної групи КН-23

Ситенкова В.В

ПЕРЕВІРИВ

викладач кафедри кібербезпеки

та програмного забезпечення

Ганна ДРЄЄВА

Кропивницький – 2024

**Тема:** Реалізація програмних засобів оброблення динамічних структур даних та бінарних файлів

**Мета роботи:** Набуття ґрунтовних вмінь і практичних навичок командної (колективної) реалізації програмного забезпечення, розроблення функцій оброблення динамічних структур даних, використання стандартних засобів С++ для керування динамічною пам’яттю та бінарними файловими потоками.

**Завдання до лабораторної роботи:**

1. У складі команди ІТ-проекта розробити програмні модулі оброблення динамічної структури даних.

2. Реалізувати програмний засіб на основі розроблених командою ІТ-проекта модулів.

**Варіант 8**



**Склад команди IT-проекта:** Макодзеба Павло, Ситенкова Вероніка, Шавлєнков Павло

**Розроблений план виконання IT-проекта**

**«Деканат: облік студентів»**

**1. Аналіз задач ІТ-проекта та вимог до програмного забезпечення**

**● Ціль проекту:**

* розробка бази даних «Деканат: облік студентів» та додатку для взаємодії з нею

**● Функціональні вимоги:**

1. ***Виведення*** всього реєстру на екран або у заданий текстовий файл
2. ***Додавання*** нового запису до реєстру студентів з такою інформацією: прізвище, ім'я, по батькові студента, дата народження, місце народження, громадянство, сімейний стан, місце проживання, факультет, спеціальність, спеціалізація.
3. ***Пошук***запису в реєстрі за даним прізвищем;
4. ***Вилучення*** заданого запису з реєстру.
5. ***Завершення***роботи програми з автоматичним записом реєстру у файл бази.

**● Нефункціональні вимоги:**

1. Програма повинна бути ефективною та швидкою у роботі;
2. Надійність та стійкість програми до помилок та відмов.

**● Дані:**

1. Прізвище, ім'я, по батькові студента;
2. Дата народження;
3. Місце народження;
4. Громадянство;
5. Сімейний стан;
6. Місце проживання;
7. Факультет;
8. Спеціальність;
9. Спеціалізація;

***Вимоги до програмного забезпечення***

**●Інтерфейс користувача:**

- інтуїтивно зрозумілий та зручний для введення та виведення інформації.

**● Продуктивність:**

- швидке виконання запитів, оптимізація для великих обсягів даних.

**● Сумісність:**

- підтримка різних операційних систем.

**● Масштабованість:**

- легкість оновлення та розширення функціоналу.

**● Технічні обмеження:**

- врахування можливостей використовуваного обладнання та програмного середовища.

**● Документація:**

- повна документація коду

**2. Специфікації ПЗ, концептуальні проектні рішення, архітектура програмного засобу, загальні алгоритми функціонування та інтерфейси модулів**

***Специфікація ПЗ***

**Обговорення щодо:**

1) вимог до програмного забезпечення;

2) вибрання типу динамічної структури даних (список, стек, черга, дерево) для реалізації бази даних ПЗ;

3) створення заголовочного файлу;

4) розподілити підзадачі реалізації операції над динамічною структурою даних;

5) складання плану робіт проекту з урахуванням стандарту ISO/IEC 12207

***Концептуальні проектні рішення***

**1) Модульна архітектура**

• розподіл системи на незалежні модулі для спрощення розробки та тестування:

• модуль виведення бази - відповідає за виведення всієї бази на екран або у заданий текстовий файл.

• модуль додавання нового запису - відповідає за додавання нового запису про студента до бази.

• модуль пошуку за державним номером - відповідає за пошук запису в базі даних за прізвищем. Якщо запис не буде знайдено, модуль виведе відповідне повідомлення.

• модуль вилучення запису - відповідає за вилучення заданого запису з бази даних.

• модуль автоматичного запису бази даних у файл - відповідає за автоматичний запис у файл під час завершення роботи програми.

**2) Використання списку для реалізації обліку студентів**

• оптимізація швидкості пошуку, вставки та видалення даних

• поліпшення впорядкувння та доступної інформації

***Архітектура програмного засобу***

**1)Модуль управління даними**

**●** Основною структурою даних, є структура **Student**

**●** Операції:

- виведення реєстру: функція для виведення всієї бази даних на екран або у заданий текстовий файл. Ця операція дозволяє користувачу переглядати всі дані, що зберігаються у реєстрі.

- додавання нового запису: функція для додавання нового запису про студента до бази даних обліку. Ця операція дозволяє користувачу ввести дані студента та додати його до обліку студентів.

- пошук за прізвищем: функція для пошуку запису в обліку студентів за заданим прізвищем. Ця операція дозволяє знайти інформацію про студента за його прізвищем.

- вилучення  запису: функція для вилучення заданого запису з бази даних. Ця операція дозволяє користувачу вилучити інформацію про студента з бази даних.

- завершення: функція для завершення роботи програми з автоматичним записом бази у файл. Під час завершення роботи програми всі дані про студентів автоматично записуються у файл, щоб забезпечити збереження інформації.

**2) *Інтерфейс користувача***

• Консольний інтерфейс: простий для інтерфейс управліня обліком студентів в БД;

• Взаємодія з користувачем: чіткі інструкції та запити для забезпечення правильного введення даних;

***Загальні алгоритми функціонування***

1.**Виведення всієї бази на екран або у заданий текстовий файл:**

- проходження через всі записи в базі;

- виведення кожного запису на екран або збереження у файл.

**2. Додавання нового запису до бази:**

- перевірка наявності даних про студента;

- введення нових даних;

- збереження нового запису в базі.

**3. Пошук запису в базі за даним прізвищем:**

- введення прізвища для пошуку;

- пошук в базі за прізвищем;

- виведення результату пошуку або відповідного повідомлення, якщо запис не знайдено.

**4. Вилучення заданого запису з бази:**

- введення даних для ідентифікації запису, який потрібно вилучити;

- пошук запису в базі;

- вилучення знайденого запису.

**5. Завершення роботи програми з автоматичним записом бази у файл:**

- збереження всіх змін у базі у файл.

-завершення виконання програми.

**6. Автоматичне завантаження бази з файлу під час запуску програми:**

- перевірка наявності файлу з базою;

- якщо файл існує, завантаження даних з файлу у пам'ять програми;

- якщо файл відсутній, створення порожньої бази.

**3.  Обгрунтування вид динамічної структури даних для реалізації бази даних ПЗ, складових її елементів (поля структури і їх типи).**

Для реалізації бази даних програмного забезпечення «Деканат: облік студентів» у мові програмування C++ було обрано двонапрямлений (двозв’язний)  список.

**Обґрунтування цього вибору наступне:**

**1) Швидкий доступ:**

Двонапрямлені списки дозволяють швидше переміщатися як вперед, так і назад по списку. Це важливо для операцій, таких як пошук та видалення записів за державним номером, де може знадобитися звернення до попередніх записів.

**2) Ефективність додавання та вилучення:**

Додавання та видалення елементів у двонапрямленому списку може бути ефективними, оскільки не потрібно переміщатися через весь список для зміни посилань.

**3) Простота реалізації:**

Двонапрямлені списки є достатньо простими для реалізації та керуються досить зрозумілими правилами, що полегшує їх розуміння та використання.

**4) Зручність використання:**

Для операцій, таких як виведення всього реєстру на екран або у заданий текстовий файл, двонапрямлений список також може бути зручним, оскільки його можна ітерувати у будь-якому напрямку.

**5) Навчальний аспект:**

Часто використовується у курсах зі структур даних. Відповідно, він ідеально підходить для проектів, спрямованих на вивчення студентами фундаментальних принципів роботи структур даних.

**Складові елементи структури**

1. Прізвище

* Поле структури: last\_name
* Тип даних: string

1. Ім’я

* Поле структури: first\_name
* Тип даних: string

1. По-батькові

* Поле структури: middle\_name
* Тип даних: string

4. Дата нароження

1. Місце народження

* Поле структури: place\_of\_birth
* Тип даних: string

1. Громадянство

* Поле структури: citizenship
* Тип даних: string

1. Сімейний стан

* Поле структури: marital\_status
* Тип даних: string

1. Місце проживання

* Поле структури: place\_of\_residence;
* Тип даних: string

1. Факультет

* Поле структури: faculty
* Тип даних: string

1. Спеціальність

* Поле структури: specialty
* Тип даних: string

1. Спеціалізація

* Поле структури: specialization
* Тип даних: string

**4. Створення заголовочного файлу**

**Задача:** створити заголовочний файл struct\_type\_project\_N.h (N ― номер варіанта завдання) з описом елементів динамічної структури даних мовою С++

**5. Розподіл завдань між учасниками команди підзадачі з реалізації операцій над динамічною структурою даних відповідно до розробленої архітектури програмного засобу.**

**Шавлєнков Павло**

1. ***Виведення*** всього реєстру на екран або у заданий текстовий файл
2. ***Додавання*** нового запису до обліку студентів з такою інформацією: прізвище, ім'я, по батькові студента, дата народження, місце народження, громадянство, сімейний стан, місце проживання, факультет, спеціальність, спеціалізація

**Ситенкова Вероніка**

1. ***Пошук***запису в базі за даним прізвищем;
2. ***Вилучення*** заданого запису з бази.

**Макодзеба Павло**

1. ***Завершення***роботи програми з автоматичним записом реєстру у файл бази.

**Шавлєнков Павло**

1. ***Виведення*** всієї бази на екран або у заданий текстовий файл
2. ***Додавання*** нового запису до обліку студентів з такою інформацією: прізвище, ім'я, по батькові студента, дата народження, місце народження, громадянство, сімейний стан, місце проживання, факультет, спеціальність, спеціалізація

**Основні функції:**

* getStudents(int status, string path = "") виведення всієї бази на екран або у заданий текстовий файл.
* addStudent(Student student) додавання нового запису до бази студентів

**Лістинг ModulesShavlienkov**

#include <iostream>

#include <fstream>

#include <string>

#include <cctype>

#include <sstream>

#include <vector>

#include "struct\_type\_project\_8.h"

using namespace std;

void getStudents(int status, string path = "") {

try {

if (status == 1) {

ifstream file("temp.txt");

string content;

if (file.is\_open()) {

string line;

while (getline(file, line)) {

content += line + "\n";

}

} else {

throw runtime\_error("Файл бази даних не знайдено!");

}

istringstream iss(content);

string line;

vector<string> labels = {

"Прізвище", "Ім'я", "По-батькові", "Дата народження",

"Місце народження", "Громадянство", "Сімейний стан",

"Місце проживання", "Факультет", "Спеціальність", "Спеціалізація"

};

cout << "-----------------" << endl;

while (getline(iss, line, '\n')) {

istringstream lineStream(line);

string word;

int labelIndex = 0;

while (lineStream >> word) {

cout << labels[labelIndex] << ": " << word << endl;

++labelIndex;

if (labelIndex >= labels.size()) {

labelIndex = 0;

}

}

cout << "-----------------" << endl;

}

} else if (status == 2 && !path.empty()) {

ifstream file("temp.txt");

string content;

if (file.is\_open()) {

string line;

while (getline(file, line)) {

content += line + "\n";

}

} else {

throw runtime\_error("Файл бази даних не знайдено!");

}

ofstream output(path);

istringstream iss(content);

string line;

vector<string> labels = {

"Прізвище", "Ім'я", "По-батькові", "Дата народження",

"Місце народження", "Громадянство", "Сімейний стан",

"Місце проживання", "Факультет", "Спеціальність", "Спеціалізація"

};

output << "-----------------" << endl;

while (getline(iss, line, '\n')) {

istringstream lineStream(line);

string word;

int labelIndex = 0;

while (lineStream >> word) {

output << labels[labelIndex] << ": " << word << endl;

++labelIndex;

if (labelIndex >= labels.size()) {

labelIndex = 0;

}

}

output << "-----------------" << endl;

}

}

} catch (const exception& e) {

cout << "Помилка: " << e.what() << endl;

}

}

void addStudent(Student student) {

ofstream output("temp.txt", ios\_base::app);

output << student.last\_name << " " << student.first\_name << " " << student.middle\_name << " "

<< student.place\_of\_birth << " " << student.date\_of\_birth << " " << student.citizenship << " "

<< student.marital\_status << " " << student.place\_of\_residence << " " << student.faculty << " "

<< student.specialty << " " << student.specialization << "\n";

output.close();

}

**Ситенкова Вероніка**

1. ***Пошук*** запису за введеним диспетчером прізвищем студента.
2. ***Вилучення*** заданого оператором запису з бази.

**Основні функції:**

* getStudentData(string last\_name = "") пошук запису за введеним диспетчером прізвищем студента.
* deleteStudentData(string last\_name = "") вилучення заданого оператором запису з бази.

**Лістинг ModulesSytenkova**

#include <iostream>

#include <fstream>

#include <string>

#include <cctype>

#include <sstream>

#include <vector>

#include "struct\_type\_project\_8.h"

using namespace std;

void getStudentData(string last\_name = "") {

try {

ifstream file("temp.txt");

if (!file.is\_open()) {

throw runtime\_error("Файл бази даних не знайдено!");

}

string line;

bool found = false;

cout << "-----------------" << endl;

while (getline(file, line)) {

vector<string> elements;

stringstream ss(line);

string item;

while (ss >> item) {

elements.push\_back(item);

}

if (!elements.empty() && elements[0] == last\_name) {

cout << "Прізвище: " << elements[0] << endl;

cout << "Ім'я: " << elements[1] << endl;

cout << "По-батькові: " << elements[2] << endl;

cout << "Місце народження: " << elements[3] << endl;

cout << "Дата народження: " << elements[4] << endl;

cout << "Громадянство: " << elements[5] << endl;

cout << "Сімейний стан: " << elements[6] << endl;

cout << "Місце проживання: " << elements[7] << endl;

cout << "Факультет: " << elements[8] << endl;

cout << "Спеціальність: " << elements[9] << endl;

cout << "Спеціалізація: " << elements[10] << endl;

found = true;

}

cout << "-----------------" << endl;

}

file.close();

if (!found) {

throw runtime\_error("Студент відсутній у базі даних!");

}

} catch (const runtime\_error& e) {

cerr << "Помилка: " << e.what() << endl;

}

}

void deleteStudentData(string last\_name = "") {

try {

ifstream file("temp.txt");

if (!file.is\_open()) {

throw runtime\_error("Файл бази даних не знайдено!");

}

string line;

ofstream temp\_file("temp\_tmp.txt");

bool found = false;

while (getline(file, line)) {

vector<string> elements;

stringstream ss(line);

string item;

while (ss >> item) {

elements.push\_back(item);

}

if (!elements.empty() && elements[0] == last\_name) {

found = true;

} else {

temp\_file << line << endl;

}

}

file.close();

if (!found) {

throw runtime\_error("Студент відсутній у базі даних!");

} else {

cout << "Данні студента видалено!" << endl;

}

temp\_file.close();

remove("temp.txt");

rename("temp\_tmp.txt", "temp.txt");

} catch (const runtime\_error& e) {

cerr << "Помилка: " << e.what() << endl;

}

}

**Макодзеба Павло**

1. ***Завершення***роботи програми з автоматичним записом бази у файл бази.

**Основні функції:**

* exitProgram() Завершенняроботи програми з автоматичним записом бази у файл.

**Лістинг ModulesMakodzeba**

#include <iostream>

#include <fstream>

#include <cstdlib>

using namespace std;

void exitProgram() {

ifstream temp("temp.txt");

ofstream database("database.txt");

database << temp.rdbuf();

temp.close();

database.close();

remove("temp.txt");

}

Висновок: У ході даної лабораторної роботи я здобув ґрунтовні вміння і практичні навички командної (колективної) реалізації програмного забезпечення. Я навчився розробляти функції для оброблення динамічних структур даних, а також використовувати стандартні засоби C++ для керування динамічною пам'яттю та бінарними файловими потоками.