Міністерство освіти і науки України

Чернівецький національний університет імені Юрія Федьковича

\_\_\_\_\_\_\_Програмного забезпечення комп’ютерних систем\_\_\_\_\_\_\_

(повна назва кафедри, циклової комісії)

**КУРСОВИЙ ПРОЄКТ**

з Принципи конструювання програмного забезпечення

(назва дисципліни)

на тему:\_ **Створення системи керування**

**базою даних в готелі засобами мови С++**

Виконав: студент (ка) \_\_2\_\_\_ курсу, групи\_\_\_243\_\_\_\_

спеціальності\_\_\_\_\_\_\_\_\_\_\_\_\_\_121\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(шифр спеціальності)

\_\_\_\_Інженерія програмного забезпечення\_\_\_\_\_

(назва спеціальності)

\_\_\_\_\_\_\_\_\_\_ \_\_\_Худенко Ю.С.\_\_\_\_

(підпис) (прізвище, ініціали)

Керівник \_\_\_\_\_\_\_Валь О.О.\_\_\_\_\_\_

(прізвище, ініціали)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Дата захисту** « » 20 р. | | **Оцінка:** | | за національною шкалою | | (словами) | | кількість балів | | (цифра) | | за шкалою ECTS | | (літера) | |  |  | | | |
|  |  |  | | | |
|  |  |  | | | |
|  |  |  | | | |
|  |  |  | | | |
|  |  |  | | | |
|  | | |  |  |  | |
|  | | |  | (літера) | (літера) | |

Чернівці, 2024

Чернівецький національний університет імені Юрія Федьковича

Кафедра Програмного забезпечення комп’ютерних систем\_\_\_

Спеціальність Інженерія програмного забезпечення

**ЗАВДАННЯ**

**на курсовий проєкт студента**

\_ Худенко Юлії Сергіївни

(прізвище, ім’я, по-батькові)

1. Тема проєкту: Система керування базою даних "Готель"

1. Вихідні дані до проєкту:
   * розробити базу даних – текстовий файл, в який будуть заноситись дані про клієнта;
   * розробити можливість додавати нові записи в базу даних;
   * розробити можливість переглядати записи в базі даних;
   * розробити можливість змінювати записи в базі даних;
   * розробити можливість видаляти записи з бази даних;
   * розробити можливість сортувати записи в базі даних за номером кімнати;
   * розробити збереження даних у файл;
   * розробити пошук клієнта за номером кімнати;
   * розробити виведення номерів готелю та їхньої вартості ;
2. Зміст розрахунково-пояснювальної записки (перелік питань, які необхідно розробити):
   * описати загальні вимоги до програми;
   * описати призначення та область застосування;
   * описати модулі проєкту та алгоритм виконання;
   * описати методи програми;
   * описати користувацький інтерфейс;
   * зробити висновки по розробленій програмі.
3. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень):
   * скріншоти роботи з програмою;
   * блок-схеми роботи з програмою;
   * відео роботи з програмою.

Завдання прийняв до виконання \_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис студента)

Керівник проєкту \_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис керівника)

**РЕФЕРАТ**

У курсовому проєкті розроблено систему управління базою даних «Готель» для комп’ютерів, які працюють під керуванням операційної системи Windows.

Програмне забезпечення розраховане на користувачів, які не мають спеціальної комп’ютерної підготовки. Досвід роботи не вимагається.

Область застосування – програмне забезпечення для ноутбуків або персональних комп’ютерів, що працюють на операційній системі Windows.

Програма реалізована засобами середовища Visual Studio мовою C++. Дане середовище є зручним у використанні для швидкого та якісного створення додатків на OC Windows.

Ця система може бути змінена із додаванням нового функціоналу та зміною користувацького інтерфейсу в майбутньому.

Курсовий проєкт містить 44 с., 15 рис., 5 табл., 1 додатки, 8 джерел.

*ОПЕРАЦІЙНА СИСТЕМА, БАЗА ДАНИХ, СИСТЕМА УПРАВЛІННЯ БАЗОЮ ДАНИХ, WINDOWS, VISUALSTUDIO, C++.*

**SUMMARY**

In this course project, a database management system "Hotel" has been developed for computers operating under the Windows operating system.

The software is designed for users without specialized computer training, and no prior experience is required.

Application area – software for laptops or personal computers running the Windows operating system.

The program is implemented using the Visual Studio environment in C++. This environment is convenient for the quick and high-quality development of applications on Windows OS.

This system can be modified with the addition of new functionality and a changed user interface in the future.

The course project includes 44 pages,15 figures, 5 tables, 1 appendix, and 8 references.

*OPERATING SYSTEM, DATABASE, DATABASE MANAGEMENT SYSTEM, WINDOWS, C++, WINDOWS FORM.*

**ЗМІСТ**

[ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ……………………………………………6](#_Toc121056468)

[1. АРХІТЕКТУРА ТА ФУНКЦІОНАЛЬНІ ПОКАЗНИКИ 7](#_Toc121056469)

[1.1 Загальні вимоги до програми 7](#_Toc121056470)

[1.2 Призначення та область застосування 9](#_Toc121056474)

[1.3 Функціональні вимоги 9](#_Toc121056475)

[2. ОПИС ПРОГРАМИ 10](#_Toc121056476)

[2.1 Структура програми 10](#_Toc121056477)

[2.2 Опис методів програми 13](#_Toc121056480)

[2.3 Програмні засоби 19](#_Toc121056481)

[2.4 Опис користувацького інтерфейсу 20](#_Toc121056482)

[ВИСНОВКИ 28](#_Toc121056483)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 29](#_Toc121056484)

[ДОДАТКИ 30](#_Toc121056485)

[Додаток А. Скролінг (текст) програми 30](#_Toc121056486)

**ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ**

ОС – операційна система.

ПЗ – програмне забезпечення.

**1. АРХІТЕКТУРА ТА ФУНКЦІОНАЛЬНІ ПОКАЗНИКИ**

* 1. **Загальні вимоги до програми**

**1.1.1 Вимоги до графічного інтерфейсу користувача**

1. Робоча мова інтерфейсу – українська.
2. Використовувати терміни зрозумілі для користувача.
3. Навігаційна панель (меню), яка повинна забезпечувати перегляд, редагування, додавання, зчитування та видалення даних.
4. Панель для фільтрації, сортування та пошуку по ключу
5. «Посібник користувача» з усією потрібною інформацією для користування системою.
6. Повідомлення про некоректно введені дані.
7. Консольна програма, тому колірна гамма за замовчуванням.
   * 1. **Вимоги до архітектури програми**
8. Використання стандартних контейнерних класів, які реалізують:

* Введення даних про клієнта вручну та з файлу
* Вивід даних
* Змінення даних
* Сортування даних
* Збереження даних у файл
* Пошук клієнта за номером кімнати
* Виведення номерів готелю та їх вартості

1. Стійкість програми:

* Програма повинна не втрачати працездатності при будь-яких діях користувача;
* Інформація, що вводиться, скрізь, де це можливо, піддається логічному забезпеченню цілісності даних;
* При будь-яких діях користувача не повинні втрачатись дані або їх цілісність.
  + 1. **Вимоги до функціональності додатка**

1. Створення файлу бази даних.
2. Додавання даних про клієнта до бази даних.
3. Перегляд введених або зчитаних з файлу даних.
4. Редагування даних.
5. Видалення даних.
6. Сортування даних за номером кімнати.
7. Збереження даних у файл Input.txt.
8. Пошук за номером кімнати.
9. Виведення номерів в готелі та їх вартості.
10. Наявність посібника користувача для ознайомлення з принципом взаємодії з програмою
11. Можливість виходу з програми.

**1.2 Призначення та область застосування**

**Мета роботи** полягає у розробці системи управління базою даних «Готель» для комп’ютерів, які працюють під керуванням ОС Windows.

Реалізована версія системи управління базою даних готелю призначена для ділового використання працівниками готелю для додавання інформації про користувачів, ПІБ клієнта, серія та номер паспорта, дата народження, адреса, тип та номер кімнати, дата в’їзду та виїзду.

**Область застосування** – програмне забезпечення для ноутбуків або персональних комп’ютерів, що працюють на операційній системі Windows.

**1.3 Функціональні вимоги**

До програмного забезпечення висуваються такі функціональні вимоги:

1. Сторінка (меню) на якій відбуваються всі операції з базою даних.
2. Перехід на сторінку, яка містить довідку по інструкції з використання системи.
3. База даних – це текстовий файл, тому передбачити:

* Збереження даних у файл;
* Зчитування даних з файлу.

1. Меню повинне забезпечувати наступні можливості:

* додавання даних;
* перегляд даних;
* редагування даних;
* видалення даних;
* збереження даних;
* перегляд посібника користувача.

1. Передбачити сортування, фільтрацію та пошук даних за номером кімнати.
2. Передбачити перевірку коректності вводу даних та обробку виключень.

**2. ОПИС ПРОГРАМИ**

* 1. **Структура програми**

**2.1.1 Модулі програми**

Робота розробленого програмного забезпечення реалізується наступними модулями:

1. Hotel – клас в якому відбуваються всі маніпуляції з записом та зчитуванням з бази даних.
2. Initial - клас в якому зберігається ініціали (ПІБ) клієнта.
3. Address - клас в якому зберігаються дані про місце проживання.
4. Date - клас в якому зберігаються дані про дату народження, дату в’їзду та виїзду.
5. Data - клас в якому зберігаються всі дані про клієнта.
6. Subd.hotel - клас який відображає меню та роботу з меню.

**2.1.2 Алгоритми роботи програми**

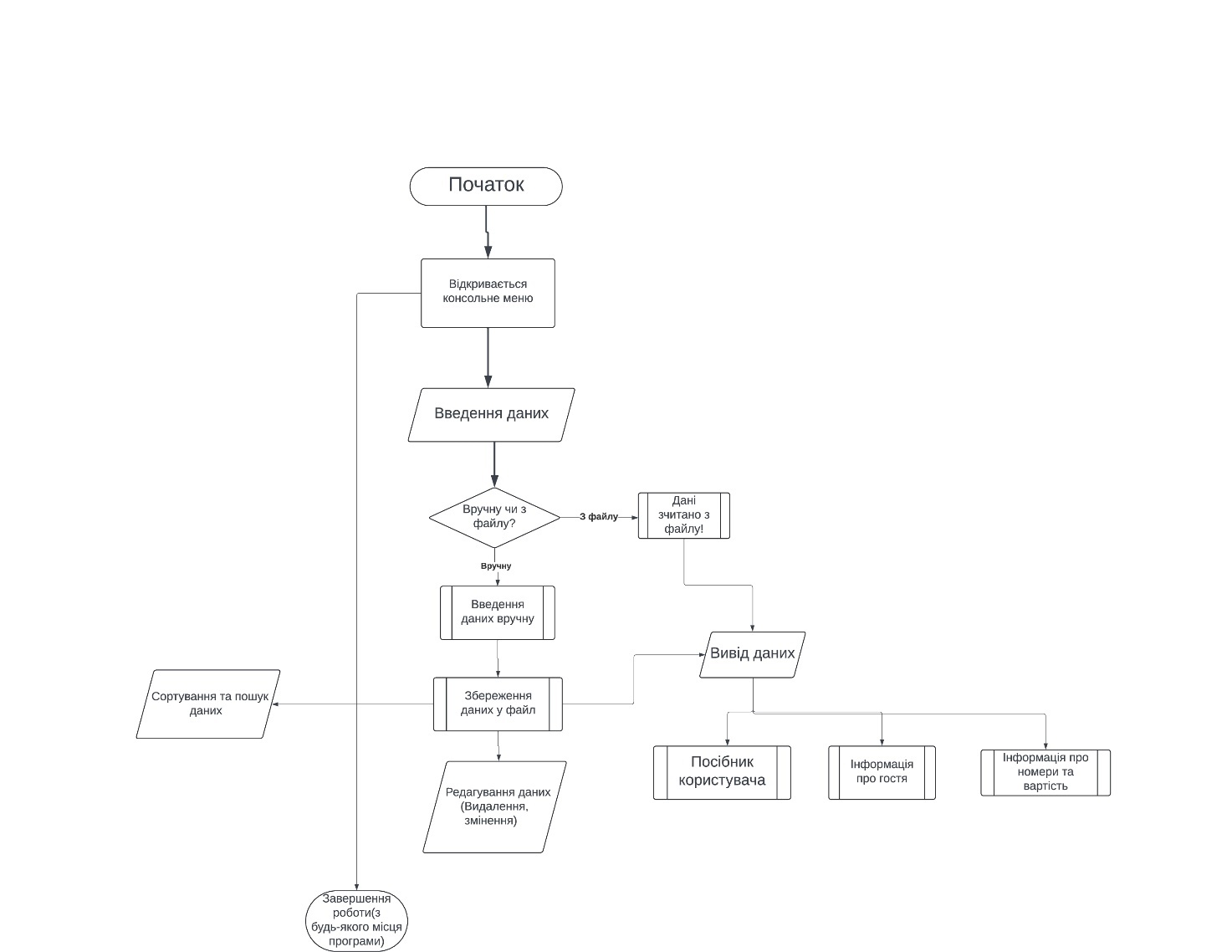


Рисунок 1 – Схема роботи програми

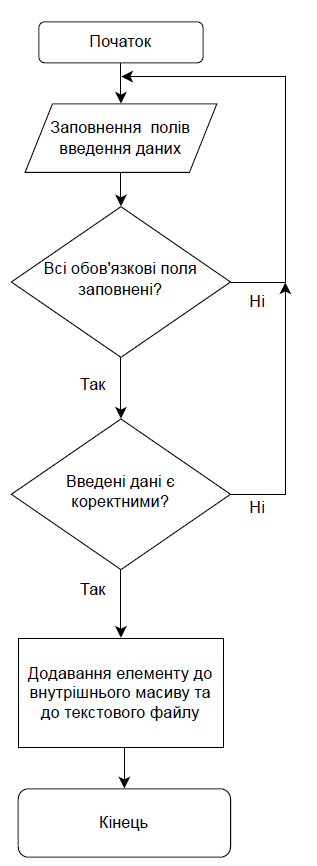


Рисунок 2 – Узагальнена блок-схема алгоритму додавання даних

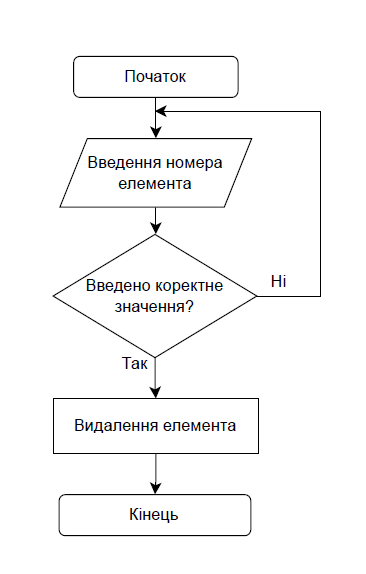


Рисунок 3 – Узагальнена блок-схема алгоритму видалення даних

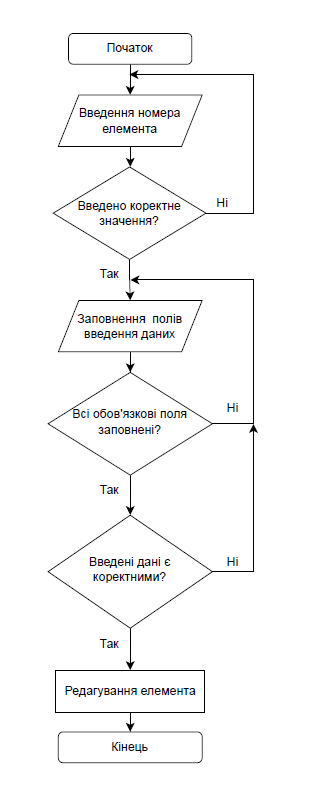


Рисунок 4 – Узагальнена блок-схема алгоритму редагування даних

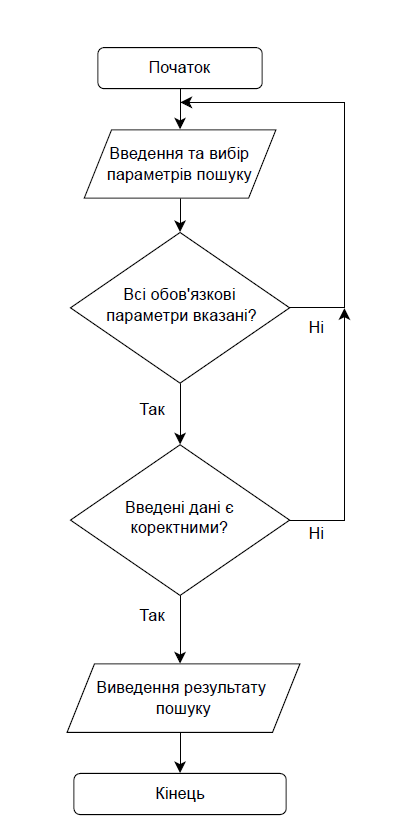


Рисунок 5 – Узагальнена блок-схема

алгоритму пошуку клієнта

* 1. **Опис методів програми**

Список методів класу *Hotel* та їх опис наведено в табл.1

Таблиця 1 – Основні методи класу

|  |  |  |
| --- | --- | --- |
| № | Метод | Короткий опис |
| 1. | *void Hotel::DataEntry(vector<Data>& data)* | Введення даних вручну. |
| 2. | *void Hotel::readDataFromFile(vector<Data>& data, const string& fileName)* | Читання даних з файлу. |
| 3. | *void Hotel::printData(const vector<Data>& data)* | Виведення даних. |
| 4. | *void Hotel::modifyData(vector<Data>& data)* | Редагування даних. |
| 5. | *void Hotel::deleteData(vector<Data>& data)* | Видалення даних. |
| 6. | *void Hotel::sortData(vector<Data>& data)* | Сортування даних за номером кімнати . |
| 7. | *void Hotel::saveDataToFile(const vector<Data>& data, const string& fileName) const* | Збереження даних у файл. |
| 8. | *void Hotel::searchRoom(int roomNumber) const* | Пошук гостя за номером кімнати. |
| 9. | *void Hotel::displayRooms() const* | Виведення номерів та їх вартості. |

Список методів класу *Date* та їх опис наведено в табл.2

Таблиця 2 – Основні методи класу

|  |  |  |
| --- | --- | --- |
| № | Метод | Короткий опис |
| 1. | *Date::Date()* | Ініціалізує дату на 1 січня 2000 року. |
| 2. | *Date::Date(int \_day, int \_month, int \_year)* | Ініціалізує дату з вказаними значеннями дня, місяця та року. |
| 3. | *Date::Date(const Date& other)* | Копіює значення з іншого об'єкта Date. |
| 4. | *Date::Date(Date&& other) noexcept* | Переміщує значення з іншого об'єкта Date і обнуляє його значення. |
| 5. | *int Date::getDay() const* | Повертає день. |
| 6. | *int Date::getMonth() const* | Повертає місяць. |
| 7. | *int Date::getYear() const* | Повертає рік. |
| 8. | *void Date::setDay(int \_day)* | Задає день. |
| 9. | *void Date::setMonth(int \_month)* | Задає місяць. |
| 10. | *void Date::setYear(int \_year)* | Задає рік. |
| 11. | *Date& Date::operator=(const Date& other)* | Перевизначає оператор присвоєння для копіювання значень з одного об'єкта Date в інший. |
| 12. | *std::string Date::toString() const* | Перетворює дату на рядок у форматі dd/mm/yyyy. |

Список методів класу *Address* та їх опис наведено в табл.3

Таблиця 3 – Основні методи класу

|  |  |  |
| --- | --- | --- |
| № | Метод | Короткий опис |
| 1. | *Address();* | Ініціалізує об'єкт Address без параметрів. |
| 2. | *Address(const std::string& \_roomType, int \_roomNumber);* | Ініціалізує об'єкт Address із заданими типом кімнати та номером кімнати. |
| 3. | *Address(const Address& other);* | Копіює значення з іншого об'єкта Address. |
| 4. | *Address(Address&& other) noexcept;* | Переміщує значення з іншого об'єкта Address і обнуляє його значення (якщо потрібно). |
| 5. | *~Address();* | Очищує ресурси, якщо вони є, коли об'єкт знищується. |
| 6. | *std::string getRoomType() const;* | Повертає тип кімнати. |
| 7. | *int getRoomNumber() const;* | Повертає номер кімнати. |
| 8. | *void setRoomType(const std::string& \_roomType);* | Задає тип кімнати. |
| 9. | *void setRoomNumber(int \_roomNumber);* | Задає номер кімнати. |
| 10. | *Address& operator=(const Address& other);* | Перевизначає оператор присвоєння для копіювання значень з одного об'єкта Address в інший. |
| 11. | *std::string toString() const;* | Перетворює адресу на рядок. |

Список методів класу *Initial* та їх опис наведено в табл.4

Таблиця 4 – Основні методи класу

|  |  |  |
| --- | --- | --- |
| № | Метод | Короткий опис |
| 1. | *Initial();* | Ініціалізує об'єкт Initial з порожніми рядками для прізвища, по батькові та імені. |
| 2. | *Initial(const string& \_surname, const string& \_patronymic, const string& \_name);* | Ініціалізує об'єкт Initial заданими значеннями для прізвища, по батькові та імені. |
| 3. | *Initial(const Initial& other);* | Створює новий об'єкт Initial, копіюючи значення з вже існуючого об'єкта (other). |
| 4. | *Initial(Initial&& other) noexcept;* | Переносить значення з іншого об'єкта Initial (other), звільняючи його ресурси. |
| 5. | *string getSurname() const;* | Повертає прізвище. |
| 6. | *string getPatronymic() const;* | Повертає по батькові. |
| 7. | *string getName() const;* | Повертає ім'я. |
| 8. | *void setSurname(const string& \_surname);* | Задає нове значення прізвища. |
| 9. | *void setPatronymic(const string& \_patronymic);* | Задає нове значення по батькові. |
| 10. | *void setName(const string& \_name);* | Задає нове значення імені. |
| 11. | *Initial& operator=(const Initial& other);* | Копіює значення з одного об'єкта Initial (other) в інший. Він також перевіряє на самоприсвоєння, щоб уникнути помилок. |

Список методів класу *Data* та їх опис наведено в табл.5

Таблиця 5 – Основні методи класу

|  |  |  |
| --- | --- | --- |
| № | Методи | Короткий опис |
| 1. | *Data::Data();* | Ініціалізує passportNumber і roomNumber значеннями 0. |
| 2. | *Data::Data(const string& \_passportSeries, int \_passportNumber, Initial&& \_surname, Date&& \_birthDate, Address&& \_address, Date&& \_checkInDate, Date&& \_checkOutDate, int \_roomNumber, const string& \_guestName);* | Ініціалізує об'єкт Data з усіма переданими значеннями, включаючи переміщення об'єктів Initial, Date та Address для ефективності. |
| 3. | *Data::Data(const Data& other);* | Створює новий об'єкт Data, копіюючи значення з вже існуючого об'єкта other. |
| 4. | *Data::Data(Data&& other) noexcept;* | Переносить значення з іншого об'єкта Data, звільняючи ресурси (за потреби). |
| 5. | *Data& Data::operator=(const Data& other);* | Копіює значення з одного об'єкта Data (other) в інший, захищаючи від самоприсвоєння. |
| 6. | *Data& Data::operator=(Data&& other) noexcept;* | Переносить значення з одного об'єкта Data (other) в інший, також захищаючи від самоприсвоєння. |
| 7. | *Initial\* Data::getInitial() const;* | Повертає вказівник на об'єкт Initial, асоційований з даними. |
| 8. | *Date\* Data::getBirthDate() const;* | Повертає вказівник на об'єкт Date, що представляє дату народження. |
| 9. | *Address\* Data::getAddress() const;* | Повертає вказівник на об'єкт Address, що представляє адресу. |
| 10. | *Date\* Data::getCheckInDate() const;* | Повертає вказівник на об'єкт Date, що представляє дату заїзду. |
| 11. | *Date\* Data::getCheckOutDate() const;* | Повертає вказівник на об'єкт Date, що представляє дату виїзду. |
| 12. | *string Data::getPassportSeries() const;* | Повертає серію паспорта. |
| 13. | *int Data::getPassportNumber() const;* | Повертає номер паспорта. |
| 14. | *int Data::getRoomNumber() const;* | Повертає номер кімнати. |
| 15. | *string Data::getGuestName() const;* | Повертає ім'я гостя. |
| 16. | *void Data::setInitial(const Initial& \_initial);* | Задає нове значення об'єкта Initial для даних. |
| 17. | *void Data::setBirthDate(const Date& \_birthDate);* | Задає нову дату народження. |
| 18. | *void Data::setAddress(const Address& \_address);* | Задає нову адресу. |
| 19. | *void Data::setCheckInDate(const Date& \_checkInDate);* | Задає нову дату заїзду. |
| 20. | *void Data::setCheckOutDate(const Date& \_checkOutDate);* | Задає нову дату виїзду. |
| 21. | *void Data::setPassportSeries(const string& \_passportSeries);* | Задає нову серію паспорта. |
| 22. | *void Data::setPassportNumber(int \_passportNumber);* | Задає новий номер паспорта. |
| 23. | *void Data::setRoomNumber(int \_roomNumber);* | Задає новий номер кімнати. |
| 24. | *void Data::setGuestName(const string& \_guestName);* | Задає нове ім'я гостя. |

* 1. **Програмні засоби**

Розробка реалізована засобами середовища Visual Studio на мові програмування C++.

Це середовище зручне для швидкого і якісного створення додатків для операційної системи Windows.

Microsoft Visual Studio – це серія продуктів компанії Microsoft, яка об'єднує інтегроване середовище для розробки програмного забезпечення та різні інструменти для створення додатків. Ці засоби дозволяють розробляти як консольні програми, так і додатки з графічним інтерфейсом, включаючи підтримку Windows Forms. Також доступна розробка вебсайтів, вебзастосунків та вебсервісів як у нативному, так і керованому коді для платформ, підтримуваних Microsoft: Windows, Windows Mobile, Windows Phone, Windows CE, .NET Framework, .NET Compact Framework та Microsoft Silverlight [1].

**2.4 Опис користувацького інтерфейсу**

При запуску програми користувач має можливість обрати одну з запропонованих дій на головному вікні меню, а саме: «Введення даних»(вручну та з фалу), «Вивід даних», «Редагування даних», «Видалення даних», «Додавання даних», «Сортування даних», «Збереження даних», «Пошук за номером кімнати», «Виведення номерів готелю та їх вартості», «Посібник користувача» та «Вихід з програми».

При запуску програми користувачу доступна форма керування системою. Розглянемо функціональні можливості згаданого вікна (рис.6).

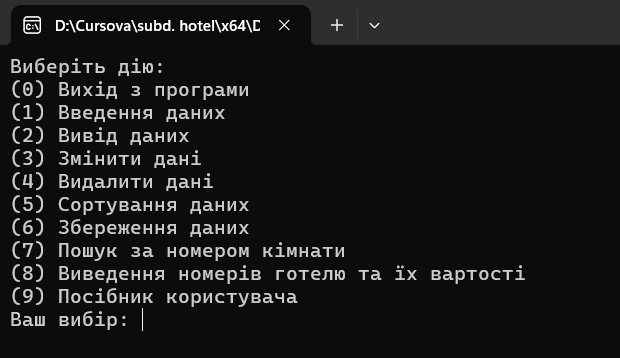


Рисунок 6

На сторінці знаходиться консольне меню, яке має 11 пунктів для додавання, виведення, редагування та видалення даних.

При натисканні на пункт «(1) Введення даних» відкривається нове вікно консолі, де можна обрати спосіб введення даних – вручну(натискаючи на 1) або з файлу(натискаючи на 0). При виборі введення даних вручну потрібно по черзі вписувати серію та номер паспорта, ім’я, номер кімнати, дату народження(день, місяць, рік), дату заїзду (день, місяць, рік), дату виїзду (день, місяць, рік), тип кімнати. При виборі введення даних з файлу, дані буде зчитано з файлу «Input.txt» (рис. 7).

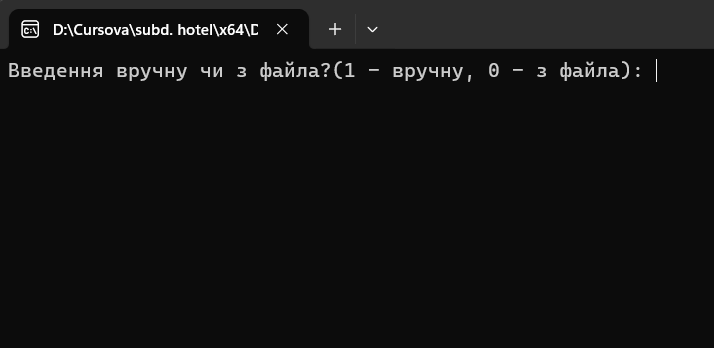


Рисунок 7

При виборі пункту «(2) Виведення даних», з’являється відповідні записані або зчитані з файлу дані.

При виборі пункту «(3) Змінити дані», з’являється повідомлення, де потрібно ввести відповідний номер кімнати, згідно якого треба внести зміни, якщо введеного номера кімнати не існує, з’явиться повідомлення, що такого гостя з номером кімнати не знайдено (рис. 8, 9).

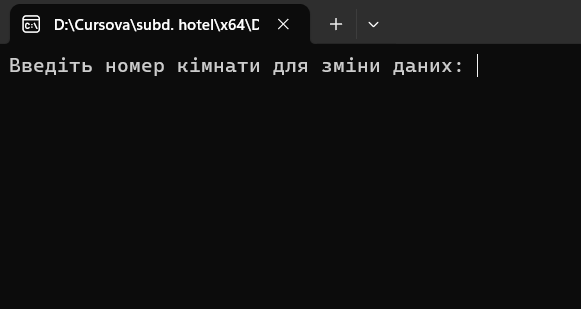


Рисунок 8

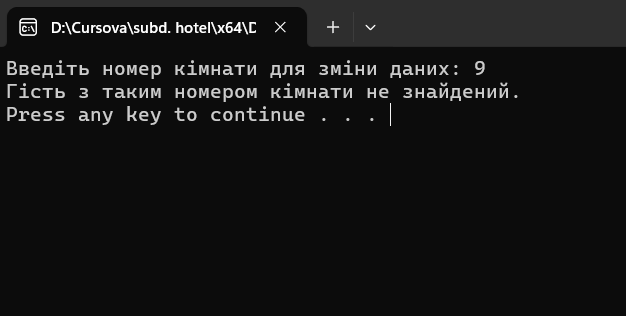


Рисунок 9

При виборі пункту «(4) Видалити дані», з’являється поле, де потрібно ввести номер кімнати, згідно якої потрібно видалити дані, якщо введеного номера кімнати не існує, з’явиться повідомлення, що гість з такою кімнатою не знайдений (рис. 10, 11).

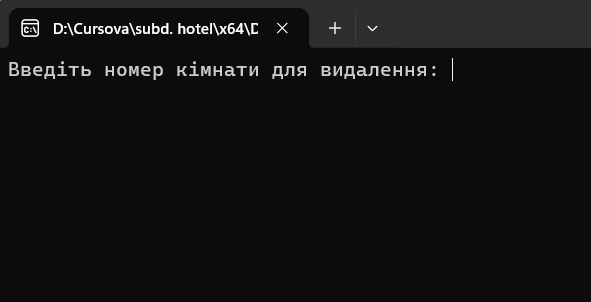


Рисунок 10

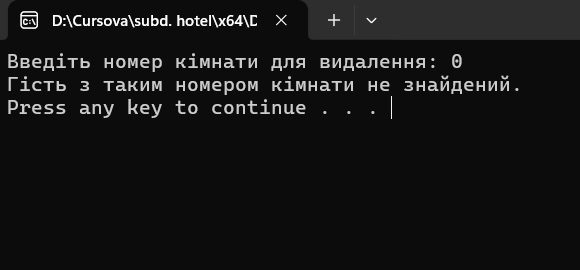


Рисунок 11

При виборі пункту «(5) Сортування даних», дані буде відсортовано за номером кімнати (рис. 12).

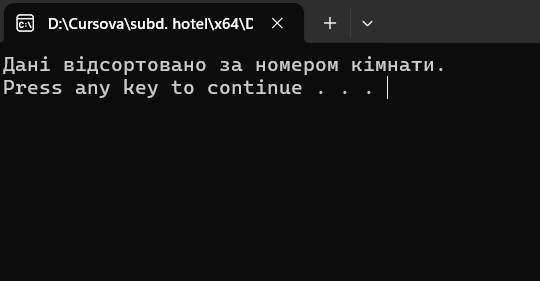


Рисунок 12

При виборі пункту «(6) Збереження даних», з’явиться вікно, в якому потрібно вказати файл, куди потрібно зберегти дані (Input.txt) (рис. 13).

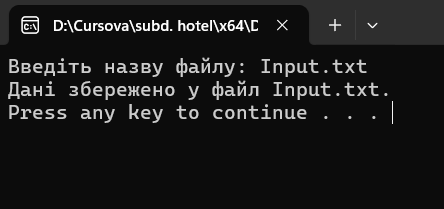


Рисунок 13

При виборі пункту «(7) Пошук за номером кімнати», з’явиться повідомлення, де потрібно ввести номер кімнати, який потрібно знайти (рис. 14).

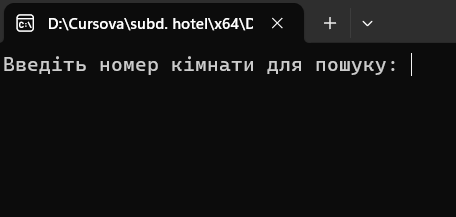


Рисунок 14

При виборі пункту «(8) Виведення номерів готелю та їх вартості», з’являється вікно з інформацією про номери в готелі (рис. 15).

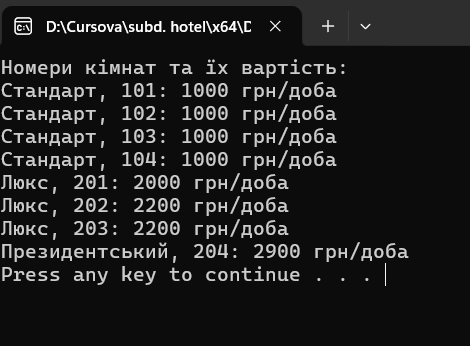


Рисунок 15

**ВИСНОВКИ**

У результаті виконання курсового проєкту розроблена система управління

базою даних «Готель» для комп’ютерів, які працюють під керуванням ОС Windows.

Розроблене програмне забезпечення має наступні можливості:

1. Можливість додавання, видалення, редагування записів.
2. Сортування та пошук серед записів в базі даних.
3. Перевірку коректності введення даних та обробку виключень.
4. Збереження даних у файл.
5. Зчитування даних з файлу.
6. Перегляд записів в базі даних.

Ця розробка може бути вдосконалена в майбутньому шляхом додавання нового функціоналу та зміни логіки обробки.

**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ**

1. <https://stackoverflow.com/questions/7352099/stdstring-to-char>

2. <https://www.udacity.com/blog/2021/05/how-to-read-from-a-file-in-cpp.html>

3. <https://www.udacity.com/blog/2021/05/how-to-read-from-a-file-in-cpp.html>

4. <https://www.w3schools.com/cpp/cpp_classes.asp>

5.<https://learn.saylor.org/course/view.php?id=65&gad_source=1&gclid=Cj0KCQjwpvK4BhDUARIsADHt9sQ4rB686zNBX8vkqr-AYZnbEyjHyfrv225Tk9Ik3H3OhibT9UB-WlcaAvxjEALw_wcB>

6. <https://www.w3schools.com/cpp/cpp_constructors.asp>

7.<https://learn.saylor.org/course/view.php?id=65&gad_source=1&gclid=Cj0KCQjwpvK4BhDUARIsADHt9sRjAQnBncyGiSyliVvbKFlmHG_RYAeWX5yIOZspbvRhvyaXP0TAs5saAgkMEALw_wcB>

8. <https://en.cppreference.com/w/cpp/container/vector>

**ДОДАТКИ**

**ДОДАТОК А. Скролінг (текст) програми**

**Програмний модуль *Hotel***

**Вміст «Hotel.h»:**

#ifndef HOTEL\_H

#define HOTEL\_H

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

#include <algorithm>

#include <stdexcept>

#include "Data.h"

#include "Initial.h"

#include "Date.h"

#include "Address.h"

using namespace std;

class Hotel {

public:

void DataEntry(vector<Data>& data);

void readDataFromFile(vector<Data>& data, const string& fileName);

void printData(const vector<Data>& data) const;

void modifyData(vector<Data>& data);

void deleteData(vector<Data>& data);

void sortData(vector<Data>& data);

void saveDataToFile(const vector<Data>& data, const string& fileName) const;

void searchRoom(int roomNumber, const std::vector<Data>& data) const;

void displayRooms() const;

};

#endif // HOTEL\_H

**Вміст «Hotel.cpp»:**

#include "Hotel.h"

#include <memory>

#include <string>

#include <cstdio>

#include <io.h>

#include <fcntl.h>

#include <sstream>

#include <fstream>

void Hotel::DataEntry(vector<Data>& data) {

std::string passportSeries, surname, name, patronymic;

int passportNumber, roomNumber;

int birthDay, birthMonth, birthYear;

int checkInDay, checkInMonth, checkInYear;

int checkOutDay, checkOutMonth, checkOutYear;

std::string roomType;

// Введення даних

std::cout << "Введіть серію паспорта: ";

std::cin >> passportSeries;

std::cout << "Введіть номер паспорта: ";

std::cin >> passportNumber;

std::cout << "Введіть прізвище гостя: ";

std::cin >> surname;

std::cout << "Введіть ім'я гостя: ";

std::cin >> name;

std::cout << "Введіть по батькові гостя: ";

std::cin >> patronymic;

// Об'єднання в повне ім'я

std::ostringstream fullName;

fullName << surname << " " << name << " " << patronymic;

std::string guestName = fullName.str();

std::cout << "Введіть номер кімнати: ";

std::cin >> roomNumber;

std::cout << "Введіть тип кімнати: ";

std::cin >> roomType;

std::cout << "Введіть дату народження (день, місяць, рік): ";

std::cin >> birthDay >> birthMonth >> birthYear;

std::cout << "Введіть дату заїзду (день, місяць, рік): ";

std::cin >> checkInDay >> checkInMonth >> checkInYear;

std::cout << "Введіть дату виїзду (день, місяць, рік): ";

std::cin >> checkOutDay >> checkOutMonth >> checkOutYear;

Initial initial(surname, name, patronymic);

Date birthDate(birthDay, birthMonth, birthYear);

Address address(roomType, roomNumber);

Date checkInDate(checkInDay, checkInMonth, checkInYear);

Date checkOutDate(checkOutDay, checkOutMonth, checkOutYear);

Data guest(passportSeries, passportNumber, std::move(initial),

std::move(birthDate), std::move(address),

std::move(checkInDate), std::move(checkOutDate),

roomNumber, guestName);

data.push\_back(guest);

// Збереження даних у файл Input.txt

std::ofstream outFile("Input.txt", std::ios::app);

if (outFile.is\_open()) {

outFile << "Серія паспорта: " << passportSeries << "\n"

<< "Номер паспорта: " << passportNumber << "\n"

<< "Прізвище: " << surname << "\n"

<< "Ім'я: " << name << "\n"

<< "По батькові: " << patronymic << "\n"

<< "Номер кімнати: " << roomNumber << "\n"

<< "Тип кімнати: " << roomType << "\n"

<< "Дата народження: " << birthDay << "/" << birthMonth << "/" << birthYear << "\n"

<< "Дата заїзду: " << checkInDay << "/" << checkInMonth << "/" << checkInYear << "\n"

<< "Дата виїзду: " << checkOutDay << "/" << checkOutMonth << "/" << checkOutYear << "\n"

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

outFile.close();

}

else {

std::cerr << "Не вдалося відкрити файл для запису." << std::endl;

}

std::cout << "Дані додано успішно." << std::endl;

}

// Читання даних з файлу

void Hotel::readDataFromFile(vector<Data>& data, const string& fileName) {

ifstream file(fileName);

if (!file) {

throw runtime\_error("Неможливо відкрити файл: " + fileName);

}

string passportSeries, name, surname, patronymic, typeroom, guestName;

int passportNumber, roomNumber;

int birthDay, birthMonth, birthYear;

int checkInDay, checkInMonth, checkInYear;

int checkOutDay, checkOutMonth, checkOutYear;

while (file >> passportSeries >> passportNumber >> surname >> name >> patronymic

>> birthDay >> birthMonth >> birthYear

>> typeroom >> roomNumber

>> checkInDay >> checkInMonth >> checkInYear

>> checkOutDay >> checkOutMonth >> checkOutYear) {

Initial guestInitial(surname, name, patronymic);

Date birthDate(birthDay, birthMonth, birthYear);

Address address(typeroom, roomNumber);

Date checkInDate(checkInDay, checkInMonth, checkInYear);

Date checkOutDate(checkOutDay, checkOutMonth, checkOutYear);

Data newGuest(

passportSeries,

passportNumber,

std::move(guestInitial),

std::move(birthDate),

std::move(address),

std::move(checkInDate),

std::move(checkOutDate),

roomNumber,

guestName

);

data.push\_back(std::move(newGuest));

}

file.close();

cout << "Дані зчитано з файлу " << fileName << "." << endl;

}

// Виведення даних

void Hotel::printData(const vector<Data>& data) const {

for (const auto& guest : data) {

cout << "Серія паспорту: " << guest.getPassportSeries() << endl;

cout << "Номер паспорта: " << guest.getPassportNumber() << endl;

cout << "ПІБ гостя: " << guest.getInitial()->getSurname() << " "

<< guest.getInitial()->getName() << " "

<< guest.getInitial()->getPatronymic() << endl;

cout << "Номер кімнати: " << guest.getRoomNumber() << endl;

cout << "Дата народження: " << guest.getBirthDate()->getDay() << "/"

<< guest.getBirthDate()->getMonth() << "/"

<< guest.getBirthDate()->getYear() << endl;

cout << "Дата в'їзду: " << guest.getCheckInDate()->getDay() << "/"

<< guest.getCheckInDate()->getMonth() << "/"

<< guest.getCheckInDate()->getYear() << endl;

cout << "Дата виїзду: " << guest.getCheckOutDate()->getDay() << "/"

<< guest.getCheckOutDate()->getMonth() << "/"

<< guest.getCheckOutDate()->getYear() << endl;

cout << "Адреса: " << guest.getAddress()->getRoomType() << " "

<< guest.getRoomNumber() << endl;

}

}

// Змінити дані

void Hotel::modifyData(vector<Data>& data) {

int roomNumber;

cout << "Введіть номер кімнати для зміни даних: ";

cin >> roomNumber;

for (auto& guest : data) {

if (guest.getRoomNumber() == roomNumber) {

string newGuestName;

cout << "Введіть нове ім'я гостя: ";

cin >> newGuestName;

guest.setGuestName(newGuestName);

cout << "Дані оновлено." << endl;

return;

}

}

cout << "Гість з таким номером кімнати не знайдений." << endl;

}

// Видалення даних

void Hotel::deleteData(vector<Data>& data) {

int roomNumber;

cout << "Введіть номер кімнати для видалення: ";

cin >> roomNumber;

auto it = remove\_if(data.begin(), data.end(), [roomNumber](const Data& guest) {

return guest.getRoomNumber() == roomNumber;

});

if (it != data.end()) {

data.erase(it, data.end());

cout << "Гостя видалено." << endl;

}

else {

cout << "Гість з таким номером кімнати не знайдений." << endl;

}

}

// Сортування даних за номером кімнати

void Hotel::sortData(vector<Data>& data) {

sort(data.begin(), data.end(), [](const Data& a, const Data& b) {

return a.getRoomNumber() < b.getRoomNumber();

});

cout << "Дані відсортовано за номером кімнати." << endl;

}

// Збереження даних у файл

void Hotel::saveDataToFile(const vector<Data>& data, const string& fileName) const {

ofstream file(fileName);

if (!file) {

throw runtime\_error("Неможливо відкрити файл для запису: " + fileName);

}

for (const auto& guest : data) {

file << guest.getPassportSeries() << " " << guest.getPassportNumber() << endl;

file << guest.getInitial()->getSurname() << " "

<< guest.getInitial()->getName() << " "

<< guest.getInitial()->getPatronymic() << endl;

file << guest.getBirthDate()->getDay() << " "

<< guest.getBirthDate()->getMonth() << " "

<< guest.getBirthDate()->getYear() << endl;

file << guest.getAddress()->getRoomType() << " "

<< guest.getRoomNumber() << endl;

file << guest.getCheckInDate()->getDay() << " "

<< guest.getCheckInDate()->getMonth() << " "

<< guest.getCheckInDate()->getYear() << endl;

file << guest.getCheckOutDate()->getDay() << " "

<< guest.getCheckOutDate()->getMonth() << " "

<< guest.getCheckOutDate()->getYear() << endl;

file << guest.getGuestName() << endl;

file << "-------------------------" << endl;

}

file.close();

cout << "Дані збережено у файл " << fileName << "." << endl;

}

// Метод для пошуку за номером кімнати

void Hotel::searchRoom(int roomNumber, const std::vector<Data>& data) const

{

bool found = false;

for (const auto& record : data) {

if (record.getRoomNumber() == roomNumber) {

found = true;

std::cout << "Дані для номера кімнати " << roomNumber << ":\n";

std::cout << "Паспортна серія: " << record.getPassportSeries() << "\n";

std::cout << "Номер паспорта: " << record.getPassportNumber() << "\n";

std::cout << "Ім'я гостя: " << record.getFullName() << "\n";

std::cout << "Дата народження: " << record.getBirthDate()->toString() << "\n"; // метод toString для виводу дати

std::cout << "Адреса: " << record.getAddress()->toString() << "\n"; // метод toString для виводу адреси

std::cout << "Дата заїзду: " << record.getCheckInDate()->toString() << "\n";

std::cout << "Дата виїзду: " << record.getCheckOutDate()->toString() << "\n";

break;

}

}

if (!found) {

std::cout << "Номер кімнати " << roomNumber << " не знайдено.\n";

}

}

// Виведення номерів кімнат та їх вартості

void Hotel::displayRooms() const {

cout << "Номери кімнат та їх вартість:" << endl;

cout << "Стандарт, 101: 1000 грн/доба" << endl;

cout << "Стандарт, 102: 1000 грн/доба" << endl;

cout << "Стандарт, 103: 1000 грн/доба" << endl;

cout << "Стандарт, 104: 1000 грн/доба" << endl;

cout << "Люкс, 201: 2000 грн/доба" << endl;

cout << "Люкс, 202: 2200 грн/доба" << endl;

cout << "Люкс, 203: 2200 грн/доба" << endl;

cout << "Президентський, 204: 2900 грн/доба" << endl;

}

**Програмний модуль *Data***

**Вміст «Data.h»:**

#pragma once

#include <memory>

#include <string>

#include "Initial.h"

#include "Date.h"

#include "Address.h"

using namespace std;

class Data {

private:

unique\_ptr<Initial> initial;

string passportSeries;

int passportNumber;

unique\_ptr<Date> birthDate;

unique\_ptr<Address> address;

unique\_ptr<Date> checkInDate;

unique\_ptr<Date> checkOutDate;

int roomNumber;

string guestName;

public:

// Конструктор за замовчуванням

Data();

// Конструктор з параметрами

Data(const string& \_passportSeries, int \_passportNumber,

Initial&& \_initial, Date&& \_birthDate,

Address&& \_address, Date&& \_checkInDate,

Date&& \_checkOutDate, int \_roomNumber,

const string& \_guestName);

// Конструктор копіювання

Data(const Data& other);

// Оператор присвоєння копіювання

Data& operator=(const Data& other);

// Конструктор переміщення

Data(Data&& other) noexcept;

// Оператор присвоєння переміщення

Data& operator=(Data&& other) noexcept;

// Геттери

Initial\* getInitial() const;

Date\* getBirthDate() const;

Address\* getAddress() const;

Date\* getCheckInDate() const;

Date\* getCheckOutDate() const;

string getPassportSeries() const;

int getPassportNumber() const;

int getRoomNumber() const;

string getGuestName() const;

string getSurname() const {

return initial ? initial->getSurname() : "";

}

string getName() const {

return initial ? initial->getName() : "";

}

string getPatronymic() const {

return initial ? initial->getPatronymic() : "";

}

string getFullName() const {

return getSurname() + " " + getName() + " " + getPatronymic();

}

// Сеттери

void setInitial(const Initial& \_initial);

void setBirthDate(const Date& \_birthDate);

void setAddress(const Address& \_address);

void setCheckInDate(const Date& \_checkInDate);

void setCheckOutDate(const Date& \_checkOutDate);

void setPassportSeries(const string& \_passportSeries);

void setPassportNumber(int \_passportNumber);

void setRoomNumber(int \_roomNumber);

void setGuestName(const string& \_guestName);

};

**Вміст «Data.cpp»:**

#include "Data.h"

using namespace std;

// Конструктор за замовчуванням

Data::Data()

: passportNumber(0), roomNumber(0) {}

// Конструктор з параметрами

Data::Data(const string& \_passportSeries, int \_passportNumber,

Initial&& \_surname, Date&& \_birthDate,

Address&& \_address, Date&& \_checkInDate,

Date&& \_checkOutDate, int \_roomNumber,

const string& \_guestName)

: passportSeries(\_passportSeries),

passportNumber(\_passportNumber),

initial(make\_unique<Initial>(std::move(\_surname))),

birthDate(make\_unique<Date>(std::move(\_birthDate))),

address(make\_unique<Address>(std::move(\_address))),

checkInDate(make\_unique<Date>(std::move(\_checkInDate))),

checkOutDate(make\_unique<Date>(std::move(\_checkOutDate))),

roomNumber(\_roomNumber){}

// Конструктор копіювання

Data::Data(const Data& other)

: passportSeries(other.passportSeries),

passportNumber(other.passportNumber),

initial(make\_unique<Initial>(\*other.initial)),

birthDate(make\_unique<Date>(\*other.birthDate)),

address(make\_unique<Address>(\*other.address)),

checkInDate(make\_unique<Date>(\*other.checkInDate)),

checkOutDate(make\_unique<Date>(\*other.checkOutDate)),

roomNumber(other.roomNumber),

guestName(other.guestName) {}

// Оператор присвоєння копіювання

Data& Data::operator=(const Data& other) {

if (this == &other) return \*this;

passportSeries = other.passportSeries;

passportNumber = other.passportNumber;

initial = make\_unique<Initial>(\*other.initial);

birthDate = make\_unique<Date>(\*other.birthDate);

address = make\_unique<Address>(\*other.address);

checkInDate = make\_unique<Date>(\*other.checkInDate);

checkOutDate = make\_unique<Date>(\*other.checkOutDate);

roomNumber = other.roomNumber;

guestName = other.guestName;

return \*this;

}

// Конструктор переміщення

Data::Data(Data&& other) noexcept

: passportSeries(std::move(other.passportSeries)),

passportNumber(other.passportNumber),

initial(std::move(other.initial)),

birthDate(std::move(other.birthDate)),

address(std::move(other.address)),

checkInDate(std::move(other.checkInDate)),

checkOutDate(std::move(other.checkOutDate)),

roomNumber(other.roomNumber),

guestName(std::move(other.guestName)) {}

// Оператор присвоєння переміщення

Data& Data::operator=(Data&& other) noexcept {

if (this == &other) return \*this;

passportSeries = std::move(other.passportSeries);

passportNumber = other.passportNumber;

initial = std::move(other.initial);

birthDate = std::move(other.birthDate);

address = std::move(other.address);

checkInDate = std::move(other.checkInDate);

checkOutDate = std::move(other.checkOutDate);

roomNumber = other.roomNumber;

guestName = std::move(other.guestName);

return \*this;

}

// Геттери

Initial\* Data::getInitial() const { return initial.get(); }

Date\* Data::getBirthDate() const { return birthDate.get(); }

Address\* Data::getAddress() const { return address.get(); }

Date\* Data::getCheckInDate() const { return checkInDate.get(); }

Date\* Data::getCheckOutDate() const { return checkOutDate.get(); }

string Data::getPassportSeries() const { return passportSeries; }

int Data::getPassportNumber() const { return passportNumber; }

int Data::getRoomNumber() const { return roomNumber; }

string Data::getGuestName() const { return guestName; }

// Сеттери

void Data::setInitial(const Initial& \_initial) { initial = make\_unique<Initial>(\_initial); }

void Data::setBirthDate(const Date& \_birthDate) { birthDate = make\_unique<Date>(\_birthDate); }

void Data::setAddress(const Address& \_address) { address = make\_unique<Address>(\_address); }

void Data::setCheckInDate(const Date& \_checkInDate) { checkInDate = make\_unique<Date>(\_checkInDate); }

void Data::setCheckOutDate(const Date& \_checkOutDate) { checkOutDate = make\_unique<Date>(\_checkOutDate); }

void Data::setPassportSeries(const string& \_passportSeries) { passportSeries = \_passportSeries; }

void Data::setPassportNumber(int \_passportNumber) { passportNumber = \_passportNumber; }

void Data::setRoomNumber(int \_roomNumber) { roomNumber = \_roomNumber; }

void Data::setGuestName(const string& \_guestName) { guestName = \_guestName; }

**Програмний модуль *Initial***

**Вміст «Initial.h»:**

#pragma once

#include <string>

#ifndef INITIAL\_H

#define INITIAL\_H

using namespace std;

class Initial {

private:

string surname;

string patronymic;

string name;

public:

// Конструктор за замовчуванням

Initial();

// Конструктор з параметрами

Initial(const string& \_surname, const string& \_patronymic, const string& \_name);

// Конструктор копіювання

Initial(const Initial& other);

// Конструктор переміщення

Initial(Initial&& other) noexcept;

// Геттери

string getSurname() const;

string getPatronymic() const;

string getName() const;

// Сеттери

void setSurname(const string& \_surname);

void setPatronymic(const string& \_patronymic);

void setName(const string& \_name);

// Оператор присвоєння

Initial& operator=(const Initial& other);

};

#endif // INITIAL\_H

**Вміст «Initial.cpp»:**

#include "Initial.h"

#include <utility>

// Конструктор за замовчуванням

Initial::Initial() : surname(""), patronymic(""), name("") {}

// Конструктор з параметрами

Initial::Initial(const string& \_surname, const string& \_patronymic, const string& \_name)

: surname(\_surname), patronymic(\_patronymic), name(\_name) {}

// Конструктор копіювання

Initial::Initial(const Initial& other)

: surname(other.surname), patronymic(other.patronymic), name(other.name) {}

// Конструктор переміщення

Initial::Initial(Initial&& other) noexcept

: surname(move(other.surname)), patronymic(move(other.patronymic)), name(move(other.name)) {}

// Геттери

string Initial::getSurname() const { return surname; }

string Initial::getPatronymic() const { return patronymic; }

string Initial::getName() const { return name; }

// Сеттери

void Initial::setSurname(const string& \_surname) { surname = \_surname; }

void Initial::setPatronymic(const string& \_patronymic) { patronymic = \_patronymic; }

void Initial::setName(const string& \_name) { name = \_name; }

// Оператор присвоєння

Initial& Initial::operator=(const Initial& other) {

if (this == &other) return \*this; // перевірка на самоприсвоєння

surname = other.surname;

patronymic = other.patronymic;

name = other.name;

return \*this;

}

**Програмний модуль *Address***

**Вміст «Address.h»:**

#pragma once

#include <string>

class Address {

private:

std::string roomType;

int roomNumber;

public:

// Конструктори

Address();

Address(const std::string& \_roomType, int \_roomNumber);

Address(const Address& other);

Address(Address&& other) noexcept;

// Деструктор

~Address();

// Геттери

std::string getRoomType() const;

int getRoomNumber() const;

// Сеттери

void setRoomType(const std::string& \_roomType);

void setRoomNumber(int \_roomNumber);

// Оператор присвоєння

Address& operator=(const Address& other);

// Метод для перетворення адреси в рядок

std::string toString() const;

};

**Вміст «Address.cpp»:**

#include "Address.h"

#include <utility>

#include <sstream>

// Конструктор за замовчуванням

Address::Address() : roomType("Unknown"), roomNumber(0) {}

// Конструктор з параметрами

Address::Address(const std::string& \_roomType, int \_roomNumber)

: roomType(\_roomType), roomNumber(\_roomNumber) {}

// Конструктор копіювання

Address::Address(const Address& other)

: roomType(other.roomType), roomNumber(other.roomNumber) {}

// Конструктор переміщення

Address::Address(Address&& other) noexcept

: roomType(std::move(other.roomType)), roomNumber(other.roomNumber) {

other.roomNumber = 0;

}

// Деструктор

Address::~Address() = default;

// Геттери

std::string Address::getRoomType() const { return roomType; }

int Address::getRoomNumber() const { return roomNumber; }

// Сеттери

void Address::setRoomType(const std::string& \_roomType) { roomType = \_roomType; }

void Address::setRoomNumber(int \_roomNumber) { roomNumber = \_roomNumber; }

// Оператор присвоєння

Address& Address::operator=(const Address& other) {

if (this != &other) {

roomType = other.roomType;

roomNumber = other.roomNumber;

}

return \*this;

}

// Метод для перетворення адреси в рядок

std::string Address::toString() const {

std::ostringstream oss;

oss << "Тип кімнати: " << roomType << ", Номер кімнати: " << roomNumber;

return oss.str();

}

**Програмний модуль *Date***

**Вміст «Date.h»:**

#pragma once

#include <string>

class Date {

private:

int day;

int month;

int year;

public:

// Конструктори

Date();

Date(int \_day, int \_month, int \_year);

Date(const Date& other);

Date(Date&& other) noexcept;

// Геттери

int getDay() const;

int getMonth() const;

int getYear() const;

// Сеттери

void setDay(int \_day);

void setMonth(int \_month);

void setYear(int \_year);

// Оператор присвоєння

Date& operator=(const Date& other);

// Метод для перетворення дати в рядок

std::string toString() const;

};

**Вміст «Date.cpp»:**

#include "Date.h"

#include <iomanip>

#include <sstream>

// Конструктор за замовчуванням

Date::Date() : day(1), month(1), year(2000) {}

// Конструктор з параметрами

Date::Date(int \_day, int \_month, int \_year) : day(\_day), month(\_month), year(\_year) {}

// Конструктор копіювання

Date::Date(const Date& other) : day(other.day), month(other.month), year(other.year) {}

// Конструктор переміщення

Date::Date(Date&& other) noexcept : day(other.day), month(other.month), year(other.year) {

other.day = 0;

other.month = 0;

other.year = 0;

}

// Геттери

int Date::getDay() const { return day; }

int Date::getMonth() const { return month; }

int Date::getYear() const { return year; }

// Сеттери

void Date::setDay(int \_day) { day = \_day; }

void Date::setMonth(int \_month) { month = \_month; }

void Date::setYear(int \_year) { year = \_year; }

// Оператор присвоєння

Date& Date::operator=(const Date& other) {

if (this != &other) {

day = other.day;

month = other.month;

year = other.year;

}

return \*this;

}

// Метод для перетворення дати в рядок

std::string Date::toString() const {

std::ostringstream oss;

oss << std::setw(2) << std::setfill('0') << day << "/"

<< std::setw(2) << std::setfill('0') << month << "/"

<< year;

return oss.str();

}

**Програмний модуль *subd.hotel***

**Вміст «subd.hotel.cpp»:**

#include <vector>

#include <iostream>

#include <map>

#include <algorithm>

#include "Hotel.h"

#include <Windows.h>

#include <direct.h>

using namespace std;

int \_stateMenu;

Hotel hotel;

void Menu() {

cout << "Виберіть дію: " << endl

<< "(0) Вихід з програми" << endl

<< "(1) Введення даних" << endl

<< "(2) Вивід даних" << endl

<< "(3) Змінити дані" << endl

<< "(4) Видалити дані" << endl

<< "(5) Сортування даних" << endl

<< "(6) Збереження даних" << endl

<< "(7) Пошук за номером кімнати" << endl

<< "(8) Виведення номерів готелю та їх вартості" << endl

<< "(9) Посібник користувача" << endl

<< "Ваш вибір: ";

cin >> \_stateMenu;

}

int main() {

try {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

Menu();

int \_actions;

string fileName;

vector<Data> data;

while (\_stateMenu != 0) {

try {

switch (\_stateMenu) {

case 1:

system("cls"); // очистка консолі

std::cout << "Введення вручну чи з файла? (1 - вручну, 0 - з файла): ";

std::cin >> \_actions;

system("cls");

if (\_actions == 1) {

hotel.DataEntry(data);

}

else {

hotel.readDataFromFile(data, "Input.txt");

if (data.empty()) {

std::cerr << "Файл не містить даних або вони зчитані некоректно." << std::endl;

}

}

system("pause");

system("cls");

Menu();

break;

case 2:

system("cls");

if (!data.empty()) {

hotel.printData(data);

}

else {

cout << "Дані пусті!" << endl;

}

system("pause");

system("cls");

Menu();

break;

case 3:

system("cls");

if (!data.empty()) {

hotel.modifyData(data);

}

else {

cout << "Дані пусті!" << endl;

}

system("pause");

system("cls");

Menu();

break;

case 4:

system("cls");

if (!data.empty()) {

hotel.deleteData(data);

}

else {

cout << "Дані пусті!" << endl;

}

system("pause");

system("cls");

Menu();

break;

case 5:

system("cls");

if (!data.empty()) {

hotel.sortData(data);

}

else {

cout << "Дані пусті!" << endl;

}

system("pause");

system("cls");

Menu();

break;

case 6:

system("cls");

cout << "Введіть назву файлу: ";

cin >> fileName;

if (!data.empty()) {

hotel.saveDataToFile(data, fileName);

}

else {

cout << "Дані пусті!" << endl;

}

system("pause");

system("cls");

Menu();

break;

case 7:

system("cls");

int roomNumber;

cout << "Введіть номер кімнати для пошуку: ";

cin >> roomNumber;

hotel.searchRoom(roomNumber, data);

system("pause");

system("cls");

Menu();

break;

case 8:

system("cls");

hotel.displayRooms();

system("pause");

system("cls");

Menu();

break;

case 9:

system("cls");

cout << " При запуску програми відкривається консольне меню, яке містить 10 пунктів. Натискаючи на 0 програма завершить свою роботу та закриється. Для введення даних натисніть на 1, після цієї дії можна обрати варіант введення даних — вручну (1) або з файлу (0). Після введення даних знову відкриється меню. Для перегляду даних натисніть на 2. Для зміни (редагування даних) натисніть на 3. Далі потрібно ввести номер кімнати для якої потрібно змінити дані. Після вводу кімнати можна змінити ім'я клієнта який проживає в номері. Для видалення даних спочатку потрібно вибрати 4 пункт, та вписати номер кімнати дані якої потрібно видалити. Для сортування даних треба спочатку обрати 5 пункт. Дані буде відсортовано за номером кімнат.

Для збереження даних у файл оберіть пункт 6. Для пошуку оберіть пункт 7, після введіть номер кімнати, про яку хочете дізнатись більше інформації. Для виведення номерів в готелі та їх вартості оберіть дію номер 8." << endl;

system("pause");

system("cls");

Menu();

break;

default:

cout << "Невірно введений номер дії!" << endl;

system("pause");

system("cls");

Menu();

break;

}

}

catch (const exception& e) {

cerr << "Виникла помилка: " << e.what() << endl;

system("pause");

system("cls");

Menu();

}

}

}

catch (const exception& e) {

cerr << "Глобальна помилка програми: " << e.what() << endl;

}

return 0;

}