Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

Механіко-технологічний факультет

Кафедра кібербезпеки та програмного забезпечення

# ЗВІТ

ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 11

з навчальної дисципліни “Базові методології та технології програмування”

РЕАЛІЗАЦІЯ ПРОГРАМНИХ ЗАСОБІВ ОБРОБЛЕННЯ ДИНАМІЧНИХ СТРУКТУР ДАНИХ ТА БІНАРНИХ ФАЙЛІВ

ВИКОНАЛИ

студенти академічної групи

КІ-21-1

Стрюк В. Є.

Барамба А.А.

Ожеховський В. В.

ПЕРЕВІРИВ

викладач кафедри кібербезпеки

та програмного забезпечення

\_\_\_\_\_\_\_\_\_  П.С. Усік

Кропивницький – 2022

# ЛАБОРАТОРНА РОБОТА №11

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

**Мета роботи:** полягає у набутті 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.

Звіт виконувався спільно, за участі всіх членів команди. Копія даного документу знаходиться в теці /lab11/Report на особистих репозиторіях Стрюка Владислава, Барамби Андрія та Ожеховського Владислава.

## Варіант 1

**Завдання:**

1. У складі команди ІТ-проекта розробити програмні модулі оброблення динамічної структури даних.
2. Реалізувати програмний засіб на основі розроблених командою ІТ-проекта модулів

# ХІД РОБОТИ

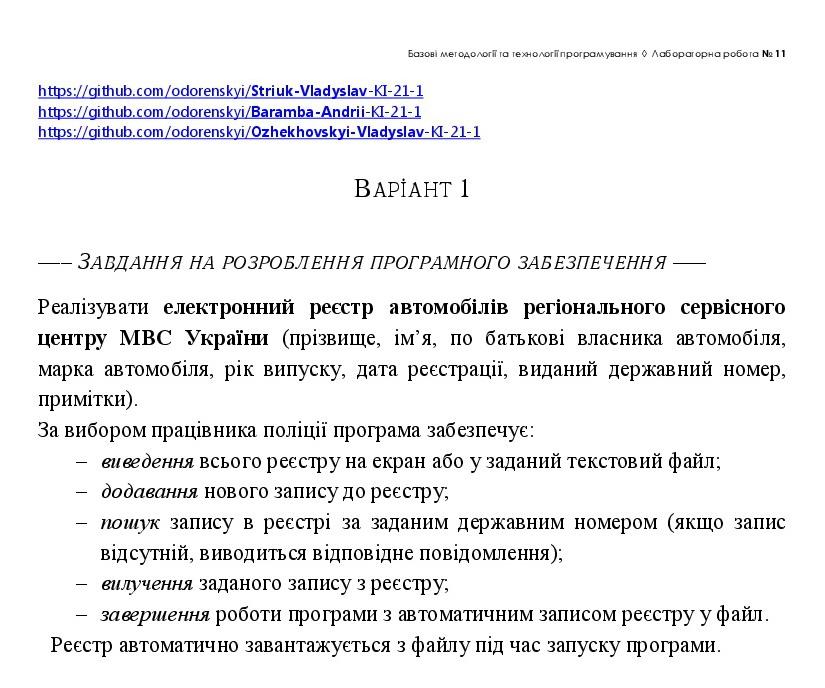


Рисунок 1 - Умова завдання до лабораторної роботи

### **Вимоги до програмного забезпечення**

Електронний реєстр автомобілів має бути реалізований динамічною структурою, кожен елемент якої - окремий запис типу структури.

Всі дані зберігаються в окремому бінарному файлі, маніпуляції з яким відбуваються лише на початку (завантаження даних з файлу в динамічну пам’ять) та наприкінці (завантаження даних із динамічної структури в бінарний файл) функціонування застосунку.

Інші процедури взаємодії з реєстром оперують лише даними, збереженими в пам’яті.

Для зручності користувача, реєстр виводиться у вигляді таблиці в консоль або текстовий файл. Некоректне введення полів структури потребує повторного введення. Інформація щодо успішності зчитування бінарного файлу виводиться повідомленням в консоль.

### **Проектні рішення**

Для зручності форматування було прийняте рішення записувати місяці та дні двома цифрами (якщо день/місяць < 10 - записується 02, 06 тощо), а рік записувати чотирма цифрами.

У функції додавання до реєстру нової людини було прийняте рішення переводити числа з рядка в цілочисельний тип для того, щоб забезпечити правильність вводу користувачем інформації. Також був використаний метод c\_str(), який переводить рядок в масив символьного типу, ще була використана функція strcpy(), яка копіює масив символьного типу в змінну яку ми вказуємо.

Задля оптимізації коду у модулі ModulesBaramba було прийняте рішення використати макрос-функцію для вибору потрібного потоку.

З метою коректності збереження інформації у бінарному файлі були використані обмежені кількості символів для запису. Кількість символів була створена за допомогою макросів.

### **Обраний вид динамічної структури**

Попередньо узгодивши з лектором, в команді було прийнято рішення використати в якості динамічної структури однонаправлений лінійний список. Додавання нового запису відбувається з кінця і сортування за певними ознаками не потребується, тому необхідність у використанні двонаправленого списку була відсутня.

В порівняння зі стеком і чергою, в списці можна видаляти довільні елементи. Тож, однонаправлений список цілком і повністю задовольняє потреби для реалізації функції на основі виданого завдання та обраної динамічної структури.

### **План робіт з виконання ІТ-проекта**

Проаналізувавши завдання лабораторної роботи, та врахувавши вимоги щодо оформлення статичних бібліотек кожним членом команди самостійно, до 5-ти визначених завданням функцій була також врахована 6-та - автоматичне завантаження реєстру з файлу під час роботи застосунку.

Реалізація функцій у особистих модулях була розподілена між групою наступним чином:

1. Стрюк Владислав:
   1. Реалізація належного зчитування даних з бінарного файлу в динамічну структуру однонаправленого списку, з попередньою перевіркою на наявність інформації в файлі та її ініціалізацією у випадку відсутності.
   2. Написання процедури, що забезпечує пошук в реєстрі за введеною частиною державного номеру. Результат пошуку - список імен власників автомобілів, чиї номера містять в собі частину пошукового запиту.
2. Барамба Андрій:
   1. Задля доповнення до функції завантаження, було видано завдання з записом динамічної структури до бінарного файлу при завершенні роботи застосунку. Прийнято до уваги, зміну режиму відкриття для запобігання можливих проблем.
   2. Для наочної демонстрації елементів структури, видано завдання виведення всього реєстру у файл або консоль. З урахування на зручність користування, виведена інформація обмежується умовною таблицею.
3. Ожеховський Владислав:
   1. Додавання запису до реєстру, оперуючи виключно динамічною пам’яттю в структурі однонаправленого списку. Взято до уваги можливість введення некоректних даних та врахована максимальна допустима довжина значень в якості рядків.
   2. Реалізується видалення елементу динамічної структури. Згідно з особливостей розташування елементів у пам’яті, для першого елемента та для інших елементів використовується різна умова.

### **Артефакти процесу реалізації модуля “ModulesStriuk”**

Виконана поставлена задача з завантаженням даних із бінарного файлу безпосередньо в пам’ять динамічної структури однонаправленого списку та пошук запису в реєстрі за частково або повністю вказаним державним номером автомобіля. Модуль містить 3 процедури: readFromFile(), searchInDatabase(), initializeDefaultRoot() (функція initializeDefaultRoot є додатковою).

readFromFile(): обробляє покажчик в якості аргументу, котрому попередньо динамічно виділена пам’ять. Функція виконується рекурсивно та припиняє свою роботу при досягненні кінця файлу методом peek(). На кожній ітерації, покажчик що вказує на наступний елемент структури ініціалізується як пустий і інформація зчитується з файлу для одного елементу. У випадку, якщо кінець файлу не досягнутий, покажчику на наступний елемент виділяється динамічна пам’ять і функція викликається повторно.

initializeDefaultRoot(): додаткова процедура, що виконує свою роботу, якщо інформація в файлі відсутня. Відбувається динамічне виділення пам’яті для першого елементу структури, поля заповнюються значеннями за замовчуванням, покажчик ініціалізується nullptr. Елемент реєстру записується в файл.

searchInDatabase(): вказівник на перший елемент структури буферизується та надалі обробляється. Користувач вводить пошуковий запит у форматі рядка, що обрізається до 8 символів (допустима довжина державного номеру) методом substr(). Надалі, в межах циклу, відбувається порівняння введеного рядка з полем елемента запису в реєстрі: якщо пошук успішний - відбувається виведення поля повного імені власника та державного номеру його автомобіля. Цикл завершується, коли елементи в динамічній структурі відсутні.

### **Артефакти процесу реалізації модуля “ModulesBaramba”**

Даний модуль реалізує виведення реєстру на екран або у заданий текстовий файл та автоматичний запис реєстру у файл при завершенні роботи застосунку. Модуль реалізований за допомогою двох функцій: writeToRegistData та dataOutput. Ці функції відповідно реалізовують запис даних до файлу при завершенні роботи застосунку та вивід даних у консоль або у файл.

writeToRegistData реалізована наступним чином. На початку роботи функції реалізована перевірка на те, чи є перший елемент порожнім. Якщо елемент порожній - виводиться повідомлення про неможливість записати порожню інформацію і робота застосунку завершується, інакше - робота функції продовжується. Далі відкривається бінарний файл у режимі дозапису із попереднім очищенням інформації. За допомогою методу seekg(0) курсор переміщується у початок файлу і далі за допомогою циклу (буферному покажчику привласнюється покажчик на наступний елемент, та запис відбувається доти, доки елемент не буде порожнім) у файл поелементно записується увесь регістр.

dataOutput реалізована наступним чином: на початку роботи функції реалізована така ж сама перевірка, як і у попередній функції, але виводиться повідомлення про неможливість показати порожній реєстр. Вибір запису у файл або у консоль реалізовано за допомогою макроса-функції, що повертає потік для виведення в залежності від вибору користувача. Далі реалізовано цикл do while (цикл відбувається, поки покажчик на наступний елемент існує), у якому реалізовано форматований вивід тексту у консоль/файл із допомогою маніпулятора setw(), який заповнює вказаною в дужках кількістю пробілів. Для запису додатків було використано окремий цикл. Таке рішення викликано великою кількістю символів, що користувач може записати, та обмеженою кількістю символів у стовпчику. Перша умова перевіряє наявність символів у рядку: якщо у рядку не залишилось символів - закрити таблицю та завершити цикл. Друга умова перевіряє можливість запису символу в межах стовпчика: якщо символ можна записати - він записується, інакше - в else відбувається перехід на новий рядок, створення порожнього першого стовпчика, та продовження запису, поки не завершаться символи. У кінці циклу реалізовано умова, якщо обрано запис у файл - виводиться відповідне повідомлення.

### **Артефакти процесу реалізації модуля “ModulesOzhekhovskyi”**

Даний модуль реалізує додавання людини до реєстру та його видалення. Модуль реалізований за допомогою 6 функцій: inputTextData(), inputNumData(), inputYearData(), inputAdditions(), addElement(), deleteElement(). (Перші 4 функції створені для заповнення функції addElement()).

inputTextData() - реалізує введення текстової інформації, таку як: ім'я, прізвище, по батькові, марка авто, державний номер. Функція забезпечує правильне введення за довжиною рядка.

inputNumData() - реалізує введення дня і місяця. Функція забезпечує правильне введення кількості днів в місяці та кількості місяців. Якщо введене число - це число від 1 до 9 то підставляється нуль перед ним для правильного виведення інформації.

inputYearData() - реалізує введення року випуску авто та року реєстрування. Функція забезпечує правильне введення року. Введене число не може бути меншим за 1800 та більшим за 2022.

inputAdditions() - реалізує введення додаткової інформації. Функція забезпечує правильне введення за довжиною рядка.

addElement() - реалізує додавання нової людини в реєстр. Функція створює новий вузол в лінійному спискові в кінці списку та поєднує функції, які були описані вище. Передає змінним інформацію, яку введе користувач символьного типу.

deleteElement() - реалізує видалення вузла по вказаному користувачем державному номеру. Видалення може відбуватися як для кореня списку так і для будь якого його елемента.

# ВИСНОВКИ

Невід’ємною складовою виконання лабораторної роботи можна вважати збори на обговорення реалізації архітектури програмного забезпечення, саме через важливість доцільної реалізації функцій та попереднього врахування всіх опущень та обмежень. Всупереч пунктам, визначеними в плані методичних рекомендацій, колективні збори відбулися не один раз. Команда прийняла рішення наслідувати спосіб виконання проекту за методологією Agile (після реалізації певної частини програми, всі збиралися на обговорення подальших реалізацій). Проте необхідно зазначити, що детальна історія змін після кожного спринту (або певної версії програми) не відслідковується на репозиторіях жодного з учасників.

Під час вибору динамічної структури та обговорення способу запису/зчитування інформації з файлу була ідея використати векторний масив в якості буферного приймача даних, що мав достатньо зручні, для обробки елементів, методи. Проте можливості, що надає динамічна пам’ять, були доцільніші з погляду оптимізації коду та задовільного виконання умов лабораторної.

Поля елементу динамічної структури піддавались змінам, через натходження нових ідей щодо методів їх обробки. Користуючись початковими теоретичним пізнанням у записі певних елементів у файл, поля структури були визначені рядками типу string. В зв’язку неможливості визначення кількості байт для запису/зчитування, тип рядків був змінений на char\*, проте з даним типом був можливий запис інформації не за елементами структури, а за полями елемента. В результаті, було прийнято рішення зупинитись на варіанті з типом рядка char[], що був зручним для запису/зчитування та ініціалізації полів структури.

З метою впевнитися у істинності виведення/запису інформації у бінарний файл, й виходячи з відсутності методів відкриття файлів у шістнадцятирічному форматі - використано безкоштовне програмне забезпечення “Far Manager 3”. Серед можливого функціоналу, що був використаний під час роботи - перегляд файлів будь-якого розширення у текстовому та шістнадцятирічному форматі.

Запис даних у файл був реалізований наступним чином. На початку функції була реалізована перевірка на те, чи є перший елемент порожнім. Якщо елемент порожній - виводиться повідомлення про неможливість записати порожню інформацію до регістру і робота застосунку завершується, інакше - робота функції продовжується. Далі відкривається бінарний файл з очищенням попередньої інформації, та переміщенням курсору на початок файлу. За допомогою циклу (умова - поки вказівник на елемент не є порожнім) відбувається запис даних у файл.

Виведення інформації у консоль або у вказаний файл реалізовано наступним чином. На початку функції так само реалізовано перевірку на те, чи є перший елемент порожнім із повідомленням про неможливість показати порожній реєстр. Принцип роботи аналогічний до того, який був реалізований у першій функції. Далі також відкривається бінарний файл з очищенням попередньої інформації, та переміщенням курсору на початок файлу. Після цього було реалізовано макрос-функцію, який, в залежності від вибору користувача, виводив дані у консоль або файл. За допомогою циклу do while(умова - цикл відбувається поки існує наступний елемент) було реалізовано форматоване виведення бази даних у консоль/файл (у вигляді таблиці). Для форматування використовувався маніпулятор setw()із бібліотеки <iomanip.h>, який заповнює вказаною в дужках кількістю пробілів. Для зручності користувача було прийняте рішення записати місяць, день та рік в один рядок, а для зручності форматування було прийняте рішення перед місяцем та днем дописувати 0 (якщо місяць і/або день < 10). Задля зручного відображення додатків було прийняте рішення за потреби записувати їх у декілька рядків. Виходячи із максимального розміру символів у додатках (256) було прийняте рішення зробити другий стовпчик довжиною 64 символи (за такої довжини стовпчика максимальна кількість символів у додатку поміститься у 4 рядки). За допомогою циклу та двох умов був реалізований коректний запис додатків, який задовольнив рішення про їх запис у декілька рядків.

Під час роботи над тест сьютом, а саме роботи над кейсом запису інформації до реєстру при завершенні програми, було прийняте наступне рішення. Для того, щоб належно показати коректний запис даних при завершенні роботи застосунку було прийняте рішення створити новий елемент реєстру та завершити роботу застосунку, потім знову запустити застосунок, та обрати виведення реєстру у консоль. Таким чином було перевірено, чи записується інформація при завершенні роботи застосунку.

Під час проектування програмного модуля виникло декілька проблем: використання рекурсії та спотворення даних. Рекурсія була замінена циклами, адже цикл дозволяє ітерувати запис елементів необмежену кількість разів. Проблема із спотворенням даних при записі у файл була вирішена використанням буферного вказівника, адже дані спотворювались саме через роботу напряму із вказівниками на елементи структури.

Додавання людини в реєстр відбувався наступним чином. Створили окремі функції для вводу інформації, для текстових значень, для значень дня і місяця, для значень року та для додаткової інформації.

Функція для вводу текстових значень приймає 3 аргументи: максимальна довжина вхідного тексту, змінна в яку буде записуватися інформація та текст повідомлення. В функції робимо цикл, який перевіряє довжину введеної інформації, якщо довжина інформації більше 0 та менше вказаної максимальної довжини то зупиняємо цикл та повертаємо введену інформацію рядкового типу.

Функція для вводу дня і місяця приймає 3 аргументи: змінна в яку буде записуватися число, максимальне значення, та текст повідомлення. В функції робимо цикл, який перевіряє введене число. Введене число переводимо з рядкового типу в цілочисельний тип для перевірки його правильності, якщо введене число більше 0 та менше 10, то ми записуємо в змінну рядкового типу 0 та додаємо до нього рядок з введеним числом та повертаємо цю змінну, а якщо число більше ніж 9 та менше максимального значення, то записуємо його в змінну рядкового типу та повертаємо цю змінну.

Функція для введення року приймає 2 аргументи: змінна в яку буде записуватися рік та текст повідомлення. В функції робимо цикл, який перевіряє правильність введеного року. Для цього ми переводимо введене значення рядкового типу в число та перевіряємо це число, якщо воно більше 1800 та менше 2023, то повертаємо це значення рядкового типу.

Функція для введення додаткової інформації приймає 2 аргументи: змінна в яку буде записуватися інформація та текст повідомлення. В функції робимо цикл, який перевіряє довжину введеної інформації, якщо її довжина менше 255, то повертаємо це значення рядкового типу. При написанні цієї функції виникла проблема з введенням інформації, виправило цю проблему написання cin.ignore() перед getline(cin, data) і функція почала працювати правильно.

Головна функція, яка додає нову людину в реєстр приймає один аргумент - корінь динамічної структури. Функція створює новий вузол динамічної структури в кінці списку та записує в її змінні інформацію, яку ми отримали за допомогою минулих чотирьох функцій. Запис в змінні структури відбувається за допомогою копіюванням змінної в яку ми ввели значення (та перевели з рядкового типу в символьний тип) в змінну структури.

Видалення людини з реєстру відбувається наступним чином.

Функція видалення людини з реєстру приймає один аргумент - корінь структури. Видалення відбувається по пошуку державного номера. Користувач вводить державний номер, який хоче видалити з реєстру та відбувається пошук по вузлам структури, коли знайшли потрібний вузол, то перевіряємо чи є він корнем списку, якщо так, то встановлюємо вказівник на наступний вузол структури та видаляємо цей вузол, а якщо потрібний вузол не є корнем, то встановлюємо вказівник з попереднього вузлу на наступний та видаляємо потрібний нам вузол.

В процесі мітингів, вирішено, реалізацію програмного забезпечення (project\_1\_Прізвище.exe), метою якого є демонстрація роботи статичних бібліотек кожного учасника команди, виконати сумісно. Рішення реалізації ПЗ кожним учасником окремо не є раціональним через факт командного обговорення архітектури ПЗ. Згідно з обраними методами, застосунки однієї команди, що написані окремо не матимуть вагових відмінностей в структурі коду та роботі програми.

Додатково оформлено три функції в project\_1\_Прізвище згідно структурною парадигмою програмування (виконано задля спрощення частини коду, реалізованої в головній процедурі main()).

fileCheckAndRead() - перевіряє можливість відкриття файлу й наскільки він є заповененний.

readFromFileToPointers() - виконує додаткові необхідні маніпуляції для зчитування інформації з бінарного файлу.

actionMenu() - функція з можливістю виклику інших функцій. Базується на умові завдання (забезпечує за вибором користувача виконання певних процедур).

**ДОДАТОК А**

(тестовий набір Artifact\_TEST\_SUITE\_lab.docx)

|  |  |  |
| --- | --- | --- |
| Artifact: Test Suite | Назва тестового набору /  **Test Suite Description** | TEST\_SUITE |
| Date: 01/05/2022 | Назва проекта/ПЗ /  **Name of project** | project\_1\_Striuk.ехе |
|  | Рівень тестування /  **Level of testing** | Системний |
|  | Автор тест-сьюта /  **Test Suite Author** | Владислав Стрюк |
|  | Виконавець /  **Implementer** | Владислав Стрюк |

|  |  |  |  |
| --- | --- | --- | --- |
| Ід-р тест кейса /  **Test Case ID** | Дії (кроки)  /  **Action (Test Steps)** | Очікуваний результат  /  **Expected Result** | Результат тестування  /  **Test Result** |
| TS.1.1 | 1. Переконатися, що RegisterDataBase.dat – пустий  2. Запустити застосунок | File is empty! Initializing default enrollment  ====================================  | <<< Reading complete >>> |  ==================================== | Passed |
| TS.1.2 | 1. Переконатися, що RegisterDataBase.dat – відсутній  2. Запустити застосунок | Could not open the data file! Exit program... | Passed |
| TS.1.3 | 1. Переконатися, що RegisterDataBase.dat містить інформацію  2. Запустити застосунок | ====================================  | <<< Reading complete >>> |  ==================================== | Passed |
| TS.2.1 | 1. Запустити за стосунок  2. Натиснути «5» | > 5  Enter government number, which you wanna find: | Passed |
| TS.2.2 | 1. Запустити за стосунок  2. Натиснути «5»  3. Ввести частину державного номера, що відсутня в базі даних | > 5  Enter government number, which you wanna find: P | Passed |
| TS.2.3 | 1. Запустити за стосунок  2. Натиснути «5»  3. Ввести частину державного номера, що присутня в базі даних | > 5  Enter government number, which you wanna find: B  Vladislav Striuk Evgeniyovych: BA7777AB  Volodymyr Chornuy Maksymovych: AK9845BB | Passed |
| TS.2.4 | 1. Запустити за стосунок  2. Натиснути «5»  3. Ввести повний державний номер | Enter government number, which you wanna find: AA0000AA  Ivan Ivanov Ivanovich: AA0000AA | Passed |
| TS.3.1 | 1. Запустити за стосунок  2. Натиснути «6» | > 6  «Завершення роботи за стосунку» | Passed |

**ДОДАТОК Б**

(тестовий набір Artifact\_TEST\_SUITE\_lab.docx)

|  |  |  |
| --- | --- | --- |
| Artifact: Test Suite | Назва тестового набору /  **Test Suite Description** | TEST\_SUITE |
| Date: 01/05/2022 | Назва проекта/ПЗ /  **Name of project** | project\_1\_Baramba.ехе |
|  | Рівень тестування /  **Level of testing** | Системний |
|  | Автор тест-сьюта /  **Test Suite Author** | Барамба Андрій |
|  | Виконавець /  **Implementer** | Барамба Андрій |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Action (Test Steps)** | **Expected Result** | **Test Result** |
| TC-01 | 1. Запустити застосунок 2. Ввести “1” 3. Натиснути Enter | Вікно застосунку:  ----------------------------------------------------------------------------------  |First name |Ivan |  |Last name |Ivanov |  |Patronymic |Ivanovich |  |Car brand |Toyota |  |Graduation year |2012 |  |Registration day |12.12.2022 |  |Registration code|AA0000AA |  |Additions |Without front left door |  ---------------------------------------------------------------------------------- | Passed |
| TC-02 | 1. Запустити застосунок 2. Ввести “2” 3. Натиснути Enter | Вміст файлу:  ----------------------------------------------------------------------------------  |First name |Ivan |  |Last name |Ivanov |  |Patronymic |Ivanovich |  |Car brand |Toyota |  |Graduation year |2012 |  |Registration day |12.12.2022 |  |Registration code|AA0000AA |  |Additions |Without front left door |  ---------------------------------------------------------------------------------- | Passed |
| TC-03 | 1. Запустити застосунок 2. Ввести “4” 3. Ввести “АА0000АА” 4. Ввести “6” | Вікно застосунку:  Error: can't write an empty information | Passed |
| TC-04 | 1. Запустити застосунок 2. Ввести 4 3. Ввести “АА0000АА” 4. Ввести 1 | Вікно застосунку:  Error: can't show an empty information | Passed |
| TC-05 | 1. Запустити застосунок 2. Ввести “3” 3. Ввести “Andrew” 4. Ввести “Baramba” 5. Ввести “Andreevich” 6. Ввести “Toyota” 7. Ввести “2020” 8. Ввести “14” 9. Ввести “2” 10. Ввести “2021” 11. Ввести “BA2431AX” 12. Ввести “Need a cup of coffee” 13. Ввести “6” 14. Запустити за стосунок 15. Ввести “1” | Вікно застосунку:  ----------------------------------------------------------------------------------  |First name |Ivan |  |Last name |Ivanov |  |Patronymic |Ivanovich |  |Car brand |Toyota |  |Graduation year |2012 |  |Registration day |12.12.2022 |  |Registration code|AA0000AA |  |Additions |Without front left door |  ----------------------------------------------------------------------------------  | |  ----------------------------------------------------------------------------------  |First name |Andrew |  |Last name |Baramba |  |Patronymic |Andreevich |  |Car brand |Toyota |  |Graduation year |2020 |  |Registration day |14.02.2021 |  |Registration code|BA2431AX |  |Additions |Need a cup of coffee |  ---------------------------------------------------------------------------------- | Passed |

**ДОДАТОК В**

(тестовий набір Artifact\_TEST\_SUITE\_lab.docx)

|  |  |  |
| --- | --- | --- |
| Artifact: Test Suite | Назва тестового набору /  **Test Suite Description** | TEST\_SUITE |
| Date: 01/05/2022 | Назва проекта/ПЗ /  **Name of project** | project\_1\_Ozhekhovskyi.ехе |
|  | Рівень тестування /  **Level of testing** | Системний |
|  | Автор тест-сьюта /  **Test Suite Author** | Владислав Ожеховський |
|  | Виконавець /  **Implementer** | Владислав Ожеховський |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Action (Test Steps)** | **Expected Result** | **Test Result** |
| TS.1.1 | 1. Запустити за стосунок  2. Натиснути «3» | Enter your name: | Passed |
| TS.1.2 | 1. Запустити за стосунок  2. Натиснути «3»  3. Вводити інформацію яку в нас запитують  4. Натиснути «1» | |First name |Vladyslav |  |Last name |Ozhekhovskyi |  |Patronymic |Volodymyrovych |  |Car brand |Renault |  |Graduation year |2012 |  |Registration day |11.03.2020 |  |Registration code|BA2525AB |  |Additions |color: brown | | Passed |
| TS.1.3 | 1. Запустити за стосунок  2. Натиснути «3»  3. В полі «Enter your graduation year:» ввести 2023 | Enter your graduation year: 2023  Enter your graduation year: | Passed |
| TS.2.1 | 1. Запустити за стосунок  2. Натиснути «4» | Choose element for deleting by state number: | Passed |
| TS.2.2 | 1. Запустити за стосунок  2. Натиснути «4»  3. Ввести повний державний номер який хочемо видалити з реєстру  4. Натиснути «1» перевірити чи видалився запис з реєстру | Choose element for deleting by state number: АА0000АА | Passed |
| TS.2.3 | 1. Запустити за стосунок  2. Натиснути «4»  3. Ввести повний державний номер який хочемо видалити з реєстру  4. Натиснути «1» перевірити чи видалився запис з реєстру | Choose element for deleting by state number: BA2525AB  Error: can't show an empty information | Passed |

# ДОДАТОК Г

(Вихідний код ModulesStriuk)

**… / ModulesStriuk / main.cpp**

#include "ModulesStriuk.h"

using namespace std;

regEnrollment\* initializeDefaultRoot(fstream &file)

{

regEnrollment \*regEn = new regEnrollment;

strcpy(regEn->firstName, "Ivan");

strcpy(regEn->lastName, "Ivanov");

strcpy(regEn->patronymic, "Ivanovich");

strcpy(regEn->carBrand, "Toyota");

strcpy(regEn->gradYear, "2012");

strcpy(regEn->dateDay, "12");

strcpy(regEn->dateMonth, "12");

strcpy(regEn->dateYear, "2022");

strcpy(regEn->govNumber, "AA0000AA");

strcpy(regEn->additions, "Without front left door");

regEn->ptr = nullptr;

file.write((char\*)&\*regEn, sizeof(regEnrollment));

return(regEn);

}

void readFromFile(fstream &file, regEnrollment \*rootNode)

{

rootNode->ptr = nullptr;

file.read((char\*)&\*rootNode, sizeof(regEnrollment));

if (file.peek() != -1) {

rootNode->ptr = new regEnrollment;

readFromFile(file, rootNode->ptr);

}

}

void searchInDatabase(regEnrollment \*rootNode)

{

regEnrollment \*bufEnrollment = rootNode;

string bufString;

cout << "Enter government number, which you wanna find: ";

cin >> bufString;

bufString.substr(0, GOV\_NUMBER\_SIZE);

do {

string bufGovNumber = bufEnrollment->govNumber;

size\_t foundStr = bufGovNumber.find(bufString);

if (foundStr != string::npos) {

cout << bufEnrollment->firstName << " " << bufEnrollment->lastName << " " << bufEnrollment->patronymic << ": "

<< bufEnrollment->govNumber << endl;

}

bufEnrollment = bufEnrollment->ptr;

} while (bufEnrollment != nullptr);

}

**… / ModulesStriuk / ModulesStriuk.h**

#ifndef MODULESSTRIUK\_H\_INCLUDED

#define MODULESSTRIUK\_H\_INCLUDED

#include "struct\_type\_project\_1.h"

#include <fstream>

#include <iostream>

#include <string>

#include <cstring>

using namespace std;

regEnrollment\* initializeDefaultRoot(fstream &file);

void readFromFile(fstream &file, regEnrollment \*rootNode);

void searchInDatabase(regEnrollment \*rootNode);

#endif // MODULESSTRIUK\_H\_INCLUDED

# ДОДАТОК Д

(Вихідний код ModulesBaramba)

**… / ModulesBaramba / main.cpp**

#include "ModulesBaramba.h"

using namespace std;

void writeToRegistData(fstream &file, regEnrollment \*regEn)

{

if (regEn == nullptr) { cout << "Error: can't write an empty information" << endl; return; }

file.close();

file.open("RegisterDataBase.dat", ios::binary | ios::out | ofstream::trunc);

file.seekg(0);

regEnrollment \*buffer = regEn;

while (buffer != nullptr) {

file.write((char\*)&\*buffer, sizeof(regEnrollment));

buffer = buffer->ptr;

}

file.close();

}

void dataOutput(fstream &file, regEnrollment \*regEn, int action) {

if (regEn == nullptr) { cout << "Error: can't show an empty information" << endl; return; }

#define OUTPUT(stream) ((stream) == 1 ? cout : file)

regEnrollment \*buffer = regEn;

bool isFirstElem = true;

file.seekg(0);

do {

if (isFirstElem == false) { buffer = buffer->ptr; }

OUTPUT(action) << " ----------------------------------------------------------------------------------" << endl << left;

OUTPUT(action) << "|" << setw(17) << "First name" << "|" << setw(64) << buffer->firstName << "|" << endl

<< "|" << setw(17) << "Last name" << "|" << setw(64) << buffer->lastName << "|" << endl

<< "|" << setw(17) << "Patronymic" << "|" << setw(64) << buffer->patronymic << "|" << endl

<< "|" << setw(17) << "Car brand" << "|" << setw(64) << buffer->carBrand << "|" << endl

<< "|" << setw(17) << "Graduation year" << "|" << setw(64) << buffer->gradYear << "|" << endl

<< "|" << setw(17) << "Registration day" << "|" << buffer->dateDay << "."

<< buffer->dateMonth << "."

<< buffer->dateYear << setw(54) << "" << "|" << endl

<< "|" << setw(17) << "Registration code" << "|" << setw(64) << buffer->govNumber << "|" << endl;

OUTPUT(action) << "|" << setw(17) << "Additions" << "|";

for(int i = 0, line = 0; line < 4; i++) {

if(buffer->additions[line \* 64 + i] == 0) {

OUTPUT(action) << setw(64 - i) << "" << "|" << endl;

break;

}

if((i+1) % 64 != 0) {

OUTPUT(action) << buffer->additions[line \* 64 + i];

}

else {

OUTPUT(action) << " |" << endl << "|" << setw(17) << "" << "|";

line++;

i = -1;

}

}

OUTPUT(action) << " ----------------------------------------------------------------------------------" << endl;

if (buffer->ptr != nullptr) {

OUTPUT(action) << "|" << setw(81) << "" << " |" << endl;

}

isFirstElem = false;

} while (buffer->ptr != nullptr);

if (action != 1 && buffer->ptr == nullptr) { cout << "Information writed to TextFile.txt" << endl; }

}

**… / ModulesBaramba / ModulesBaramba.h**

#ifndef MODULESBARAMBA\_H\_INCLUDED

#define MODULESBARAMBA\_H\_INCLUDED

#include <iostream>

#include <fstream>

#include <iomanip>

#include "struct\_type\_project\_1.h"

using namespace std;

void writeToRegistData(fstream &file, regEnrollment \*regEn);

void dataOutput(fstream &file, regEnrollment \*regEn, int action);

#endif // MODULESBARAMBA\_H\_INCLUDED

# ДОДАТОК Е

(Вихідний код ModulesOzhekhovskyi)

**… / ModulesOzhekhovskyi / main.cpp**

#include "ModulesOzhekhovskyi.h"

string inputTextData(int numBytes, string data, string msg)

{

do {

cout << msg;

cin >> data;

} while (data.length() < 1 || data.length() > numBytes);

return data;

}

string inputNumData(string data, int maxNum, string msg)

{

string num = "";

int intNum = 0;

do {

cout << msg;

cin >> num;

string::size\_type sz;

intNum = stoi(num, &sz);

if (intNum > 0 && intNum < 10) {

data += num;

return data;

} else if (intNum > 9 && intNum <= maxNum){

data = num;

return data;

}

} while(intNum < 1 || intNum > maxNum);

}

string inputYearData(string data, string msg)

{

int intNum = 0;

do {

cout << msg;

cin >> data;

string::size\_type sz;

intNum = stoi(data, &sz);

if (intNum >= 1800 && intNum < 2023) return data;

} while(intNum < 1800 || intNum > 2022);

}

string inputAdditions(string data, string msg)

{

do {

cout << msg;

cin.ignore();

getline(cin, data);

if (data.length() <= 255) return data;

} while(data.length() > 255);

}

void addElement(regEnrollment \*rootNode)

{

regEnrollment \*bufNode = rootNode;

regEnrollment \*newElement = new regEnrollment;

while (bufNode->ptr != nullptr) {

bufNode = bufNode->ptr;

}

bufNode->ptr = newElement;

newElement->ptr = nullptr;

string name = "";

string lastName = "";

string patronymic = "";

string carBrand = "";

string gradYear = "";

string dateDay = "0";

string dateMonth = "0";

string dateYear = "";

string govNum = "";

char additions[255] = "";

strcpy(newElement->firstName, inputTextData(35, name, "Enter your name: ").c\_str());

strcpy(newElement->lastName, inputTextData(35, lastName, "Enter your last name: ").c\_str());

strcpy(newElement->patronymic, inputTextData(35, patronymic, "Enter your patronymic: ").c\_str());

strcpy(newElement->carBrand, inputTextData(35, carBrand, "Enter your car brand: ").c\_str());

strcpy(newElement->gradYear, inputYearData(gradYear, "Enter your graduation year: ").c\_str());

strcpy(newElement->dateDay, inputNumData(dateDay, 31, "Enter the day of registration: ").c\_str());

strcpy(newElement->dateMonth, inputNumData(dateMonth, 12, "Enter the month of registration: ").c\_str());

strcpy(newElement->dateYear, inputYearData(dateYear, "Enter the year of registration: ").c\_str());

strcpy(newElement->govNumber, inputTextData(8, govNum, "Enter your state number: ").c\_str());

strcpy(newElement->additions, inputAdditions(additions, "Enter additions informations: ").c\_str());

}

void deleteElement(regEnrollment \*&root)

{

regEnrollment \*findNode = root;

regEnrollment \*buffer = nullptr;

char searchRequest[9] = "";

cout << endl << "Choose element for deleting by state number: ";

cin >> searchRequest;

do {

if (int(strcmp(searchRequest, findNode->govNumber)) == 0) {

if (findNode == root) {

root = root->ptr;

}

else {

buffer->ptr = findNode->ptr;

}

delete(findNode);

break;

}

buffer = findNode;

findNode = findNode->ptr;

} while (findNode != nullptr);

}**… / ModulesOzhekhovskyi / ModulesOzhekhovskyi.h**

#ifndef MODULESOZHEKHOVSKYI\_H\_INCLUDED

#define MODULESOZHEKHOVSKYI\_H\_INCLUDED

#include "struct\_type\_project\_1.h"

#include <iostream>

#include <string>

#include <cstring>

using namespace std;

string inputTextData(int numBytes, string data, string msg);

string inputNumData(string data, int maxNum, string msg);

string inputYearData(string data, string msg);

string inputAdditions(string data, string msg);

void addElement(regEnrollment \*lst);

void deleteElement(regEnrollment \*&root);

#endif // MODULESOZHEKHOVSKYI\_H\_INCLUDED