

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Кафедра програмної інженерії

КУРСОВА РОБОТА
ПОЯСНЮВАЛЬНА ЗАПИСКА
з дисципліни “Об’єктно -орієнтоване програмування”
ДОВІДНИК ФАНАТА

Керівник , проф.

Бондарєв В.М.

Студент гр. ПЗПІ-22-1

Петренко М.О.

Комісія:

Проф. Бондарєв В.М.,

Ст. викл. Черепанова Ю.Ю.,

Ст. викл. Ляпота В.М.

Харків 2023

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
РАДІОЕЛЕКТРОНІКИ

Кафедра: *програмної інженерії*

Рівень вищої освіти: *перший (бакалаврський)*

Дисципліна: *Об'єктно-орієнтоване програмування*

Спеціальність: *121 Інженерія програмного забезпечення*

Освітня програма: *Програмна інженерія*

Курс 1.

Група ПЗП-22-1.

Семестр 2.

ЗАВДАННЯ

на курсовий проект студента

Петренко Марія Олександрівна

1 Тема проекту: *Довідник фаната*

2 Термін здачі студентом закінченого проекту: *“16” - червня - 2023 р.*

3 Вихідні дані до проекту:

Специфікація програми, методичні вказівки до виконання курсової роботи

4 Зміст розрахунково-пояснювальної записки:

Вступ, опис вимог, проектування програми, інструкція користувача, висновки

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапу	Термін виконання
1	Видача теми, узгодження і затвердження теми	13.02.2023 - 14.03.2023 р.
2	Формулювання вимог до програми	28.04.2023 - 29.04.2023 р.
3	Розробка підсистеми зберігання та пошуку даних	04.05.2023 - 15.05.2023 р.
4	Розробка функцій додавання, редагування та видалення інформації про спортсмена	20.05.2023 - 23.05.2023 р.
5	Розробка функцій зберігання та завантаження даних	23.05.2023 – 26.05.2023 р.
6	Тестування і доопрацювання розробленої програмної системи.	27.05.2023 - 30.05.2023
7	Оформлення пояснювальної записки, додатків, графічного матеріалу	28.05.2023 - 31.05.2023
8	Захист	05.06.2023 - 16.06.2023 р

Студент

Петренко Марія Олександрівна

Керівник

Бондарєв Володимир Михайлович

« 13 »_лютого_____ 2023 р.

РЕФЕРАТ

Пояснювальна записка до курсової роботи: 36 с., 1 табл., 14 рис., 2 джерела.

СПОРТСМЕН, БАЗА, РЕКОРДСМЕН, ООП, МОВА C#.

Метою роботи є програма «Довідник фаната», яка буде надавати користувачу можливість працювати за даними про спортсменів.

В результаті отримана програма, яка дозволяє зберігати список спортсменів, характеристики кожного спортсмена, такі як: ім'я, прізвище, громадянство, національність, зріст, вид спорту, команда(якщо є), персональний рекорд та чи є спортсмен рекордсменом. Є можливість пошуку спортсменів і рекордсменів, а також редагування, видалення чи додавання інформації про спортсмена.

В процесі розробки використано середовища Microsoft Visual Studio 2022, фреймворк Windows Forms, платформи .NET 7.0, мова програмування C#.

ЗМІСТ

Вступ.....	6
1 Опис вимог.....	7
1.1 Сценарії використання.....	7
1.2 Функціональні вимоги.....	11
2 Проектування програми.....	16
3 Інструкція користувача.....	18
3.1 Запуск програми.....	18
3.2 Видалення програми.....	18
3.3 Використання програми.....	18
Висновки.....	26
Перелік джерел посилання.....	27
Додатки.....	28

ВСТУП

Метою даної роботи є закріплення знань, набутих при вивченні дисципліни „Об’єктно-орієнтоване програмування”, а також придбання навичок у розробці програмного продукту. Ці знання та навички допоможуть у створенні наступних, більш складних проектів.

Типовою діяльністю користувача програми є заняття спортом або перегляд спортивних змагань. Людина цікавиться спортсменами та спортсменками, любить дізнаватись нові факти з їх життя. Проте, запам’ятати інформацію про кожного спортсмена – дуже складно.

Завдяки використанню програми, користувач може зберігати всі необхідні дані про спортсменів в одному місці. Також він може редагувати та видаляти непотрібну інформацію. Для того щоб швидко знайти якогось спортсмена, користувач може скористатися пошуком по імені, прізвищу або виду спорту. Ще є пошук рекордсмена в заданому виді спорту.

1 ОПИС ВИМОГ

1.1 Сценарії використання

Сценарій 1. Пошук спортсмена

Передумова

Користувач запустив програму та відкрив головне вікно.

Основний сценарій

1. Користувач вводить ім'я, прізвище або вид спорту на панелі пошуку та натискає кнопку “search”.
2. Програма знаходить спортсменів, які відповідають умовам пошуку.
3. Програма виводить на форму список потрібних спортсменів та інформацію про них у вигляді таблиці.

Додатковий сценарій

1. Користувач вводить ім'я, прізвище або вид спорту на панелі пошуку та натискає кнопку “search”.
2. Програма не знаходить спортсменів, які задовольняють умови пошуку.
3. Користувач бачить пусту таблицю.

Сценарій 2. Додавання книги

Передумова

Користувач запустив програму та відкрив головне вікно.

Основний сценарій

1. Користувач натискає на кнопку “Sportsman” у меню, яке знаходиться у верхній частині екрану, та обирає пункт “Add” у випадному списку.

2. Програма виводить на екран вікно для додавання нового спортсмена.
3. Користувач вводить дані про спортсмена у текстові поля та натискає кнопку “OK”.
4. Програма перевіряє коректність введених даних та додає нового спортсмена у список.
5. Користувач бачить на екрані оновлений список спортсменів.

Додатковий сценарій

1. Користувач натискає на кнопку “Sportsman” у меню, яке знаходиться у верхній частині екрану, та обирає пункт “Add” у випадному списку.
2. Програма виводить на екран вікно для додавання нового спортсмена.
3. Користувач вводить дані про спортсмена у текстові поля та натискає кнопку “OK”.
4. Програма перевіряє коректність введених даних, знаходить помилки та повідомляє користувача про причину цих помилок.

Додатковий сценарій

1. Користувач натискає на кнопку “Sportsman” у меню, яке знаходиться у верхній частині екрану, та обирає пункт “Add” у випадному списку.
2. Програма виводить на екран вікно для додавання нового спортсмена.
3. Користувач вводить дані про спортсмена у текстові поля та натискає кнопку “Cancel”.
4. Програма закриває вікно для додавання спортсмена.
5. Користувач бачить головне вікно програми зі списком спортсменів.

Сценарій 3. Редагування даних

Передумова

Користувач відкрив головне вікно програми, у списку спортсменів вже є якісь дані.

Основний сценарій

1. Користувач обирає спортсмена, інформацію про якого він хоче відредагувати.
2. Користувач натискає на кнопку “Sportsman” у меню, яке знаходиться у верхній частині екрану, та обирає пункт “Edit”.
3. Програма відкриває вікно для редагування інформації.
4. Користувач змінює дані у текстових полях та натискає на кнопку “OK”.
5. Програма перевіряє коректність введених даних та змінює інформацію про обраного спортсмена.
6. Користувач бачить головне вікно програми зі зміненим списком спортсменів.

Додатковий сценарій

1. Користувач обирає спортсмена, інформацію про якого він хоче відредагувати.
2. Користувач натискає на кнопку “Sportsman” у меню, яке знаходиться у верхній частині екрану, та обирає пункт “Edit”.
3. Програма відкриває вікно для редагування інформації.
4. Користувач змінює дані у текстових полях та натискає на кнопку “OK”.

5. Програма перевіряє коректність введених даних, знаходить помилки та повідомляє користувача про причини їх виникнення.

Додатковий сценарій

1. Користувач обирає спортсмена, інформацію про якого він хоче відредагувати.

2. Користувач натискає на кнопку “Sportsman” у меню, яке знаходиться у верхній частині екрану, та обирає пункт “Edit”.

3. Програма відкриває вікно для редагування інформації.

4. Користувач змінює дані у текстових полях та натискає на кнопку “Cancel”.

5. Програма закриває вікно для редагування.

6. Користувач головне вікно програми зі списком спортсменів.

Сценарій 4. Видалення спортсмена

Передумова

Користувач відкрив головне вікно програми, у списку спортсменів є хоча б один спортсмен.

Основний сценарій

1. Користувач обирає спортсмена, інформацію про якого він хоче відредагувати.

2. Користувач натискає на кнопку “Sportsman” у меню, яке знаходиться у верхній частині екрану, та обирає пункт “Delete”.

3. Програма видаляє обраного спортсмена.

4. Користувач бачить оновлений список спортсменів.

Сценарій 5. Пошук рекорсмена

Передумова

Користувач запустив програму та відкрив головне вікно.

Основний сценарій

1. Користувач вводить вид спорту на панелі пошуку та натискає кнопку “find recordsman”.
2. Програма знаходить спортсмена або спортсменів, які є рекорсменами в заданому виді спорту.
3. Програма виводить на форму список потрібних спортсменів та інформацію про них у вигляді таблиці.

1.2 Функціональні вимоги

Функція 1. Пошук спортсмена

Панель пошуку знаходиться на головному вікні програми. У ній присутнє текстове поле, куди користувач вводить або ім'я, або прізвище, або вид спорту спортсмена, якого хоче знайти. Нижче зліва знаходиться кнопка “search”, при натисканні на яку відбувається пошук спортсменів. У текстовому полі регістр не важливий. На рисунку 1.1 наведена схема розташування елементів головної сторінки програми.

File Sportsman			
<div style="border: 1px solid black; width: 300px; height: 25px; margin: 0 auto;"></div> <div style="display: flex; justify-content: space-around; margin-top: 10px;"> <div style="border: 1px solid black; width: 100px; height: 25px;"></div> <div style="border: 1px solid black; width: 100px; height: 25px;"></div> </div>			
1	—	—	—
2	—	—	—
3	—	—	—

Рисунок 1.1 – Схема головного вікна програми

Функція 2. Пошук рекордсмена

Для пошуку рекордсмена використовується теж саме поле, як і для звичайного пошуку рекордсмена. Пошук відбувається після натискання на кнопку “find recordsman”, яка знаходиться нижче справа. Результат пошуку відображається внизу сторінки у вигляді таблиці з даними спортсмена або спортсменів.

Функція 3. Додавання спортсмена

Для додавання спортсмена користувач натискає на пункт меню “Sportsman”. Відкривається випадний список, де є пункт “Add”. При натисканні відкривається нове вікно, на якому знаходиться 9 текстових полів. Це:

- а) “First name”
- б) “Last name”
- в) “Citizenship”
- г) “Nationality”

г) “Height”

д) “Kind of sport”

е) “Team”

є) “Personal record”

ж) “Recordsman (yes or no)”

Поля в), е) та є) не є обов’язковими для заповнення. У поля а), б), в), г), д) можна вводити лише букви та дефіс, а довжина рядка не повинна перевищувати 50 символів. Поле г) користувач може заповнювати лише цифрами, максимум – 3 символи. Поля е) та є) –мають містити менше ніж 100 символів кожен. У поле ж) можна вводити лише “yes” або “no”, регістр не важливий.

У разі виникнення помилок посередині екрану з’являється вікно з повідомленням про поле, у якому користувач помилився.

Кнопки “OK” та “Cancel” знаходяться у правому нижньому куті вікна.

При натисканні кнопки “OK” програма запитує в користувача підтвердження дії додавання і, у разі позитивної відповіді, додає спортсмена до списку на головному вікні програми.

При натисканні кнопки “Cancel”, вікно додавання закривається і користувач бачить головне вікно. Схематичний зовнішній вигляд вікна для додавання спортсмена наведений на рисунку 1.2.

_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____

Рисунок 1.2 – Схема вікон для редагування та для додавання

Функція 4. Редагування інформації про спортсмена

Для того щоб відредагувати дані про вже існуючого спортсмена, користувач має обрати спортсмена за допомогою комп'ютерної миші або клавіатури. Потім потрібно натиснути на пункт меню "Sportsman". Відкривається випадний список, де є пункт "Edit". Після натискання на нього, відкривається вікно для редагування з такими ж полями як і при додаванні, також є кнопки "OK" та "Cancel". При натисканні кнопки "OK" програма запитує підтвердження дії. Відрізняється тільки те, що у вікні для редагування у полях вже є дані, які були написані раніше та які користувач хоче змінити. Вимоги до заповнення полів такі ж самі як і для додавання інформації про спортсмена.

Зовнішній вигляд вікна такий як на рисунку 1.2.

Функція 5. Видалення спортсмена

Для видалення існуючого спортсмена зі списку, користувачу потрібно обрати спортсмена та натиснути на пункт меню "Sportsman". Відкривається випадний список, потім користувач обирає "Delete". Програма виводить повідомлення по центру екрану про підтвердження дії. І, після позитивної

відповіді, на головне вікно програми виводить оновлений список спортсменів. У разі не підтвердження дії нічого не змінюється.

Функція 6. Очищення списку

Для того, щоб очистити весь список спортсменів, користувач натискає на пункт меню “File”. У випадному списку є метод “Clear”. Якщо користувач натисне на нього, програма запитує підтвердження дії, і, якщо відповідь користувача позитивна, очищає весь список спортсменів.

Функція 7. Збереження даних

У випадному списку, який з’являється після натискання “File”, є пункт “Save”. Якщо користувач натисне на нього, виникає вікно для підтвердження дії збереження. Після схвальної відповіді програма зберігає всі останні зміни.

Функція 8. Вихід з програми

У випадному списку, який з’являється після натискання “File”, останнім пунктом є “Exit”. Якщо користувач обирає цей пункт, програма пропонує зберегти незбережені зміни, якщо вони є, і після відповіді користувача закривається. Так само вихід працює, якщо натиснути на хрестик у правому верхньому куті вікна.

Функція 9. Завантаження даних

У випадному списку, який з’являється після натискання “File”, є пункт “Load”. Якщо користувач натискає на цей пункт, програма завантажує останні збережені дані про спортсменів.

2 ПРОЕКТУВАННЯ ПРОГРАМИ

Для програми була обрана настільна архітектура з GUI. Всі файли програми зберігаються в 4 папках, таких як:

- Data
- Forms
- Forms Validation
- Models

Зовнішні дані зберігаються у форматі JSON. Взаємодія частин програми зображена на рисунку 2.1.

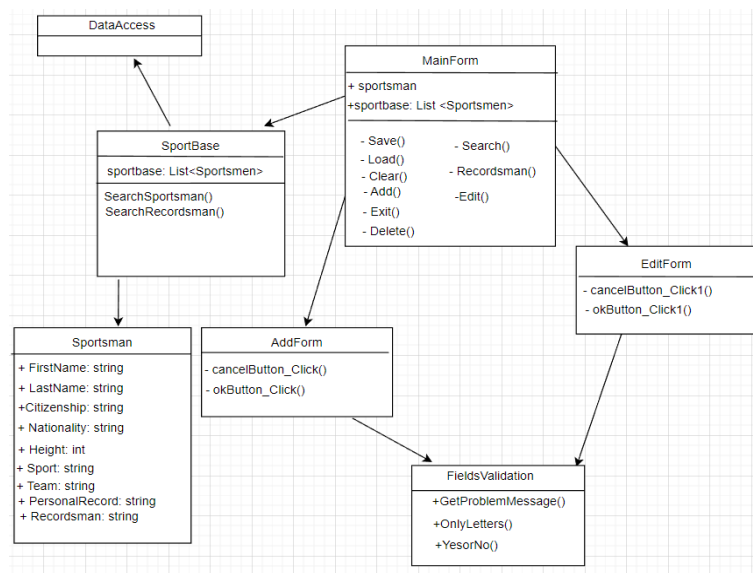


Рисунок 2.1 – діаграма класів програми

На діаграмі представлені такі класи:

- MainForm (клас головної форми)

- AddForm (клас форми для додавання)
- EditForm (клас форми для редагування)
- FieldsValidation (клас для перевірки коректності введеної інформації в полях)
- SportBase (клас списку спортсменів)
- Sportsman (клас спортсмена)
- DataAccess (клас для завантаження та збереження даних)

3 ІНСТРУКЦІЯ КОРИСТУВАЧА

3.1 Запуск програми

Для того, щоб запустити програму, необхідно завантажити “the fan`s guide.zip” та розархівувати його. Після цього потрібно зайти у папку the fan`s guide, потім у папку bin, у папку Debug, net.7.0-windows. Там потрібно знайти файл з розширенням .exe та запустити його.

3.2 Видалення програми

Для того, щоб видалити програму, користувачу достатньо видалити папку “the fan`s guide”.

3.3 Використання програми

Коли користувач відкриває програму, він бачить головне вікно з панеллю пошуку та таблицею для запису спортсменів. Інтерфейс головного вікна наведений на рисунку 3.1.

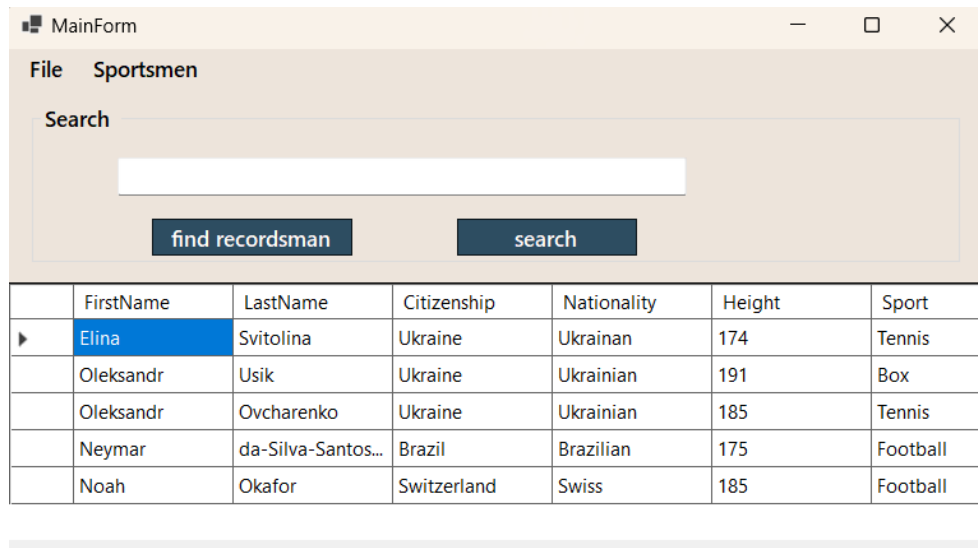


Рисунок 3.1 – Головне вікно програми

Для того щоб додати нового спортсмена, користувач повинен натиснути на “Sportsmen” у верхньому меню. Далі у випадному списку, інтерфейс якого показаний на рисунку 3.2, потрібно натиснути “Add”.

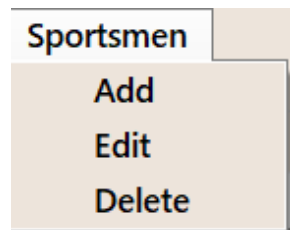


Рисунок 3.2 – Випадний список пункту “Sportsmen”

Після цього заповнити поля у формі, яка відкрилась. Приклад наведено на рисунку 3.3.

Рисунок 3.3 – Приклад заповнення даних для додавання спортсмена

Якщо користувач передумав додавати спортсмена, він може натиснути кнопку “Cancel” і вікно для додавання закриється. Для того щоб додати спортсмена необхідно натиснути кнопку “OK”, яка знаходиться внизу екрану. Після цього з’являється вікно з підтвердженням (рисунок 3.4).

Рисунок 3.4 – Вікно підтвердження

Далі потрібно натиснути кнопку “так” (мова написів на кнопках може відрізнятися в залежності від мови, яка встановлена на комп’ютері).

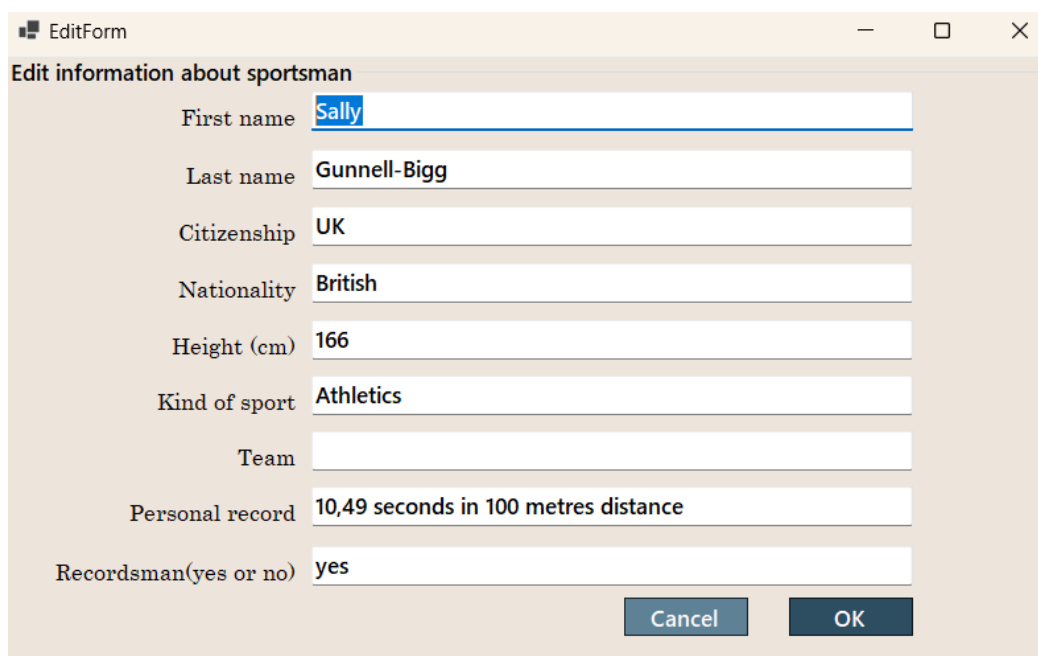
Після цього користувач бачить доданого спортсмена у кінці списку.

	FirstName	LastName	Citizenship	Nationality	Height	Sport
	Elina	Svitolina	Ukraine	Ukrainian	174	Tennis
	Oleksandr	Usik	Ukraine	Ukrainian	191	Box
	Oleksandr	Ovcharenko	Ukraine	Ukrainian	185	Tennis
	Neymar	da-Silva-Santos...	Brazil	Brazilian	175	Football
	Noah	Okafor	Switzerland	Swiss	185	Football
▶	Sally	Gunnell-Bigg	UK	British	166	Athletics

Рисунок 3.5 – Оновлений список після додавання

Для того, щоб змінити дані про вже існуючого спортсмена, користувач має обрати спортсмена натиснувши на нього правою кнопкою миші або за допомогою клавіатури. Потім натиснути на “Sportsmen” у верхньому меню. Далі у випадному списку, інтерфейс якого показаний на рисунку 3.2, потрібно натиснути “Edit”.

Після цього користувач може редагувати дані про обраного спортсмена, змінюючи текст у текстових полях. Зовнішній вигляд вікна редагування показаний на рисунку 3.6.



The screenshot shows a window titled "EditForm" with a standard Windows-style title bar (minimize, maximize, close buttons). The window contains a form titled "Edit information about sportsman". The form has several text input fields with labels to their left:

- First name: Sally
- Last name: Gunnell-Bigg
- Citizenship: UK
- Nationality: British
- Height (cm): 166
- Kind of sport: Athletics
- Team: (empty field)
- Personal record: 10,49 seconds in 100 metres distance
- Recordsman(yes or no): yes

At the bottom right of the form are two buttons: "Cancel" and "OK".

Рисунок 4.6 – Зовнішній вигляд вікна редагування

Після внесення необхідних змін, користувач має натиснути на кнопку “OK”. Далі, у вікні яке з’явилося (рисунок 3.7), підтвердити бажання змінити дані, натиснувши “Так”.

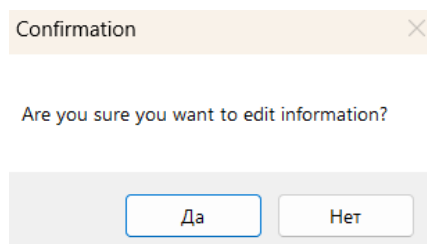


Рисунок 3.7 – Вікно підтвердження редагування

Далі користувач побачить головне вікно програми з оновленими даними.

Для того, щоб видалити спортсмена користувачу потрібно обрати спортсмена зі списку, потім він має натиснути на “Sportsmen” у верхньому меню. Далі у випадному списку, інтерфейс якого показаний на рисунку 3.2, потрібно натиснути “Delete”. Потім користувачу потрібно підтвердити своє бажання видалити спортсмена (інтерфейс вікна підтвердження наведений на рисунку 3.8) і він побачить оновлений список спортсменів.

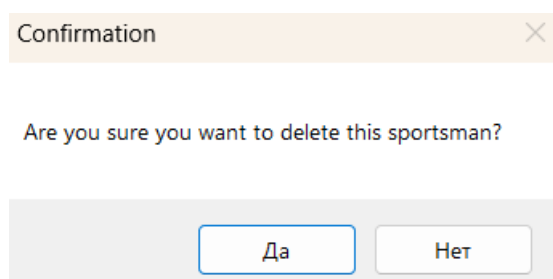
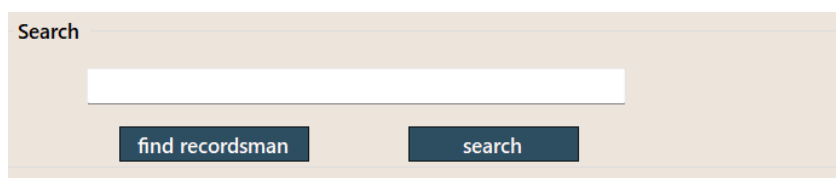


Рисунок 3.8 – Вікно підтвердження видалення

Для пошуку спортсмена за іменем, прізвищем або видом спорту користувач має ввести параметри у текстове поле, яке знаходиться на головній формі. Панель пошуку зображена на рисунку 3.9.



Search

Рисунок 3.9 – Панель пошуку

Після введення параметрів пошуку користувач має натиснути кнопку “search” і спортсмени, які задовольняють умови, з’являться у списку в нижній частині екрану. Приклад роботи пошуку наведений на рисунку 3.10.



File Sportsmen

Search

	FirstName	LastName	Citizenship	Nationality	Height	Sport
▶	Sally	Gunnell-Bigg	UK	British	166	Athletics

Рисунок 3.10 – Пошук спортсмена

Щоб повернутися назад до всіх спортсменів користувачу потрібно прибрати текст з панелі пошуку та натиснути кнопку “search”.

Для того щоб знайти рекордсмена в заданому виді спорту, користувачу потрібно ввести вид спорту у теж саме текстове поле як і при звичайному пошуку спортсмена. Після цього потрібно натиснути на кнопку “find recordsman”. Далі користувач побачить список рекордсменів у виді спорту, який він ввів.

Для того щоб повністю очистити дані в програмі, користувачу потрібно натиснути пункт “File” та обрати “Clear” у випадному списку. Зображення списку наведено на рисунку 3.11.

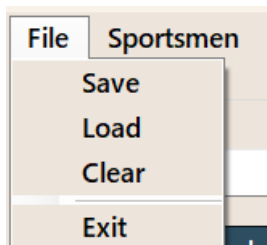


Рисунок 3.11 – Випадний список пункту “File”

Далі потрібно натиснути “Так” у вікні підтвердження і список спортсменів повністю очиститься. Зовнішній вигляд вікна підтвердження наведений на рисунку 3.12.

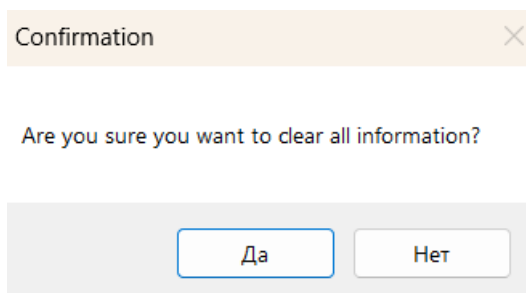


Рисунок 3.12 – Вікно підтвердження очищення

Для того щоб зберегти дані користувачу потрібно натиснути на “Save” у списку, що показаний на рисунку 3.11.

Для завантаження останніх збережених даних необхідно натиснути “Load” у списку на рисунку 3.11.

Для того щоб вийти з програми, користувачу потрібно натиснути “Exit” або хрестик, що знаходиться у правому верхньому куті програми.

ВИСНОВКИ

Розроблена програма відповідає заявленим цілям. Наявна база спортсменів з можливістю додавання, редагування та видалення даних. Також є пошук спортсмена за заданими вимогами та пошук рекордсмена за видом спорту. Є зберігання та завантаження даних.

Перед виконанням дій, які змінюють інформацію, програма запитує підтвердження в користувача. Наявні повідомлення - підказки у разі помилок при введенні даних. Інтерфейс зручний та легкий у використанні.

Програму можна було б вдосконалити більш ретельними перевітками форм додавання та редагування, також можна було б покращити дизайн застосунку.

Під виконання курсового проекту я покращила свої знання мови C#, навчилася працювати з Windows Forms та отримала досвід розробки програмного продукту.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Бондарев В.М. Електронний учбовий посібник з дисципліни «Основи програмування» URL: <http://tss.co.ua:5555/> (дата звернення: 02.05.2023)
2. Windows Forms documentation. URL: <https://learn.microsoft.com/en-us/dotnet/desktop/winforms/?view=netdesktop-7.0> (дата звернення: 29.04.2023)

ДОДАТОК А

Код програми

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using the_fan_s_guide.Data;
using the_fan_s_guide.Models;

namespace the_fan_s_guide.Forms
{
    public partial class MainForm : Form // Клас головної форми
    {
        SportBase sportbase;
        public MainForm()
        {
            InitializeComponent();
            sportbase = new SportBase();
            sportsmanBindingSource.DataSource = sportbase.Sportsmen;
        }

        private void MainForm_Load(object sender, EventArgs e)
        {
            DataAccess.Load(sportbase);
            sportsmanBindingSource.ResetBindings(true);
            sportsmanBindingSource.DataSource = sportbase.Sportsmen;
        }

        private void saveToolStripMenuItem_Click(object sender, EventArgs e)
        {
            var saveRes = MessageBox.Show("Do you want to save information?",
"Confirmation", MessageBoxButtons.YesNo);
            if (saveRes == DialogResult.Yes)
            {
                DataAccess.Save(sportbase);
            }
            if (saveRes == DialogResult.No)
            {
                return;
            }
        }

        private void loadToolStripMenuItem_Click(object sender, EventArgs e)
        {
            DataAccess.Load(sportbase);
            sportsmanBindingSource.ResetBindings(true);
            sportsmanBindingSource.DataSource = sportbase.Sportsmen;
        }

        private void clearToolStripMenuItem_Click(object sender, EventArgs e)
        {
            var clearRes = MessageBox.Show("Are you sure you want to clear all
information?", "Confirmation", MessageBoxButtons.YesNo);

```

```

        if (DialogResult.Yes == clearRes)
        {
            sportbase.Sportsmen.Clear();
            sportsmanBindingSource.ResetBindings(true);
            sportbase.Changed = true;
        }
        if (DialogResult == DialogResult.No)
        {
            return;
        }
    }

    private void addToolStripMenuItem_Click(object sender, EventArgs e)
    {
        var form = new AddForm(sportbase);
        var dialogResult = form.ShowDialog();
        if (dialogResult == DialogResult.OK)
        {
            sportbase.Sportsmen.Add(form.Sportsman);
            sportsmanBindingSource.ResetBindings(true);
            sportbase.Changed = true;
        }
    }

    private void editToolStripMenuItem_Click(object sender, EventArgs e)
    {
        var selectedRow = sportsmanGridView.CurrentRow;
        if (selectedRow == null)
        {
            return;
        }

        var selectedSportsman = selectedRow.DataBoundItem as Sportsman;

        if (selectedSportsman == null)
        {
            return;
        }

        var form = new EditForm(selectedSportsman);
        if (form.ShowDialog() == DialogResult.OK)
        {
            sportsmanBindingSource.ResetBindings(true);
            sportbase.Changed = true;
        }
    }

    private void exitToolStripMenuItem_Click(object sender, EventArgs e)
    {
        Close();
    }

    private void MainForm_FormClosing(object sender, FormClosingEventArgs e)
    {
        if (!sportbase.Changed)
        {
            return;
        }
        var result = MessageBox.Show("Do you want to save changes?",
"Confirmation", MessageBoxButtons.YesNoCancel);
        switch (result)
        {
            case DialogResult.Yes:

```

```

        DataAccess.Save(sportbase);
        break;
    case DialogResult.No:
        break;
    case DialogResult.Cancel:
        e.Cancel = true;
        break;
    }
}

private void searchButton_Click(object sender, EventArgs e)
{
    List<Sportsman> res = sportbase.SearchSportsmen(searchTextBox.Text);
    sportsmanBindingSource.DataSource = res;
}

private void deleteToolStripMenuItem_Click(object sender, EventArgs e)
{
    var selectedSportsman = sportsmanBindingSource.Current as Sportsman;

    if (selectedSportsman == null)
    {
        return;
    }
    var result = MessageBox.Show("Are you sure you want to delete this
sportsman?", "Confirmation", MessageBoxButtons.YesNo);
    switch (result)
    {
        case DialogResult.Yes:
            sportsmanBindingSource.Remove(selectedSportsman);
            sportbase.Changed = true;
            break;
        case DialogResult.No:
            break;
    }
    sportbase.Changed = true;
}

private void recordsmanButton_Click(object sender, EventArgs e)
{
    List<Sportsman> recordsmanRes =
sportbase.SearchRecordsman(searchTextBox.Text);
    sportsmanBindingSource.DataSource = recordsmanRes;
}
}

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace the_fan_s_guide.Models
{
    public class Sportsman // Клас спортсмена.
    {
        public string FirstName { get; set; } // Поле для імені спортсмена.
        public string LastName { get; set; } // Поле для прізвища спортсмена.
        public string Citizenship { get; set; } // Поле для громадянства спортсмена.
        public string Nationality { get; set; } // Поле для національності
спортсмена.
        public int Height { get; set; } // Поле для висоти спортсмена.
        public string Sport { get; set; } // Поле для виду спорту спортсмена.
    }
}

```

```

        public string Team { get; set; } // Поле для команди спортсмена.
        public string PersonalRecord { get; set; } // Поле особистого рекорду
спортсмена.
        public string Recordsman { get; set; } // Поле для визначення чи є
спортсмена рекордсменом.
    }
}

```

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

```

```

namespace the_fan_s_guide.Models
{

```

```

    public class SportBase // Клас для списку спортсменів.
    {
        public bool Changed { get; set; } // Поле для розуміння коли були змінені
дані.
        public SportBase()
        {
            Sportsmen = new List<Sportsman>();
        }
        public List<Sportsman> Sportsmen { get; set; } // Список спортсменів.

        internal List<Sportsman> SearchSportsmen(string text)
        {
            List<Sportsman> res = new List<Sportsman>();
            foreach (Sportsman sportsman in Sportsmen)
            {
                if (sportsman.FirstName.ToLower().IndexOf(text.ToLower()) > -1 ||
sportsman.LastName.ToLower().IndexOf(text.ToLower()) > -1 ||
sportsman.Sport.ToLower().IndexOf(text.ToLower()) > -1)
                {
                    res.Add(sportsman);
                }
            }
            return res;
        }

        internal List<Sportsman> SearchRecordsman(string sportLine)
        {
            List<Sportsman> recordsmanRes = new List<Sportsman>();
            foreach (Sportsman sportsman in Sportsmen)
            {
                If (sportsman.Sport.ToLower() == sportLine.ToLower() &&
sportsman.Recordsman.ToLower() == "yes")
                {
                    recordsmanRes.Add(sportsman);
                }
            }
            return recordsmanRes;
        }
    }
}

```

```

using System;
using System.Collections.Generic;

```

```

using System.Linq;
using System.Text;
using System.Text.Json;
using System.Threading.Tasks;
using the_fan_s_guide.Models;

namespace the_fan_s_guide.Data
{
    internal static class DataAccess
    {
        const string DATA_PATH = "SportBase.json";
        public static void Save(SportBase sportbase) // Метод для зберігання даних.
        {
            string jsonString = JsonSerializer.Serialize(sportbase.Sportsmen);

            File.WriteAllText(DATA_PATH, jsonString);
            sportbase.Changed = false;
        }

        public static void Load(SportBase sportbase) // Метод для завантаження
даних.
        {
            string jsonString = File.ReadAllText(DATA_PATH);

            var newSportsmen =
JsonSerializer.Deserialize<List<Sportsman>>(jsonString);
            sportbase.Sportsmen.Clear();
            sportbase.Sportsmen.AddRange(newSportsmen);
            sportbase.Changed = false;
        }
    }
}

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using the_fan_s_guide.Forms_Validation;
using the_fan_s_guide.Models;

namespace the_fan_s_guide.Forms
{
    public partial class AddForm : Form // Клас форми додавання спортсмена.
    {
        public Sportsman Sportsman;
        SportBase _sportBase;
        public AddForm()
        {
            InitializeComponent();
        }
        public AddForm(SportBase sportBase)
        {
            InitializeComponent();
            _sportBase = sportBase;
        }
        private void cancelButton_Click(object sender, EventArgs e)
        {
            this.DialogResult = DialogResult.Cancel;
        }
    }
}

```



```

        private void okButton_Click(object sender, EventArgs e)
        {
            string problemMessage =
FieldsValidation.GetProblemMessage(firstNameTextBox.Text, lastNameTextBox.Text,
citizenshipTextBox.Text, nationalityTextBox.Text, heightTextBox.Text,
sportTextBox.Text, teamTextBox.Text, persRecordTextBox.Text, recordsmanTextBox.Text);
            if (problemMessage != "")
            {
                MessageBox.Show(problemMessage, "Problems:");
                return;
            }
            Sportsman = new Sportsman()
            {
                FirstName = firstNameTextBox.Text,
                LastName = lastNameTextBox.Text,
                Citizenship = citizenshipTextBox.Text,
                Nationality = nationalityTextBox.Text,
                Height = Convert.ToInt32(heightTextBox.Text),
                Sport = sportTextBox.Text,
                Team = teamTextBox.Text,
                PersonalRecord = persRecordTextBox.Text,
                Recordsman = recordsmanTextBox.Text,
            };
            var addRes = MessageBox.Show("Are you sure you want to add this
sportsman?", "Confirmation", MessageBoxButtons.YesNo);
            if (addRes == DialogResult.Yes)
            {
                DialogResult = DialogResult.OK;
            }
            If (addRes== DialogResult.No)
            {
                return;
            }
        }
    }
}

```

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using the_fan_s_guide.Forms_Validation;
using the_fan_s_guide.Models;

namespace the_fan_s_guide.Forms
{
    public partial class EditForm : Form // Клас форми редагування даних спортсмена.
    {
        public Sportsman Sportsman;
        public EditForm()
        {
            InitializeComponent();
        }
        public EditForm(Sportsman sportsman)
        {
            InitializeComponent();
        }
    }
}

```

```

        Sportsman = sportsman;
        firstNameTextBox.Text = sportsman.FirstName;
        lastNameTextBox.Text = sportsman.LastName;
        citizenshipTextBox.Text = sportsman.Citizenship;
        nationalityTextBox.Text = sportsman.Nationality;
        heightTextBox.Text = sportsman.Height.ToString();
        sportTextBox.Text = sportsman.Sport;
        teamTextBox.Text = sportsman.Team;
        persRecordTextBox.Text = sportsman.PersonalRecord;
        recordsmanTextBox.Text = sportsman.Recordsman;
    }

    private void okButton_Click_1(object sender, EventArgs e)
    {
        string problemMessage =
FieldsValidation.GetProblemMessage(firstNameTextBox.Text, lastNameTextBox.Text,
citizenshipTextBox.Text, nationalityTextBox.Text, heightTextBox.Text,
sportTextBox.Text, teamTextBox.Text, persRecordTextBox.Text, recordsmanTextBox.Text);
        if (problemMessage != "")
        {
            MessageBox.Show(problemMessage, "Problems:");
            return;
        }
        Sportsman.FirstName = firstNameTextBox.Text;
        Sportsman.LastName = lastNameTextBox.Text;
        Sportsman.Citizenship = citizenshipTextBox.Text;
        Sportsman.Nationality = nationalityTextBox.Text;
        Sportsman.Height = Convert.ToInt32(heightTextBox.Text);
        Sportsman.Sport = sportTextBox.Text;
        Sportsman.Team = teamTextBox.Text;
        Sportsman.PersonalRecord = persRecordTextBox.Text;
        Sportsman.Recordsman = recordsmanTextBox.Text;
        var editRes = MessageBox.Show("Are you sure you want to edit
information?", "Confirmation", MessageBoxButtons.YesNo);
        if (editRes == DialogResult.Yes)
        {
            DialogResult = DialogResult.OK;
        }
        if (editRes == DialogResult.No)
        {
            return;
        }
    }

    private void cancelButton_Click_1(object sender, EventArgs e)
    {
        DialogResult = DialogResult.Cancel;
    }
}

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using the_fan_s_guide.Models;

namespace the_fan_s_guide.Forms_Validation
{
    public static class FieldsValidation // Клас для перевірки введених користувачем
даних на коректність.
    {
        public static string GetProblemMessage(string firstName, string lastName,
string citizenship, string nationality,

```

```

        string height, string sport, string team, string persRecord, string
recordsman) // Метод який використовується для повідомлень про помилки при вводити
даних.
    {
        var problemMessage = "";
        if (string.IsNullOrEmpty(firstName) || !OnlyLetters(firstName) ||
firstName.Length>50 && firstName.Length<1)
        {
            problemMessage += "First name cannot be empty and must contain only
less than 50 letters!\n";
        }
        if (string.IsNullOrEmpty(lastName) || !OnlyLetters(lastName) ||
lastName.Length>50)
        {
            problemMessage += "Last name cannot be empty and must contain only
less than 50 letters!\n";
        }

        if (citizenship.Length>50 )
        {
            problemMessage += "Citizenship cannot be empty and must contain only
less than 50 letters!\n";
        }

        if (string.IsNullOrEmpty(nationality) || !OnlyLetters(nationality) ||
nationality.Length>50)
        {
            problemMessage += "Nationality cannot be empty and must contain only
less than 50 letters!\n";
        }

        bool heightOk = int.TryParse(height, out int parsedHeight);
        if (!heightOk || parsedHeight <= 0 || height.Length >= 4)
        {
            problemMessage += "Height must by a positive number and must contain
only less than 4 symbols!\n";
        }

        if (string.IsNullOrEmpty(sport) || !OnlyLetters(sport) ||
sport.Length>50)
        {
            problemMessage += "Kind of sport cannot be empty and must contain
only less than 50 letters!\n";
        }

        if (team.Length > 100)
        {
            problemMessage += "Team must only contain less than 100 symbols!";
        }

        if (persRecord.Length > 100)
        {
            problemMessage += "Personal record must only contain less than 100
symbols!";
        }

        if (string.IsNullOrEmpty(recordsman) || !YesorNo(recordsman))
        {
            problemMessage += "Please, enter 'yes' or 'no'.";
        }
        return problemMessage;
    }

    public static bool OnlyLetters(string info) // Метод, який перевіряє чи всі
символи рядку є літерами.

```

```

    {
        foreach (char i in info)
        {
            if ((!char.IsLetter(i)) && i != '-')
            {
                return false;
            }
        }
        return true;
    }

    public static bool YesorNo(string input) // Метод, який перевіряє введені
дані у поле Recordsman.
    {
        string lowerInput = input.ToLower();
        if (lowerInput == "no")
        {
            return true;
        }
        if (lowerInput == "yes")
        {
            return true;
        }
        return false;
    }
}
}

```