МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

**Національний університет «Запорізька політехніка»**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Кафедра програмних засобів\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(найменування кафедри)

**КУРСОВИЙ ПРОЕКТ**

з дисципліни «Об’єктно-орієнтоване програмування»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(назва дисципліни)

на тему:\_\_\_\_ «Чат-бот для підтримки міжнародної академічної мобільності»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студентів 2 курсу групи \_\_\_\_\_\_\_\_\_ КНТ-229\_\_\_\_\_\_\_\_

спеціальності Комп'ютерні науки

напряму підготовки: Комп'ютерні науки\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Сепетий О. В.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Колбаса Д.\_.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Ващенко М. .\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Момот В. .\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 (прізвище та ініціали)

Керівник Доцент к.н.т., Каплієнко Т. І.

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала

Кількість балів: \_\_\_\_\_\_\_ Оцінка: ECTS

Члени комісії Каплієнко Т. І.

(підпис) (прізвище та ініціали)

Миронова Н. О.

(підпис) (прізвище та ініціали)

Табунщик Г. В.

(підпис) (прізвище та ініціали)

м. Запоріжжя

2020 рік

**ЗАВДАННЯ  
на курсовий проект студентів**

Сепетого О.В., Колбаси Д. ., Ващенко М. ., Момота В. .

1. Тема проекту: Чат-бот для підтримки міжнародної академічної мобільності

2. Термін здачі студентами закінченого проекту: 1 грудня 2020

3. Вихідні дані до проекту: Створити розклад для закладу вищої освіти

дані: інформація про організацію навчального процесу, кафедри, викладачів, аудиторії та навчальний план для кафедри КНТ;

вихідні дані: створений розклад для заданих потоків.

4. Зміст розрахунково-пояснювальної записки:

1. Аналіз предметної області;

2. Аналіз програмних засобів;

3. Основні рішення з реалізації компонентів системи;

4. Керівництво програміста;

5. Керівництво корисутвача;

Висновки

Додаток А. Текст програми

Додаток Б. Інтерфейс програми

Додаток В. Слайди презентації

5. Дата видачі завдання: 11 вересня 2020 року

Календарний план

|  |  |  |  |
| --- | --- | --- | --- |
| № | Назва етапів курсового  проекту (роботи) | Термін виконання етапів  проекту (роботи) | Примітка |
| 1. | Аналіз індивідуального завдання. | 1-2 тиждень |  |
| 2. | Аналіз програмних засобів, що будуть використовуватись в роботі. | 3-4 тиждень |  |
| 3. | Аналіз структур даних, що необхідно використати в курсової роботі. | 4-5 тиждень |  |
| 4. | Вивчення можливостей програмної реалізації структур даних та інтерфейсу користувача. | 5-9 тиждень |  |
| 5. | Аналіз вимог до апаратних засобів | 9 тиждень |  |
| 6. | Розробка програмного забезпечення | 9-15 тиждень |  |
| 7. | Оформлення, відповідних пунктів пояснювальної записки. | 10-16 тиждень | Розділи 1-5 ПЗ |
| 8. | Захист курсової роботи. | 17 тиждень |  |

Студент Терлецький С. В.

Студент Терлецький С. В.

Студент Ліпіхін С. О.

Студент Денисенко С. В.

Керівник Каплієнко Т. І.

«\_\_\_\_\_\_\_» 2020 р.

реферат

Мета роботи – створення програмного забезпечення для автоматизованого складання розкладу для закладів вищої освіти.

Проведено аналіз предметної області, досліджено аналогічне програмне забезпечення, а також існуючі методи та програмні засоби для вирішення завдання.

Для реалізації програмного продукту використовувалася мова програмування С# та середа розробки Visual Studio.

Здійснено опис прийнятих рішень, реалізованих класів, наведено опис полів та методів реалізованих класів. Також у роботі були використані контейнери List для збереження вхідних даних та Dictionary для збереження розкладу.

У даній роботі було проведено дослідження процесу складання розкладу у реаліях сьогоденних вищих навчальних закладів, розглянуто особливості мови програмування С# у середовищі розробки Visual Studio 2019. За основу для розробки графіного інтерфейсу для користувача була узята мова розмітки сторінки XAML, що використовувалася на базі графічної системи WPF. Потреби у збереженні, виведенні, конвертації та роботі з даними реалізовані у роботі із реляційною базою даних на основі SQLiteStudio.

C#, DICTIONARY, LIST, SQLITESTUDIO, WPF, XAML, ЕВРИСТИЧНІ АЛГОРИТМИ, РОЗКЛАД.

зміст

[Календарний план 2](#_Toc27629364)

[реферат 3](#_Toc27629365)

[зміст 4](#_Toc27629366)

[перелік умовних позначень, символів, одиниць, скорочень і термінів 7](#_Toc27629367)

[вступ 8](#_Toc27629368)

[1 Аналіз предметної області 9](#_Toc27629369)

[1.1 Аналіз начального процесу, як основи предметної області 9](#_Toc27629370)

[1.2 Огляд існуючих методів вирішення завдання 9](#_Toc27629371)

[1.2 Огляд існуючих програм та сервісів для пошуку найкоротшого шляху 10](#_Toc27629372)

[1.2.1 Передмова 10](#_Toc27629373)

[1.2.2 Система «Розклад ПРО» 11](#_Toc27629374)

[1.2.3 Университет 3.2.0.711 12](#_Toc27629375)

[1.2.4 «Програма складання розкладу занять у ВНЗ 2001» 13](#_Toc27629376)

[2 Аналіз програмних засобів 15](#_Toc27629377)

[2.1 Огляд особливостей мови програмування 15](#_Toc27629378)

[2.2 Огляд особливостей обраного компілятору 16](#_Toc27629379)

[2.3 Огляд класової ієрархії WPF 16](#_Toc27629380)

[2.4 Висновки з розділу 18](#_Toc27629381)

[3 основні рішення з реалізації компонентів системи 19](#_Toc27629382)

[3.1 Основні рішення щодо розроблених класів 19](#_Toc27629383)

[3.1.1 Клас LoadFiles 20](#_Toc27629384)

[3.1.2 Клас ExportFiles 22](#_Toc27629385)

[3.1.3 Клас Flows 23](#_Toc27629386)

[3.1.4 Клас Auditorium 25](#_Toc27629387)

[3.1.5 Клас Professor 27](#_Toc27629388)

[3.1.6 Клас ExpectedSchedule 28](#_Toc27629389)

[3.1.7 Клас Group 30](#_Toc27629390)

[3.1.8 Клас Curriculum 30](#_Toc27629391)

[3.1.9 Клас SchoolWeek 31](#_Toc27629392)

[3.1.10 Клас Day 32](#_Toc27629393)

[3.1.11 Клас Lesson 33](#_Toc27629394)

[3.1.12 Клас LessonForGroup 34](#_Toc27629395)

[3.1.13 Клас LessonForOneFlow 35](#_Toc27629396)

[3.1.14 Клас LessonForFlows 35](#_Toc27629397)

[3.1.15 Клас Professorss\_for\_combox 36](#_Toc27629398)

[3.1.16 Клас Item 37](#_Toc27629399)

[3.1.17 Клас MainWindow 37](#_Toc27629400)

[3.2 Основні розроблені алгоритми 39](#_Toc27629401)

[3.2.1 Аналіз головного алгоритму 39](#_Toc27629402)

[3.2.2 Алгоритм лінійного пошуку 41](#_Toc27629403)

[3.2.3 Алгоритм сортування вставками 43](#_Toc27629404)

[3.2.4 Алгоритми з використанням динамічного програмування 46](#_Toc27629405)

[3.2.5 Перший етап. Формування списку заннять 47](#_Toc27629406)

[3.2.6 Другий етап. Пошук часу проведення заняття 49](#_Toc27629407)

[3.3 Основні рішення щодо розробки інтерфейсу 51](#_Toc27629408)

[3.3.1 Вибір платформи. 51](#_Toc27629409)

[3.3.2 Основні рішення з впорядкування елементів інтерфейсу 52](#_Toc27629410)

[3.3.3 Основні рішення з комунікації з користувачем 52](#_Toc27629411)

[3.3.4 Основні рішення з створенням інтерфейсу 53](#_Toc27629412)

[3.4 Основні рішення щодо розробки бази даних 55](#_Toc27629413)

[3.4.1 Основні рішення щодо вибору бази даних 55](#_Toc27629414)

[3.4.2 Структура збереження даних у БД 56](#_Toc27629415)

[3.4.2 Основні запити SQL, що було використано 59](#_Toc27629416)

[4 керівництво програміста 61](#_Toc27629417)

[4.1 Призначення та умови застосування програми 61](#_Toc27629418)

[4.1.1 Призначення програми 61](#_Toc27629419)

[4.1.2 Функції програми 61](#_Toc27629420)

[4.1.3 Умови застосування програми 61](#_Toc27629421)

[4.2 Характеристика програми 62](#_Toc27629422)

[4.2.1 Структура програми 62](#_Toc27629423)

[4.2.2 Алгоритм роботи програми 63](#_Toc27629424)

[4.2.3 Засоби контролю правильності виконання та самовідновлення програми 63](#_Toc27629425)

[4.3 Звертання до програми 64](#_Toc27629426)

[4.4 Вхідні та вихідні дані 64](#_Toc27629427)

[4.4.1 Вхідні дані 64](#_Toc27629428)

[4.4.2 Вихідні дані 64](#_Toc27629429)

[4.5 Повідомлення 64](#_Toc27629430)

[5 керівництво користувача 66](#_Toc27629431)

[5.1 Призначення програми 66](#_Toc27629432)

[5.2 Умови виконання програми 66](#_Toc27629433)

[5.2.1 Апаратні вимоги програми 66](#_Toc27629434)

[5.2.2 Вимоги до користувача 66](#_Toc27629435)

[5.3 Виконання програми 67](#_Toc27629436)

[5.3.1 Запуск програми 67](#_Toc27629437)

[5.3.2 Виконання роботи з програмою 68](#_Toc27629438)

[5.4 Повідомлення користувачу 68](#_Toc27629439)

[5.5 Довідка програми 69](#_Toc27629440)

[Висновки 70](#_Toc27629441)

[ПЕРЕЛІК ПОСИЛАНЬ 71](#_Toc27629442)

[Додаток А Діаграма класів 72](#_Toc27629443)

[Додаток Б Код Програми 73](#_Toc27629444)

[Додаток В інтерфейс програми 74](#_Toc27629445)

перелік умовних позначень, символів, одиниць, скорочень і термінів

БД – база даних.

Запит – запит в базу даних.

ООП - об’єктно-орієнтоване програмування.

СУБД – система управління базами даних.

Dictionary – структура, що зберігає об'єкти, які представляють пару ключ-значення.

List – структура з простору імен System.Collections.Generic, що представляє найпростіший список однотипних об'єктів.

SQL – Structured Query Language.

WPF – Windows Presentation Foundation.

XAML – eXtensible Application Markup Language.

Е.г.і. - елементи графічного інтерфейсу

АМ – академічна мобільність

вступ

Завдання складання розкладу є предметом широких наукових досліджень та дискусій ще з середини XX сторіччя, але навіть у реаліях сучасного технічного прогресу ця тема не втрачає своєї насущності та проблематики.

Отже, актуальність даної курсової роботи полягає у створенні програмного забезпечення, що має на меті автоматизацію складання розкладу для закладів вищої освіти.

Область використання такого застосунку включає в себе різні сфери людської діяльності, такі як: промисловість, транспортні перевезення, обслуговування, банківські системи, а насамперед – освіта.

Наразі, сучасна освітня система вимагає оптимізації процесу навчання та використання більш прогресивних методів вирішення проблеми складання розкладу у зв’язку з постійним збільшенням обсягів вхідної інформації, що, як наслідок, приводить до дефіциту часу та робочої сили через інвестування ресурсів у таку рутинну справу. Інновації у цій сфері вивільнять необхідні матеріали для підвищення актуалізації, розширення і популяризації вищої освіти. Тому автоматизація створення навчального розкладу є особливо гострою для ефективного не тільки планування, а й ведення освітнього процесу.

Отже, мета даного дослідження, а саме складання розкладу, є не тільки актуальною, а й необхідною для сьогодення задачею.

1 Аналіз предметної області

В даному розділі розглянуті існуючі методи вирішення завдання складання розкладу та програми, що реалізують дану потребу, а саме створення розкладу для вищого навчального закладу.

1.1 Аналіз начального процесу, як основи предметної області

Проблема створення розкладу активно вирішується у багатьох вищих навчальних закладах у сфері освіти, тому предметна область роботи - це організація навчального процесу. Отже, основні вимоги до навчального процесу наступні:

* …..

1.2 Огляд існуючих методів вирішення завдання

Планування розкладу навчальних занять - це завдання на складання розкладу комбінаторного типу, тобто знаходження усіх можливих варіантів, які відповідають вимогам, та вибір з них найбільш оптимального.

Є ряд евристичних і переборних методів, які виконують пошук розкладу, який відповідає початковим умовам. Один з таких методів був створений у результаті виконання завдання.

Методи знаходження найбільш оптимального розкладу характеризується величезною розмірністю та наявність великого числа обмежень складної форми. Найбільш розповсюдженими є:

* …..

Фактично, в даний час, не існує універсальних методів вирішення таких завдань.

**1.2 Огляд існуючих програм та сервісів для пошуку найкоротшого шляху**

1.2.1 Передмова

Перед розгядом аналогів, слід зазначити, що безкоштових версій, які задовольняли б назві Укладач розкладу для ЗВО», або не знаходилося у відкритому доступі, або вони надавалися у значно обмеженому варіанті, що унеможливлювало справедливу і критичну оцінку продукту.

Оновними програмами для складання розкладу у ЗВО є:

* «Ректор-вуз»;
* «Університет 3.2.0.711»;
* «Розклад ПРО»;
* «Програма складання розкладу занять у ВНЗ 2001».

В ході написання роботи був проведений аналіз основних програм для складання розкладу, який проводився за настуними критеріями:

* виконання завдань складання розкладу;
* необхідність закупівлі додаткових програм;
* вартість ліцензії;
* платформа використання;
* зручність і простота інтерфейсу.

1.2.2 Система «Розклад ПРО»

Компанія-видавець: Digsee.

Посилання: <https://www.digsee.com/files/download/timetable/timetable_info.pdf>.

Короткий опис: програма призначена для зручного та швидкого складання розкладу занять шкіл і ЗВО(закладів вищої освіти) в ручному і автоматичному режимі.

Переваги:

* …;
* …;

Недоліки

* ….

Робота програми продемонстрована на рисунку 1.1.

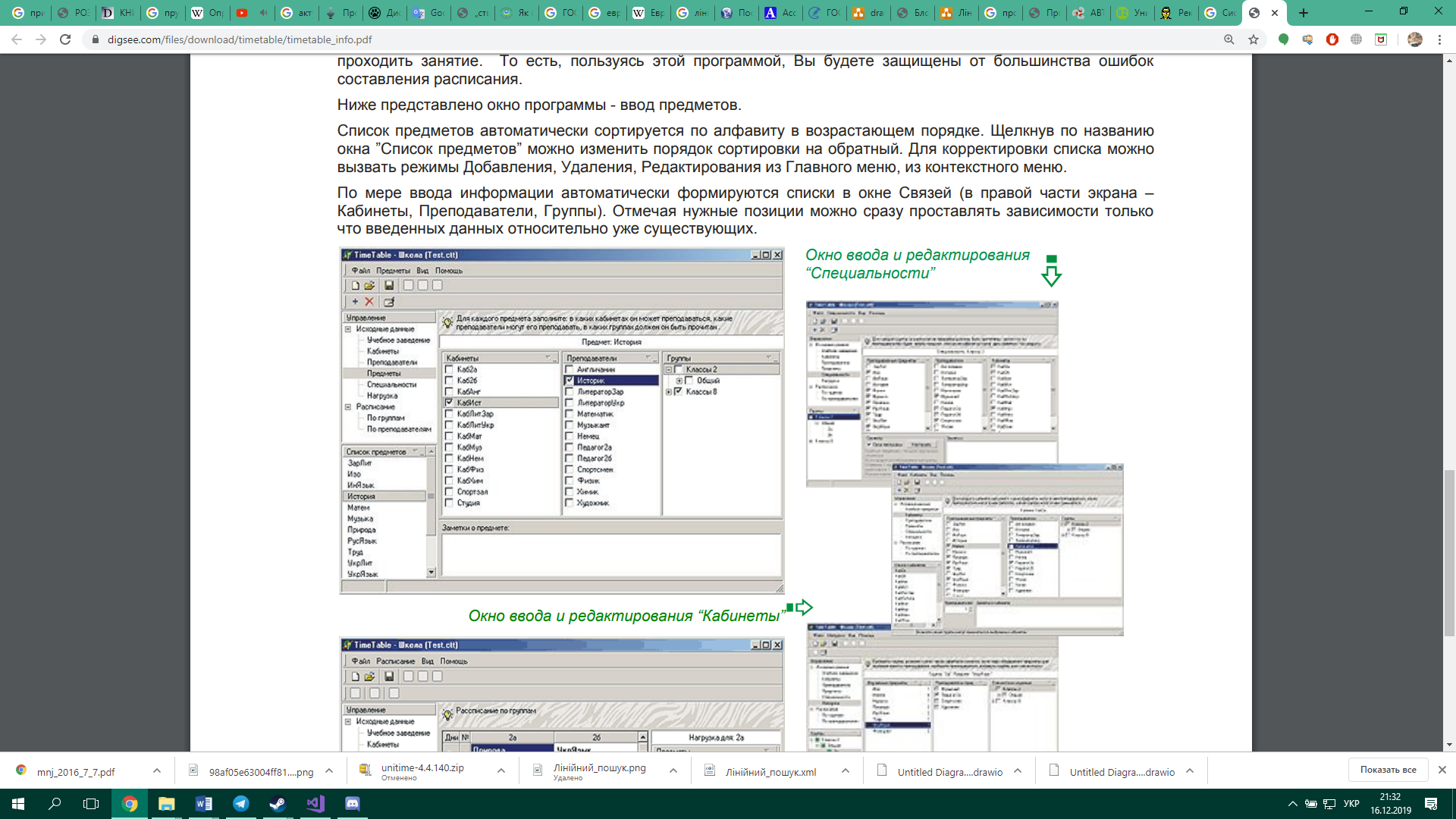


Рисунок 1.1 – Робота програми «Розклад ПРО»

**1.2.3** Университет 3.2.0.711

Компанія-видавець: AzSoft.

Посилання: <http://azsoft.com/ru/timetable-of-classes/5361-universitet-320711.html>.

Короткий опис: Програма дозволяє складати розклад факультету ВНЗ з урахуванням напрямків, змін і розташування аудиторій в різних корпусах в ручному та автоматичному режимах.

Переваги:

* …..

Недоліки:

* …..

1.2.4 «Програма складання розкладу занять у ВНЗ 2001»

Компанія-видавець: невідома.

Посилання: <http://www.interface.ru/home.asp?artId=29844>.

Короткий опис: Програма призначена для автоматизації процесу складання розкладів занять у ВНЗ. Вона володіє по-перше, можливістю редагування розкладу, а по-друге, його автоматичної добудови.

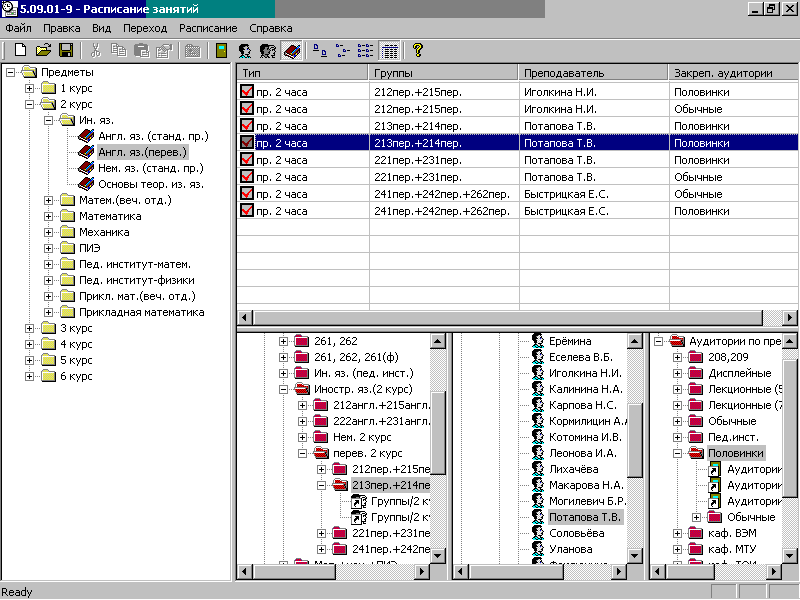
Переваги:

* …..

Недоліки:

* ….

Робота програми продемонстрована на рисунку 1.2.

Рисунок 1.2 – Робота з програмою «Створення розкладу занять у ВНЗ 2001»

2 Аналіз програмних засобів

В даному розділі розглянуті основні особливості програмних засобів, за допомогою який реалізовано курсовий проект.

2.1 Огляд особливостей мови програмування

C# - об'єктно-орієнтована мова програмування.

Теорія про обрану мову програмування, чому її обрали [1].

2.2 Огляд особливостей обраного компілятору

Теорія про обраний компілятор

2.3 Огляд класової ієрархії

Опис класів, що використовувались при розробці

2.4 Висновки з розділу

У даному розділі було описано мову програмування та її особливості. Також було описано ….

3 основні рішення з реалізації компонентів системи

В даному розділі розглянуті основні рішення з розробки класів, основні розроблені алгоритми, рішення щодо розробки інтерфейсу користувача, рішення щодо збереження даних та використання бази даних.

3.1 Основні рішення щодо розроблених класів

Структура програми складається з 18 класу, 15 з яких – користувацькі, 3 – класи форм. Один з користувацьких класів є абстрактним. Діаграми користувацьких класів знаходяться у додатку А.

Класову структуру програми зображено на рисунку 3.1.

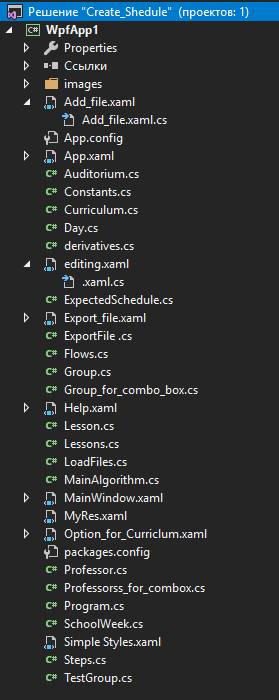


Рисунок 3.1 – Класова структура програми

3.1.1 Клас LoadFiles

Клас LoadFiles містить у собі основний інструментарій для роботи із завантаження файлів Excel, перевірки та форматування цих даних перед занесенням у базу даних (БД), встановлення зв’язку із БД, розриву цього зв’язку та завантаження інформації з таблиць, що є в наявності у БД. Код файлу визначення класу знаходиться в додатку Б. Дані та методи класу наведені у таблиці 3.1.

Таблиця 3.1 – Опис полів та методів класу LoadFiles

|  |  |
| --- | --- |
| Поля та методи класу | Опис |
| **1** | **2** |
| ***public:*** | |
| SQLiteConnection DB | Змінна, за допомогою якої встановлюється зв'язок із БД. |
| public static DataTable loadTable(string nameOfTable, bool flag = true) | Метод, що дозволяє отримати дані певної таблиці з БД та, за бажання, очистити поле Id. |
| public void LoadDB(string dataSource) | Метод, що створює зв'язок із БД. |
| public void CloseDB() | Метод, що розриває зв'язок із БД. |
| public void LoadExcelFileForCurriculum (string filename, string name\_of\_course) | Метод, що дозволяє отримати дані про навчальний план за допомогою Excel-файлу, замінити тип аудиторії та назву кафедри на відповідне Id та перенести їх у БД |

Продовження таблиці 3.1

|  |  |
| --- | --- |
| **1** | **2** |
|  | файлу, замінити тип аудиторії та назву кафедри |
| public void LoadExcelFileForProfessors(string filename) | Отримує дані про викладачів за допомогою Excel-файлу, замінити посаду(ранг) викладача та назву кафедри на відповідне Id та перенести їх у БД |
| public void LoadExcelFileForGroup(string filename) | Отримує дані про потоки та групи, що їм належить, за допомогою Excel-файлу (замінивши назву кафедри на відповідне Id) та перенести їх у БД, розділивши на 2 окремі таблиці. |
| ***protected:*** | |
| protected static void inputCheckForColumnsNames(ref DataTable dataTable) | Превіряє на правильність та відповідність шаблону назви колонок у Excel-файлі, та замінює на стандартну для БД назву. |
| protected static int inputCheckForAuditoriumField(string input\_data) | Превіряє на правильність та відповідність шаблону назви типів аудиторій у Excel-файлі, та замінює на стандартну для БД назву. |
| protected static int inputCheckForDepartmentField(string input\_data) | Превіряє на правильність та відповідність шаблону назви кафедр у Excel-файлі, та замінює на стандартну для БД назву. |

Продовження таблиці 3.1

|  |  |
| --- | --- |
| protected static int inputCheckForProfessorsRankField(string input\_data) | Превіряє на правильність та відповідність шаблону назви посад викладачів у Excel-файлі, та замінює на стандартну для БД назву. |

3.1.2 Клас ExportFiles

….

3.1.3 Клас Flows

….

3.1.4 Клас Auditorium

3.2 Основні розроблені алгоритми

3.2.1 Аналіз головного алгоритму

В ході написання курсової роботи був створений власний алгоритм складання розкладу.

Так і розроблений алгоритм, видає прийнятний розклад, але не гарантує, що він є найкращим серед можливих. Це позначає найголовнішу особливість алгоритму, а саме подальшу оптимізацію та можливе розширення виконуваних завдань.

Вхідними даними для алгоритму є:

* …..

Під час підрахунку остаточного результату алгоритм працює з проміжними даними:

* …..

У кінці виконання алгоритм формує вихідні данні:

* …..

Усі значення мають бути коректними та строго формалізованими (це виконується на етапі створення бази даних).

Увесь алгоритм можна поділити на 2 етапи:

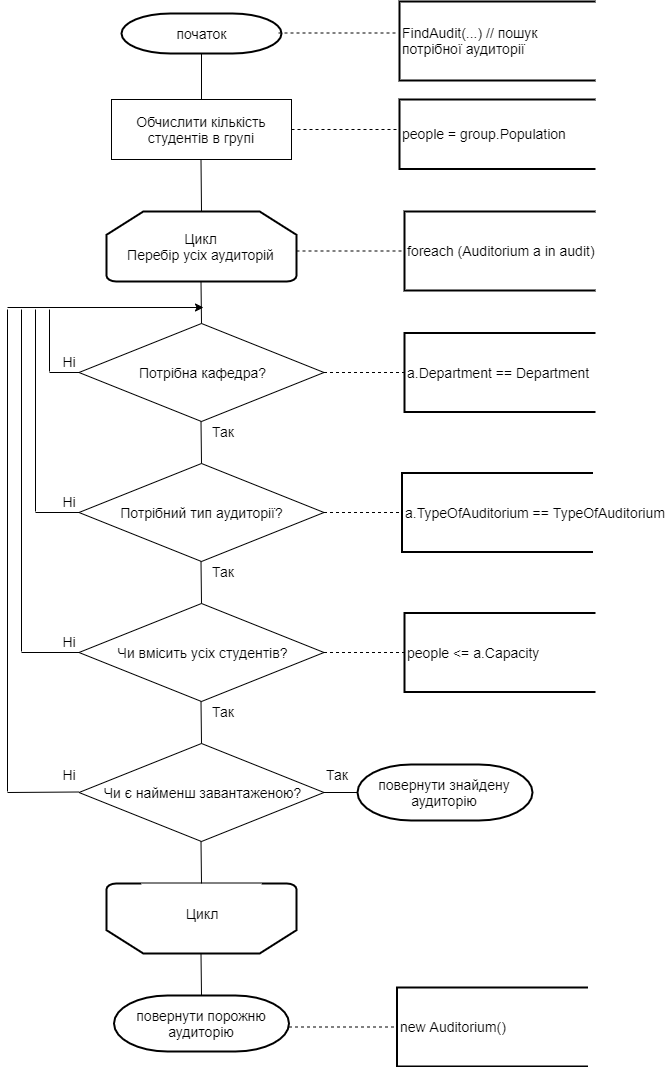
1. Формування списку занять;
2. Пошук часу проведення заняття.

Виконання кожного етапу базується та таких алгоритмах:

* алгоритм лінійного пошуку;
* алгоритм сортування вставками;
* елементи динамічного програмування.

Розглянемо основні типові алгоритми.

Опис розроблених алгоритмів

Рисунок 3.19 – Пошук відповідної аудиторії для заняття

….

Після виконання алгоритму маємо складений розклад для навчальних груп на потоках, аудиторій та викладачів.

3.3 Основні рішення щодо розробки інтерфейсу

3.3.1 Вибір платформи.

Головнним рішенням, щодо розробки інтрерфейсу, було вибрати нову та сучасну платформу для розробки інтерфейсу WPF (Windows Presentation Foundation). Ця платформа відзначаеться кросплатфоменістю та більш зручним інтерфейсом для програміста.

Опис рішень, які використовувались для реалізації інтерфейсу

3.3.2 Основні рішення з впорядкування елементів інтерфейсу

Під час розробки інтерфейсу користувача було використано наступні е.г.і.:

* MainWindow
* VHLayouts
* TextBrowser
* PushButton
* Label
* LineEdit
* MenuBar
* TextEdit
* Frame
* ComboBox

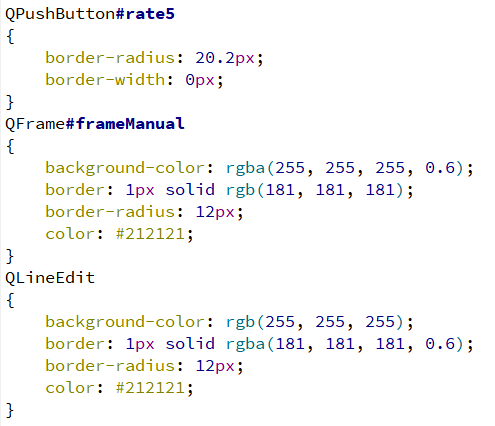
Саме вони були використані тому що, по перше, це одні з найпростіших та найпопулярніших е.г.і. в цілому, по друге, до цих елементів доволі просто застосовувати стилі та інші графічні зміни, по третє, ми вже маємо досвід роботи з цими е.г.і.

3.3.3 Основні рішення з комунікації з користувачем

Для кращої комунікації з користувачем та рівномірного розподілення місця у додатку були використані кнопки(PushButton) для запиту певної стандартної інформації у бота, та строка вводу (TextEdit, LineEdit) для нестандартної інформації такої як: особиста інформація користувача для надання ії керівникам АМ, або повідомлення про помилку\баг у додатку.

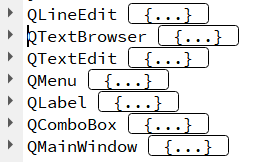
3.3.4 Основні рішення з створенням інтерфейсу

Було активно використано стилі. Тобто відокремлення певної групи елементів і створення для них власного стилю. Приклад данного рішення наведений нижче на рисунку 3.27.1…..

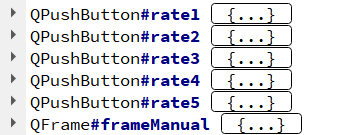


Основним варіантом застосування стилыв є використання Qt Style Sheets. Як стає очевидно з назви, це трохи специфічний аналог звичних всім каскадних таблиць стилів CSS(Cascading Style Sheets), без яких немислимий сучасний інтернет. Оскільки в минулому семестрі у нас була дисципліна «WEB-дизайн» і ми активно використовували CSS в своїх роботах, то нам здалося зручним використовувати даний аналог і тут. Також це зручно в тому плані, що всі графічні рішення виводяться в окремий файл, не займають місце в основному коді і довантажуються під час запуску програми, а також потрібний стиль застосовується при виборі теми програми в Налаштуваннях.

Було використано звернення по класу. Це зручно тим, що зміни застосовуються відразу до всіх елементів даного класу. Приклад:



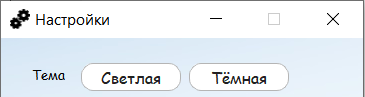
Звернення по імені дає можливість кастомізувати конкретний елемент з усього класу. Приклад:



Псевдо-стан елемента допомагає зробити програму більш динамічною і "живий" для користувача. Приклад:



Також було прийнято рішення додати повністю альтернативну тему для усього додатку, т.з. темний стиль зокрема від світлого. Ця ідея прийшла до нас у процесі створення програми вночі та ми подумали, що користувачеві буде зручно змінювати колірний стиль усього додатку одним кліком у Налаштуваннях.

3.4 Основні рішення щодо розробки бази даних

3.4.1 Основні рішення щодо вибору бази даних

База даних, як один з основних компонентів програми, потребує зваженого вибору середовища та платформи розробки, бо від цього залежатиме якість та комфорт не тільки у користуванні, а й у написанні програмного продукту, отже, саме тому основними критеріями для вибору БД були:

* …..

Отже, проаналізувавши наявні пропозиції у цій сфері, платформа …. вирізняється з-поміж інших варіантів, та відповідає усім заданим критеріям, а саме:…..

3.4.2 Структура збереження даних у БД

У процесі збереження даних використовуються таблиці, як ключові носії інформації, та представлені наступними екземплярами:

«*Auditorium*» - таблиця для збереження даних аудиторій, а саме: номер, літера, кількість місць, тип аудиторії та кафедра, якій вона належить (рис. 3.28).

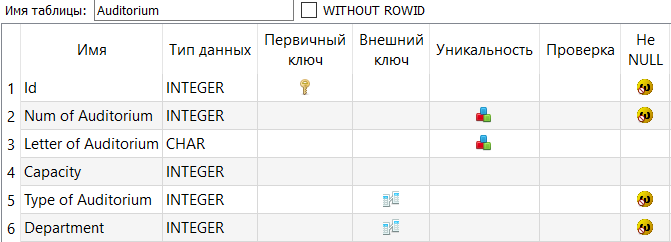


Рисунок 3.28 – Структура таблиці «Auditorium»

Первинним ключем є поле Id, яке ідентифікує кожен запис та допомагає зекономити пам'ять у інших таблицях, адже замість сполучення номеру, літери та кафедри використовується саме Id. Поля «Types of Auditorium» та «Department» пов’язані зовнішнім ключем з допоміжними таблицями для економії пам’яті та збереження цілісності даних, бо зовнішній ключ виступає перешкодою для внесення неіснуючих чи нерелевантних даних.

….

3.4.2 Основні запити SQL, що було використано

Мова для створення запитів SQlite є SQL-подібна, відрізняючись лише синтаксисом (несуттєво) та функціоналом. В ході роботи з базою даних були використані наступні запити для взаємодії:

* ….

4 керівництво програміста

В даному розділі розглянуті призначення, умови застосування, характеристика програми, звертання до програми, початкові та вихідні дані та представлені повідомлення.

4.1 Призначення та умови застосування програми

4.1.1 Призначення програми

Програма “Створення розкладу” призначена для автоматичного стоврeння розкладу на основі даних користувача.

4.1.2 Функції програми

Функції, що виконує програма:

* …..

4.1.3 Умови застосування програми

До комп’ютеру, на якому виконується програма, висуваються наступні вимоги:

* операційна система Windows 7, 8, 10;
* наявність дисплею;
* наявність миші та клавіатури;
* до 300 МБ оперативної пам’яті за умови використання функції створити розклад.

4.2 Характеристика програми

Програма виконана за допомогою мови програмування високого рівня С# в середовищі розробки Visual Studio 2017 , 2019 за допомогою платформи WPF.

4.2.1 Структура програми

Програма складається з 18 класу, 15 з яких – користувацькі, 3 – класи форм. Один з похідних класів інтерфейсу, яких ще 5, є один абстрактний клас. Проект містить наступні класи та їх реалізацію, файли ресурсів, базу даних SQL. Ієрархія представлена на рисунку 4.1.

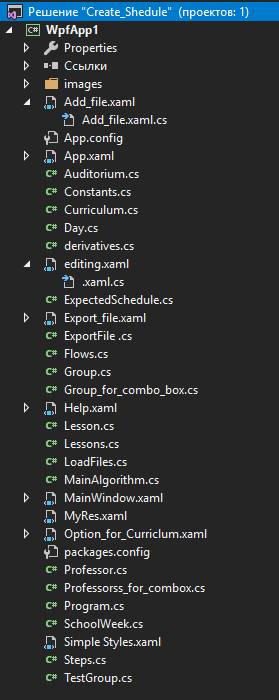


Рисунок 4.1 – Ієрархія класів проекту

4.2.2 Алгоритм роботи програми

….

4.3 Звертання до програми

Для запуску програми потрібно попередньо упевнитися в відповідності характеристик комп’ютера, на який встановлена програма, до системних вимог програми. Потрібно упевнитися у наявності всіх файлів програми.

Звертання до програми передбачене одним способом: через виконуваний файл Create\_Shedule.exe. Звертання до програми через командний рядок не передбачене, адже програма розроблена саме для роботи через інтерфейс для складання розкладу.

4.4 Вхідні та вихідні дані

4.4.1 Вхідні дані

Вхідні дані уявляють собою файли Екселю, з розширення .xlxs та .xlx. Вони повинні буди чотирьох різних типів. У кожному з яких за шаблоном повинні міститися дані в залежності від таблиці.

4.4.2 Вихідні дані

Файли Екселю з розширення .xlxs, .xlx. Вони повинні буди чотирьох різних типів. У кожній з яких за шаблоном повинні міститися дані в залежності від таблиці. Та є можливість зберегти файл з готовим розкладом.

4.5 Повідомлення

* ….

5 керівництво користувача

В даному розділі розглянуто призначення програми, умови її виконання, процес виконання програми та повідомлення для користувача.

5.1 Призначення програми

Програма “Створення розкладу” призначена для автоматичного стоврeння розкладу на основі даних користувача.

5.2 Умови виконання програми

5.2.1 Апаратні вимоги програми

До комп’ютеру, на якому виконується програма, висуваються наступні вимоги:

* наявність дисплею;
* наявність клавіатури та миші;
* до 300 МБ оперативної пам’яті за умови використання функції створення розкладу.

5.2.2 Вимоги до користувача

До користувачів висуваються наступні вимоги:

* …..

5.3 Виконання програми

5.3.1 Запуск програми

Для запуску програми потрібно попередньо упевнитися в відповідності характеристик комп’ютера, на який встановлена програма, до системних вимог програми. Другим кроком потрібно упевнитися в наявності всіх бібліотек для роботи програми та в наявності бази даних для повноцінного функціонування.

Звертання до програми передбачене одним способом: через виконуваний файл. Звертання до програми через командний рядок не передбачене, адже програма розроблена саме для візуалізації роботи алгоритмів пошуку найкоротшого шляху.

Після запуску виконуваного файлу Create\_Shedule.exe з’являється головна форма програми, що свідчить про початок роботи з програмою. Головна форма програми зображена на рисунку 5.1.

Рисунок 5.1 – Головна форма програми

5.3.2 Виконання роботи з програмою

Спершу користувач має ….

5.4 Повідомлення користувачу

….

5.5 Довідка програми

Для зручності користувача та для отримання допомоги українською мовою в будь-який момент часу було створено довідку для програми. Форма довідки зображена на рисунку 5.3.

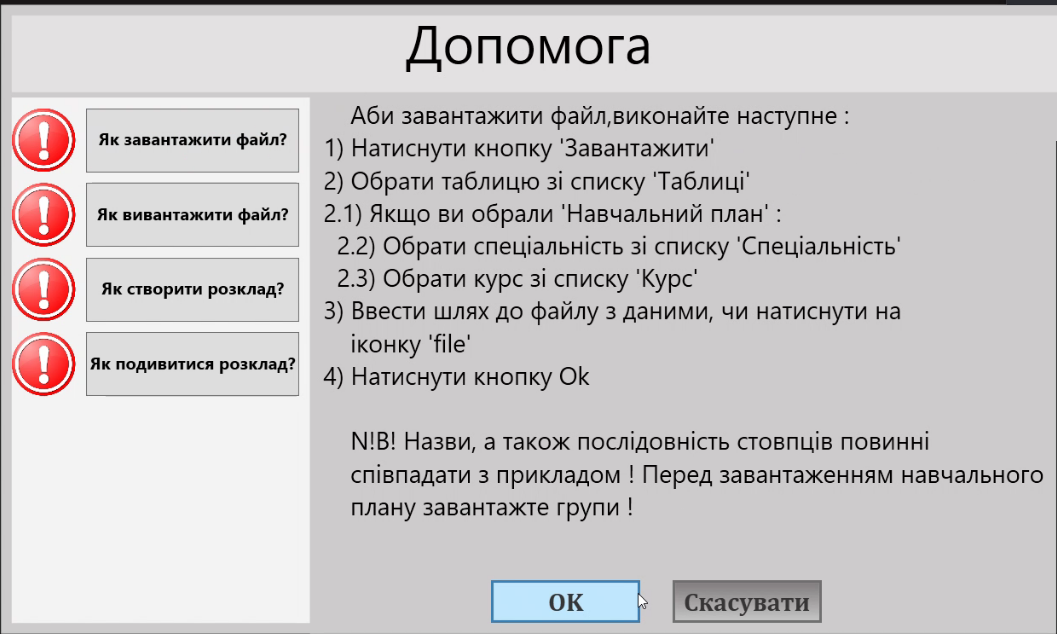


Рисунок 5.3 – Форма довідки

Висновки

Під час виконання курсового проекту було розроблено програмне забезпечення для складання розкладу для навчального закладу.

Було виконано огляд існуючих методів вирішення завдання та аналогів, проаналізовано переваги та недоліки і розроблено функціонал власного програмного продукту.

Для програми було реалізовано інструментарій для роботи з файловим вводом та вивидом інформації, на основі MS Excel, що є звичним для цільової аудиторії користувачів та значно облегшує роботу з продуктом.

Було розроблено алгоритм побудови розкладу, основними ознаками якими є:

* результат, який чітко відповідає початковим вимогам користувача;
* можливість змінювати початкові параметри та впливати на хід роботи алгоритму.

Було розроблено максимально прости та зрозумілий інтерфейс, у якому не має зайвих функцій.

Одною з ключових переваг розробленого продукту було закладення засад для подальшого розвитку та модернізації. Зважаючи на актуальність технологій, які були використані, якісне проектування бази даних, алгоритму та інтерфейсу, цей програмним продукт має усі перспективи до подальшої розробки, а саме:

* створення алгоритму, який знаходить найбільш оптимальне рішення;
* синтез інтерфейсу та бази даних, для перетворення програми на повноцінний редактор розкладу.

ПЕРЕЛІК ПОСИЛАНЬ

1) Кормен Т., Лейзерсон Ч., Ривест Р. Глава 16. Динамическое программирование // Алгоритмы: построение и анализ(третье издание). — М.: «МІТ», 1990 — С. 299—317.

2) Кормен Т., Лейзерсон Ч., Ривест Р. Введение // Алгоритмы: построение и анализ(третье издание). — М.: «МІТ», 1990 — С. 7—17.

3) Mark Allen Weiss, Using a Table Instead of Recursion // Data Structures and Algorithm in C++(fourth edition). — Florida International University, 2013. –482 – 485p.

4) Шилдт Г. Полный справочник по C# / Ручко Н.М. — М.: «Вильямс», 2005. — 748 с.

5) «Евристичні алгоритми» [Електрон. ресурс]. – Режим доступу: <http://mmsa.kpi.ua/sancho/ASD_HTM/Arti04.html>.

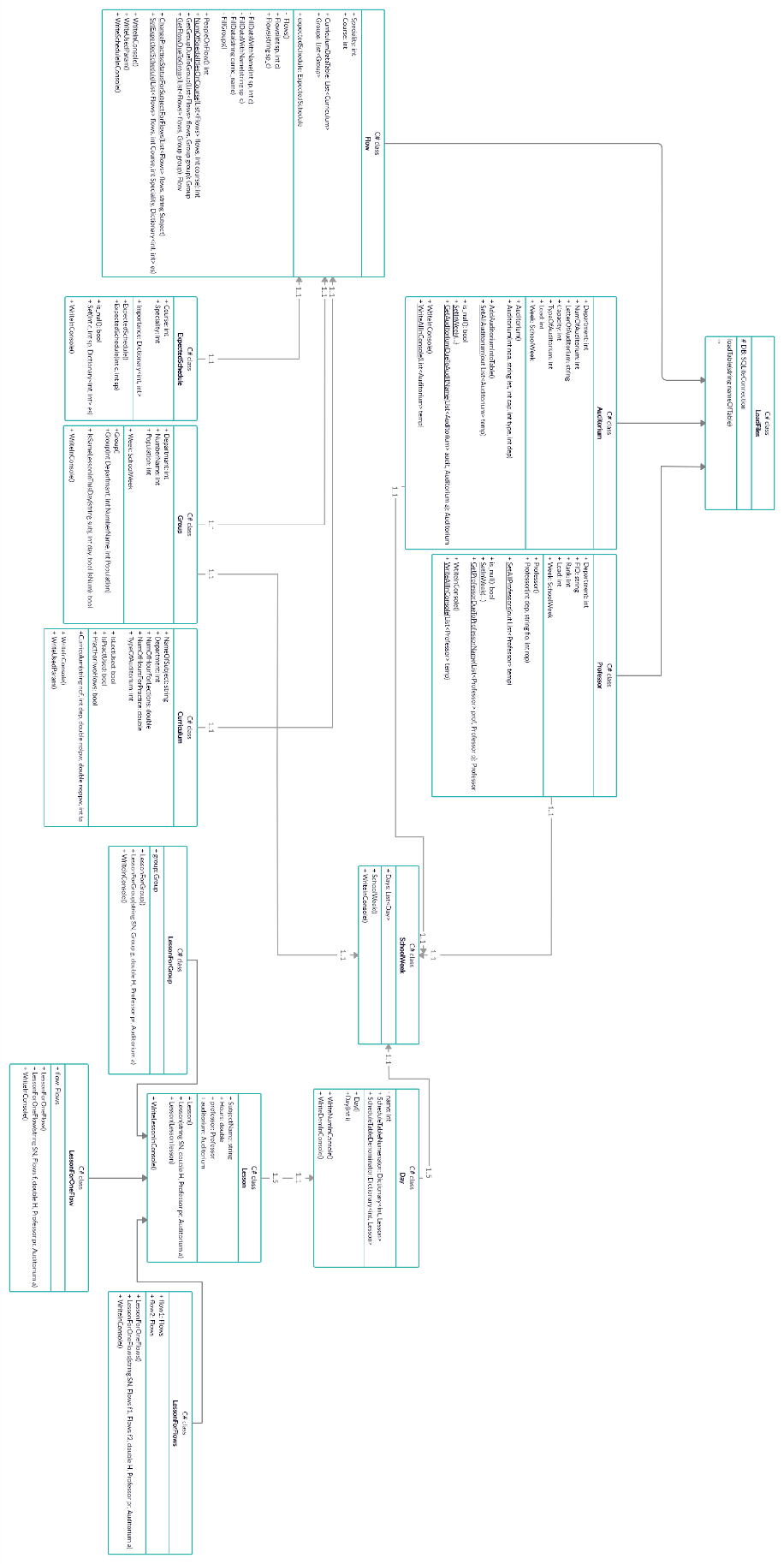
6) “What Is SQLite?” [Electronic resource]. – Access mode: <https://www.sqlite.org/index.html>.

7) “Free Spire.XLS for .NET” [Electronic resource]. – Access mode: https://www.e-iceblue.com/Introduce/free-xls-component.html#.XfmogfxoRPY.

8) “ADO.NET” [Electronic resource]. – Access mode: https://docs.microsoft.com/en-us/dotnet/framework/data/adonet/.

9) «Алгоритм лінійного пошуку» [Електрон. ресурс]. – Режим доступу: http://kytok.org.ua/?p=311.

Додаток А Діаграма класів



Додаток Б Код Програми

Б1 - Клас Auditorium

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SQLite;

using System.Windows;

using System.Data;

namespace Schedule

{

class Auditorium : LoadFiles

{

public int Department { get; set; }

public int NumOfAuditorium { get; set; }

public string LetterOfAuditorium { get; set; }

public int Сapacity { get; set; }

public int TypeOfAuditorium { get; set; }

public SchoolWeek Week; // расписание для аудиторий