МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»

Кафедра інженерії програмного забезпечення

**КУРСОВА РОБОТА**

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

з дисципліни: «Об’єктно-орієнтоване програмування»

на тему:

**«Електронний підручник для вивчення мови програмування»**

студента І курсу групи ВТ-23-1

спеціальності 121 «Інженерія програмного забезпечення»

\_\_\_\_\_Нагорного Тараса Григоровича\_\_\_\_\_

(прізвище, ім’я та по-батькові)

Керівник: старший викладач кафедри ІПЗ

Дмитро ПРОХОРЧУК ,

Дата захисту: " \_\_\_ " \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ р.

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оцінка: ECTS \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Члени комісії \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Тетяна ВАКАЛЮК

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Олексій ЧИЖМОТРЯ

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Олег ВЛАСЕНКО

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Дмитро ПРОХОРЧУК

(підпис) (прізвище та ініціали)

Житомир – 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»

Факультет інформаційно-комп’ютерних технологій

Кафедра інженерії програмного забезпечення

Освітній рівень: бакалавр

Спеціальність 121 «Інженерія програмного забезпечення»

«ЗАТВЕРДЖУЮ»

Завідувач кафедри ІПЗ

\_\_\_\_\_\_\_\_\_\_\_\_ Тетяна ВАКАЛЮК

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2024 р.

ЗАВДАННЯ

НА КУРСОВУ РОБОТУ СТУДЕНТУ

Нагорному Тарасу Григоровичу

1. Тема роботи: розробка електронного підручника для вивчення мови програмування, керівник роботи: ст. викладач кафедри ІПЗ ПРОХОРЧУК Дмитро
2. Строк подання студентом: « 19 » травня 2024 р.
3. Вихідні дані до роботи: Розробити програму для вивчення мови програмування зі збереженням прогресу навчання та переглядом навчального матеріалу.
4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці)
   * + 1. Постановка завдання
       2. Аналіз аналогічних розробок
       3. Алгоритми роботи програми

4. Опис роботи програми

5. Програмне дослідження

1. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

1. Презентація до КР\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. Посилання на репозиторій: https://git.ztu.edu.ua/vt231\_ntg/university/-/tree/master/OOP/course\_work

1. Консультанти розділів роботи

|  |  |  |  |
| --- | --- | --- | --- |
| Розділ | Прізвище, ініціали та посади консультанта | Підпис, дата | |
| завдання  видав | завдання прийняв |
| 1-3 | ПРОХОРЧУК Дмитро | 04.03.2024 | 04.03.2024 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. Дата видачі завдання “ 04 ” березня 2024 р.

|  |  |  |  |
| --- | --- | --- | --- |
| № з/п | Назва етапів курсового проєктування | Строк виконання етапів роботи | Примітки |
| 1 | Постановка задачі | 05.03.2024 | виконано |
| 2 | Пошук, огляд та аналіз аналогічних розробок | 07.03.2024 | виконано |
| 3 | Формулювання технічного завдання | 12.03.2024 | виконано |
| 4 | Опрацювання літературних джерел | 18.03.2024 | виконано |
| 5 | Проектування структури | 22.03.2024 | виконано |
| 6 | Написання програмного коду | 05.04.2024 | виконано |
| 7 | Відлагодження | 07.05.2024 | виконано |
| 8 | Написання пояснювальної записки | 15.05.2024 | виконано |
| 9 | Захист | 06.06.2024 |  |

**КАЛЕНДАРНИЙ ПЛАН**

**Студент** \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Тарас НАГОРНИЙ

(підпис) (прізвище та ініціали)

**Керівник роботи** \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Дмитро ПРОХОРЧУК

(підпис) (прізвище та ініціали)

РЕФЕРАТ

Завданням на курсовий проект було створення додатку на тему «електронний підручник для вивчення мови програмування».

Пояснювальна записка до курсової роботи на тему «електронний підручник для вивчення мови програмування» складається з вступу, трьох розділів, висновків, списку використаної літератури та додатків.

Текстова частина викладена на 28 сторінках друкованого тексту. Пояснювальна записка має 12 сторінок додатків.

Список використаних джерел містить 9 найменувань і займає 1 сторінку. В роботі наведено 21 рисунків. Загальний обсяг роботи – 47 сторінки.

Ключові слова: C#, ПІДРУЧНИК, PYTHON, JSON, АЛГОРИТМ, КЛАС.

ЗМІСТ

[ВСТУП 3](#_Toc176015802)

[РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ 4](#_Toc176015803)

[1.1 Аналіз задачі, засобів та методів її вирішення 4](#_Toc176015804)

[1.2 Аналіз існуючого програмного забезпечення за тематикою курсової роботи 5](#_Toc176015805)

[Висновки до першого розділу 11](#_Toc176015806)

[РОЗДІЛ 2 ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 12](#_Toc176015807)

[2.1 Проектування загального алгоритму роботи програми 12](#_Toc176015808)

[2.2 Розбробка функціональних алгоритмів роботи програми 14](#_Toc176015809)

[2.3 Розробка програмного забезпечення 15](#_Toc176015810)

[Висновки до другого розділу 19](#_Toc176015811)

[РОЗДІЛ 3 ОПИС РОБОТИ З ПРОГРАМНИМ ДОДАТКОМ ТА ЙОГО ТЕСТУВАННЯ 20](#_Toc176015812)

[3.1 Опис роботи з програмним додатком (Опис інтерфейсу) 20](#_Toc176015813)

[3.2 Тестування програмного забезпечення 26](#_Toc176015814)

[Висновки до третього розділу 28](#_Toc176015815)

[ВИСНОВКИ 30](#_Toc176015816)

[СПИСОК ЛІТЕРАТУРИ 31](#_Toc176015817)

[ДОДАТКИ 32](#_Toc176015818)

[progresJson.cs 33](#_Toc176015819)

[Json.cs 34](#_Toc176015820)

[Form1.cs 34](#_Toc176015821)

# ВСТУП

**Актуальність теми.** Актуальність електронного підручника для вивчення мови програмування є актуальним у зв’язку зі зростом попиту на технічні спеціальності у сфері комп’ютерних технології. Наявність такого рішення допоможе інтерактивно вивчити матеріал та добре його засвоїти за допомогою тестів. Також, таке рішення допоможе структурувати наявні знання та детальніше вивчити вже відомий матеріал.

**Об’єкт дослідження.** Об’єктом дослідження є розробка електронного підручника для вивчення мови програмування Python 3.12 з використанням середовища розробки Vusial Studio та мови програмування C#.

**Предмет дослідження.** Предметом дослідження є пошук та створення алгоритмів, навчального матеріалу та тестів для розробки застосунку з функціями перегляду навчального матеріалу, проходження тестів, збереження прогресу навчання та перегляду прогресу навчанння.

**Мета дослідження.** Метою дослідження є розробка застосунку, що дозволить початківцям, що хочуть освоїти мову програмування Python 3.12, вивчити основи, що необхідні для написання базових програм цією мовою.

# РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ

## 1.1 Аналіз задачі, засобів та методів її вирішення

Для розробки застосунку я планую використовувати популярне та потужне середовище розробки Visual Studio, яке забезпечує всі необхідні інструменти для ефективної роботи. Основною мовою програмування буде обрана C#, яка є дуже поширеною та потужною мовою, що дозволяє створювати високоякісні додатки. Інтерфейс користувача (GUI) я реалізую за допомогою Windows Forms, що дає можливість не тільки засвоїти теоретичні знання з курсу об'єктно-орієнтованого програмування (ООП), але й на практиці застосувати їх для створення зручного та інтуїтивно зрозумілого інтерфейсу.

Для відображення навчальних матеріалів я оберу віджет WebView, який дозволить інтегрувати в застосунок можливість перегляду підготовлених заздалегідь веб-сторінок, створених у форматі HTML-файлів. Це забезпечить гнучкість та легкість у подачі інформації користувачеві.

Щодо збереження прогресу користувача, я оберу роботу з JSON-файлами. Це рішення обумовлене тим, що JSON має просту структуру, з якою легко працювати, що забезпечує надійне зберігання даних без надмірного ускладнення процесу.

Для організації зручної навігації всередині застосунку, буде використано віджет TabView, що дозволить легко переходити між різними розділами програми, забезпечуючи користувачеві максимально комфортний досвід взаємодії з програмою.

## 1.2 Аналіз існуючого програмного забезпечення за тематикою курсової роботи

Є безліч подібних рішень, проте розглянемо ті, що мають одні з найкращих відгуків серед користувачів.

**Sololearn**

Цей застосунок є на мобільні платформи та є у вигляді вебсайту. Розглянемо веб версію. Ця платформа має безліч різноманітних курсів, які можна вивчати паралельно

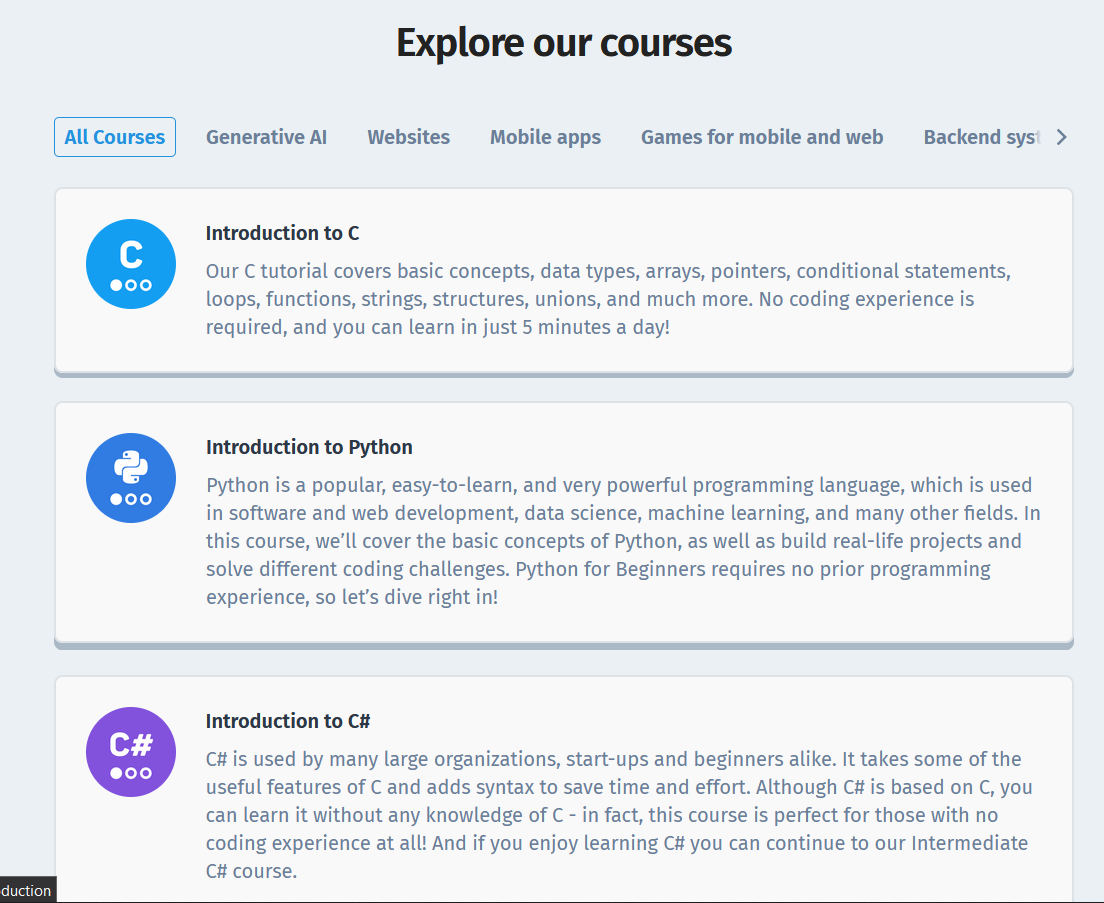


Рис. 1.1 – список доступних курсів на sololearn

Тут можна обрати курс для вивчення та вивчати його модуль за модулем. У кожного модуля є уроки. Коли користувач проходить урок, він спочатку ознайомлюється з теоретичним матеріалом та після проходить тестування. Якщо користувач не оформив преміум підписку, після кожного невдалого проходження тесту у нього зменшується доступна кількість спроб для проходження подальших тестів. За замовчуванням він має 5 спроб та з часом ці спроби поновлюються до 5. Також користувач може пропустити проходження тестів, які він не бажає пройти, за внутрішню валюту, яку отримує у процесі проходження уроків.

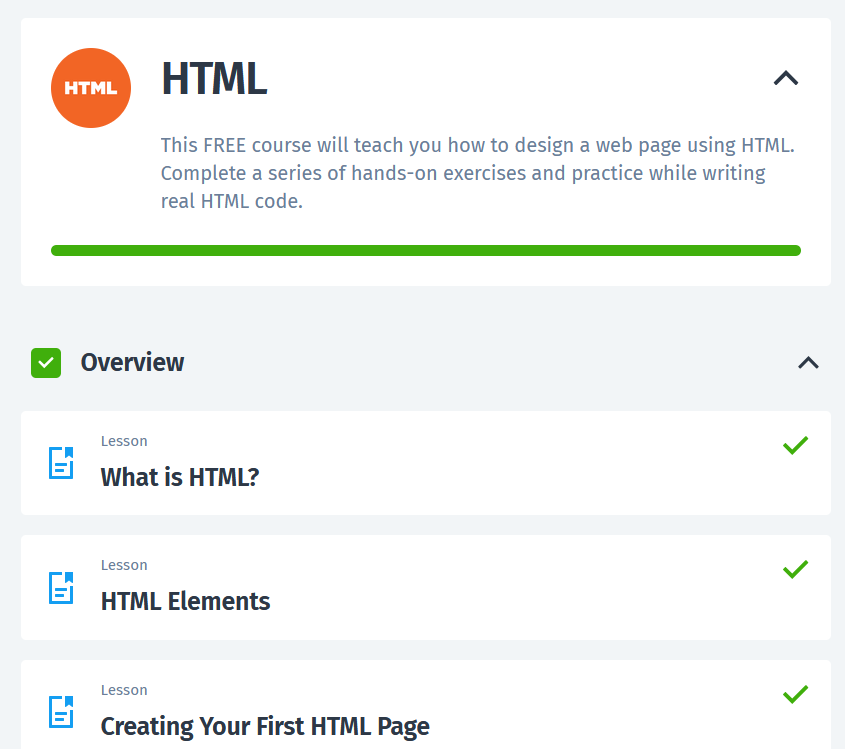


Рис. 1.2 – приклад доступного курсу на sololearn

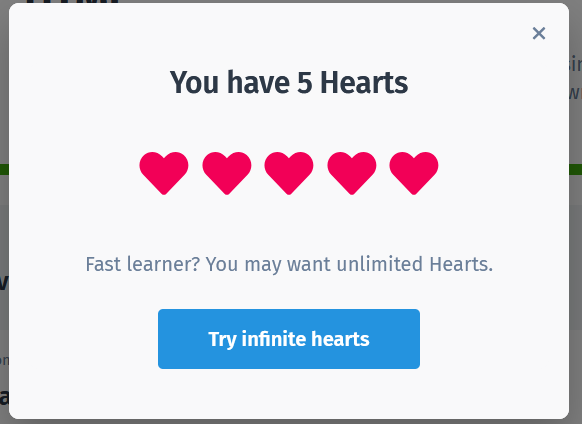


Рис. 1.3 – залишок доступних спроб для проходження тестів на sololearn

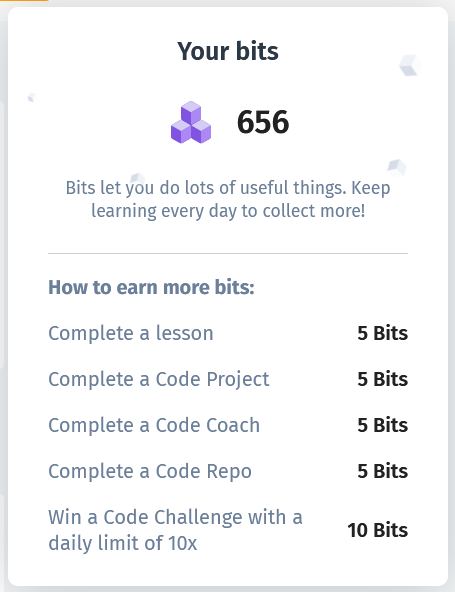


Рис. 1.4 – залишок доступної внутрішньої валюти для пропуску тестів

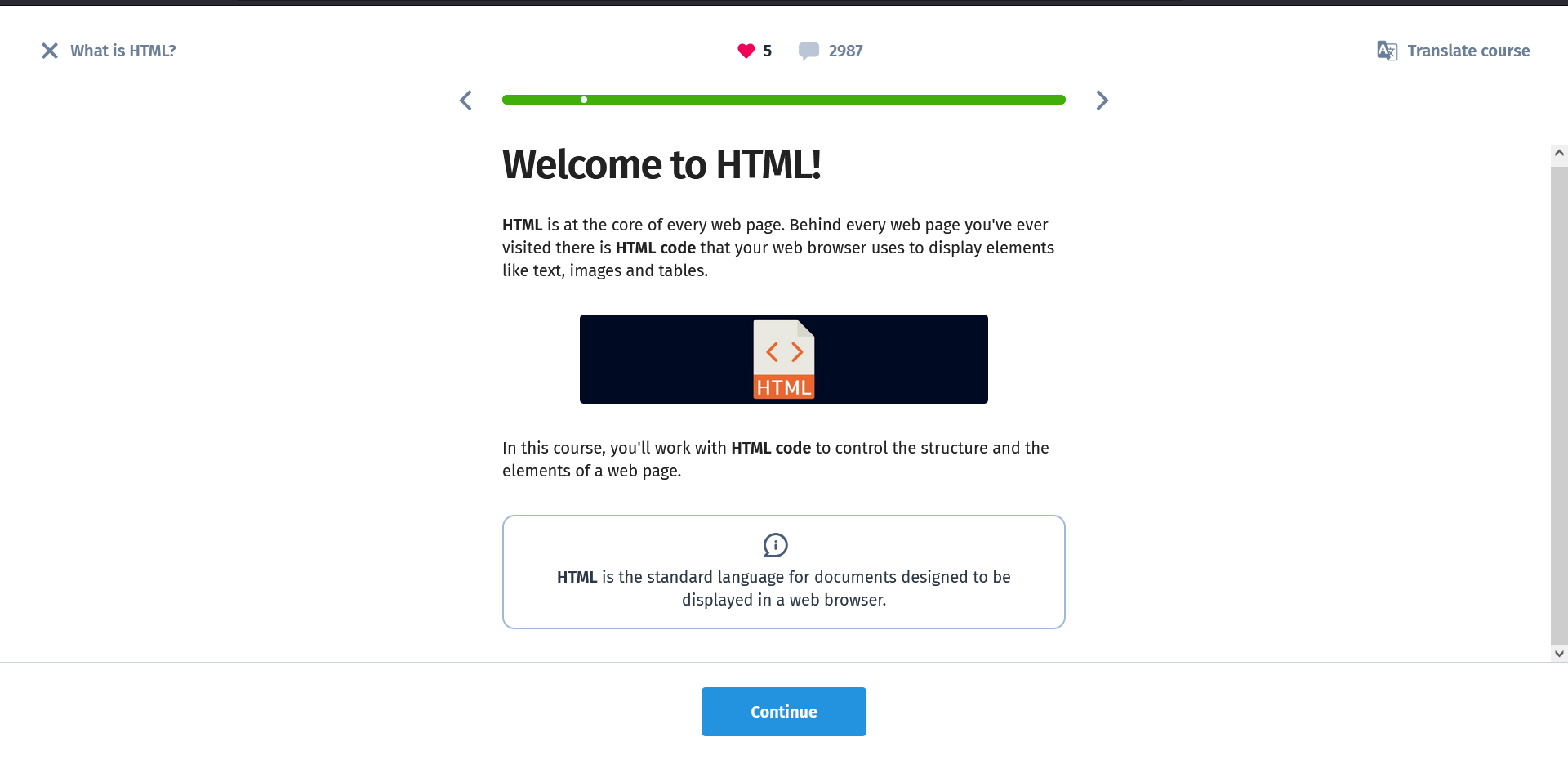


Рис. 1.5 – приклад навчального матеріалу під час проходження уроку на sololearn

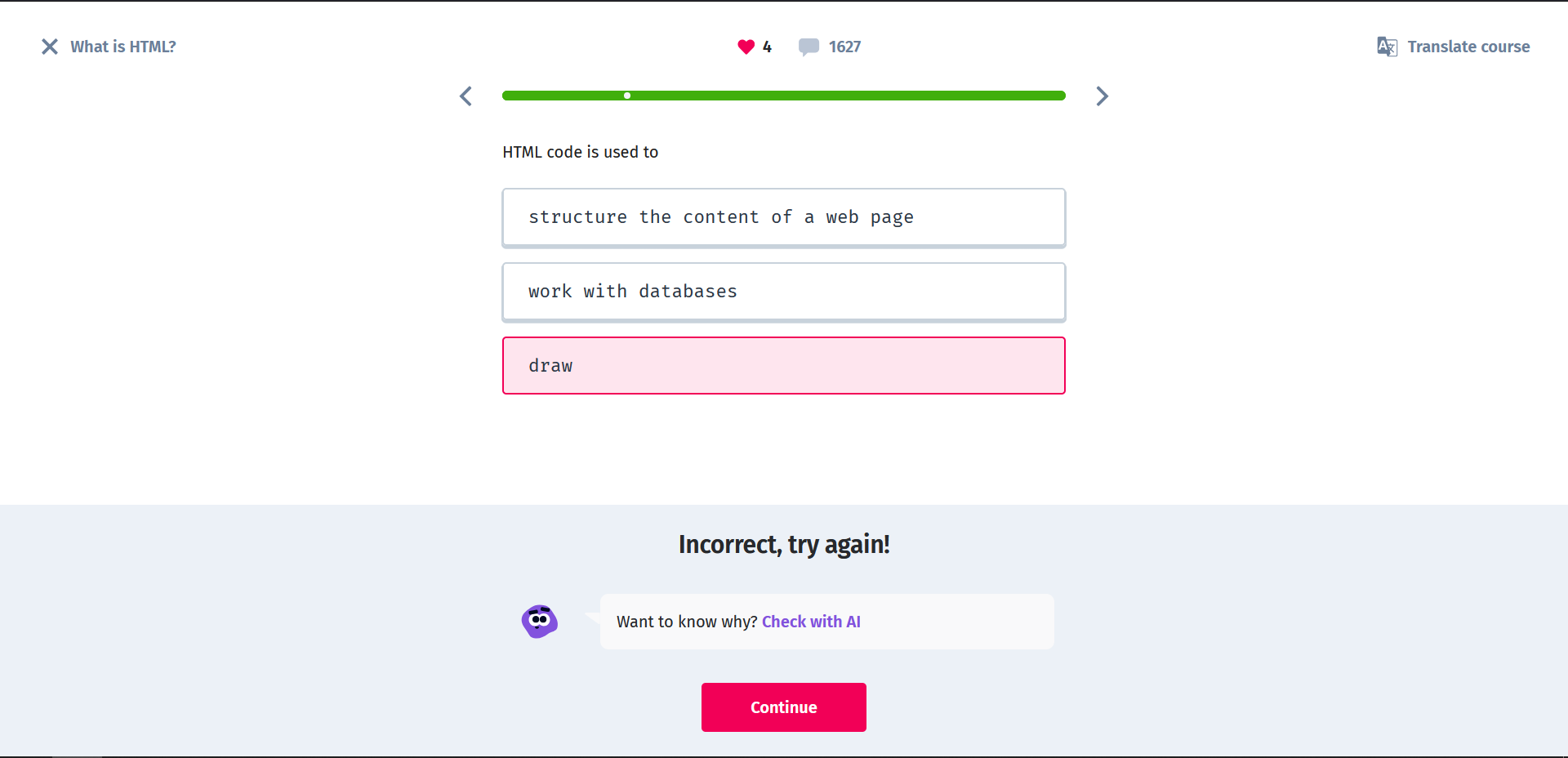


Рис. 1.6 – приклад проходження тесту під час проходження уроку на sololearn

Переваги:

* Доступний з будь-якого девайсу
* Багатомовний
* Велика кількість курсів
* Інтерактивні тести в кожному урокі
* Структурована інформація

Недоліки:

* Платний преміум функціонал
* Відсутня повна українська локалізація на всіх курсах
* Можливість проходження курсів лише до 5 помилок, після – потрібно зачекати декілька годин для поновлення спроб

**Udemy**

Це схожа до Sololearn платформа, проте основною відмінністю є те, що тут є безліч різноманітних курсів, тобто можна вивчати не лише курси, пов’язані з програмуванням, а навіть курси з кулінарії чи менеджменту. Це міжнародна платформа, за рахунок чого тут є безліч курсів різних напрямків, проте більшість з них англомовна, що є певним мінусом. Основною особливістю є те, що матеріал тут викладений у вигляді відеороликів, де користувач переглядає відео за відео, модуль за модулем

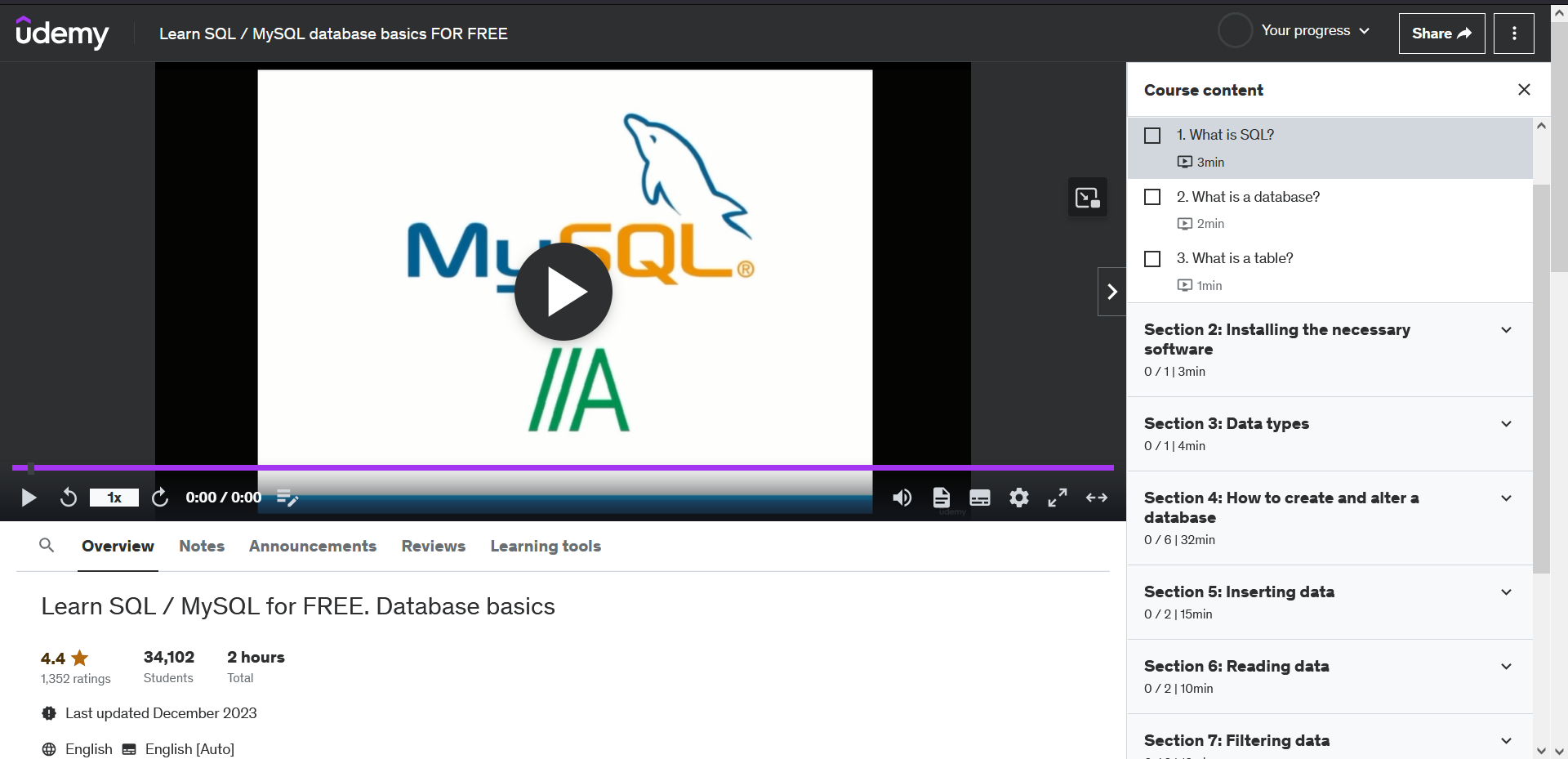


Рис. 1.7 приклад проходження курсу на Udemy

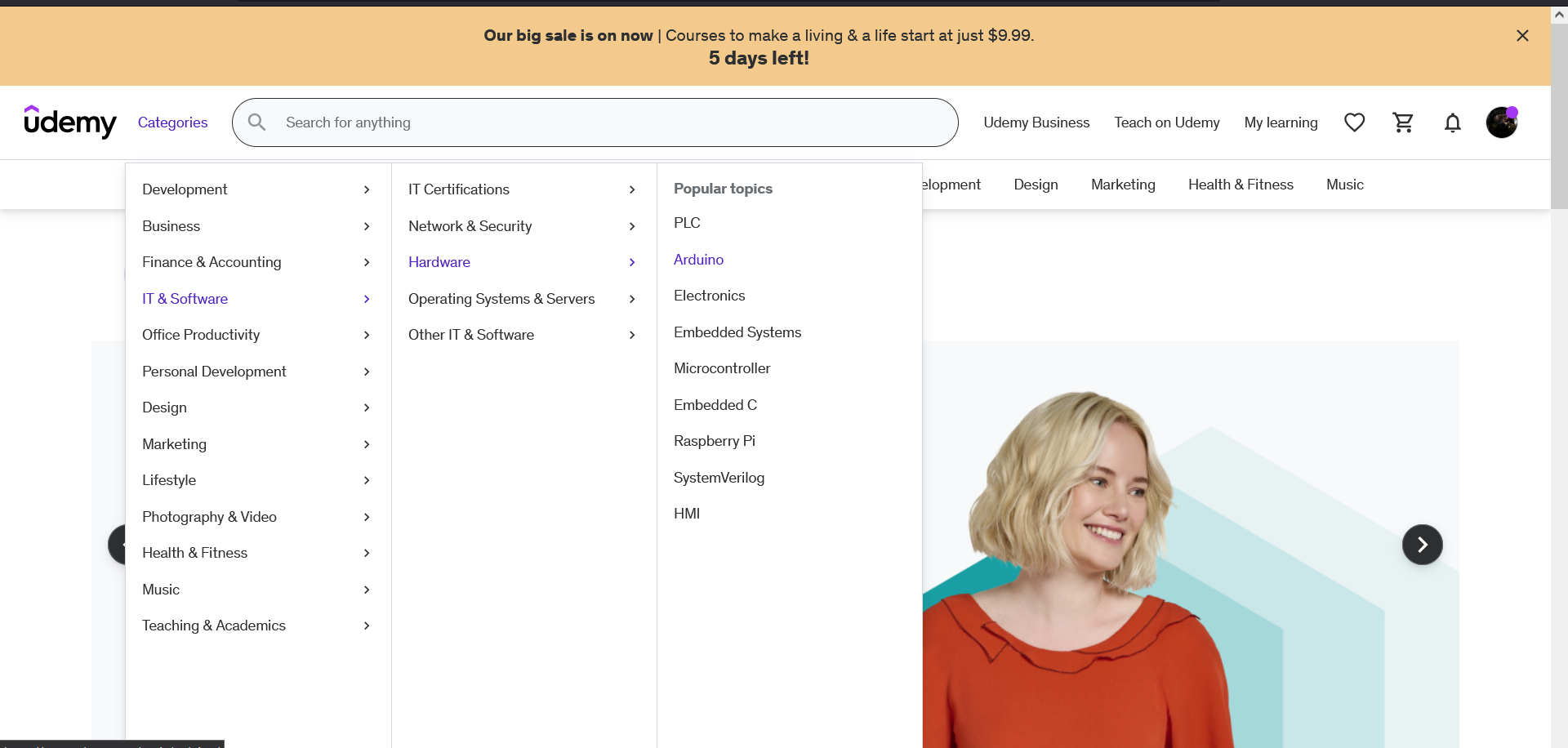


Рис. 1.8 приклад пошуку курсів для вивчення роботи з Arduino на Udemy

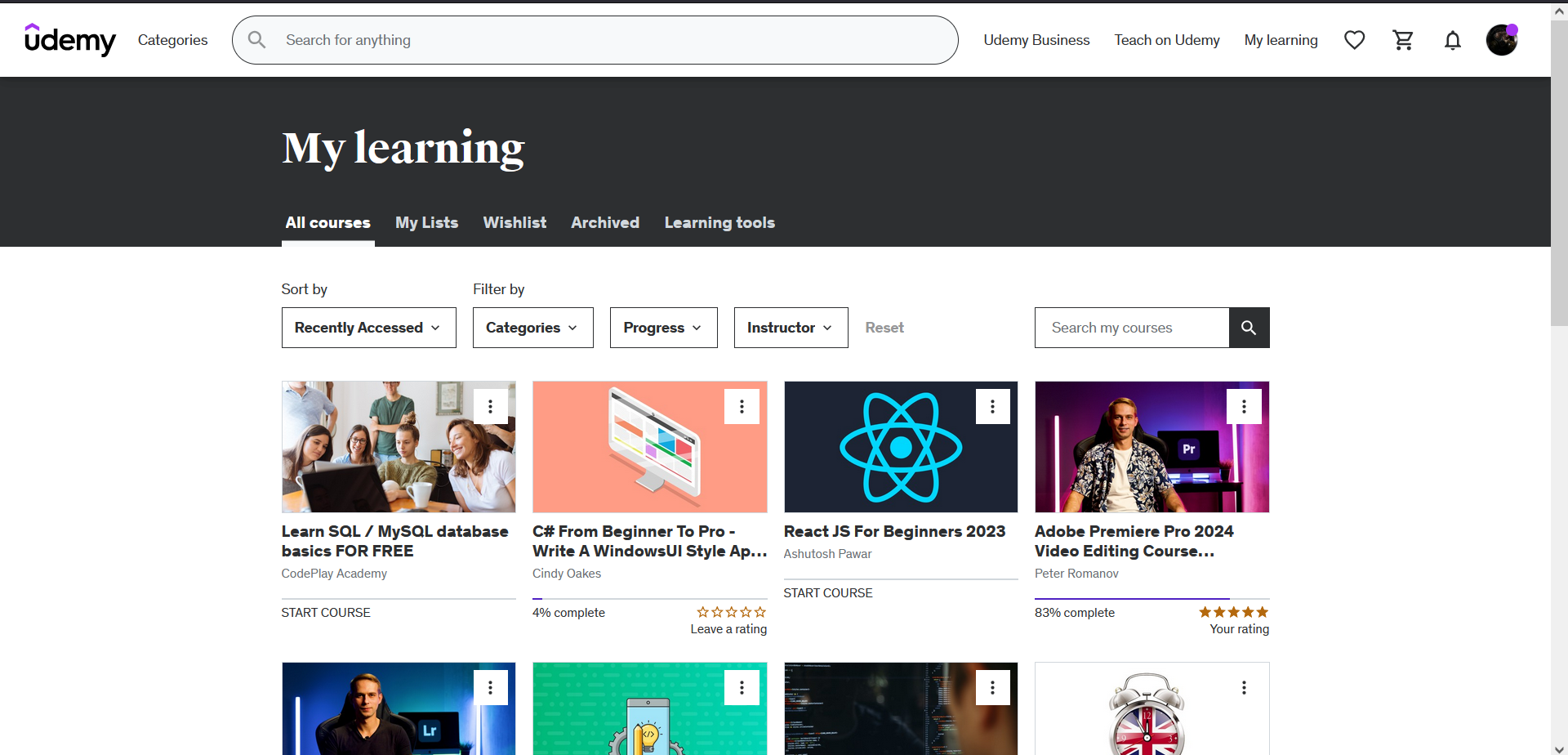


Рис. 1.9 приклад списку курсів, які може пройти користувач

Переваги:

* Безліч різноманітних курсів
* Перегляд курсів у вигляді відео, які можна переглядати безліч разів
* Наявність безкоштовних курсів
* Доступ з будь-якого куточку світу
* Структурована інформація

Недоліки:

* Більшість курсів виключно англомовні
* Більшість курсів платні
* Відсутність тестів після кожного з відеоуроків
* Відсутнє ком’юніті людей, що вивчають тей самий курс

Проаналізувавши ці готові рішення я розумію, що користувачу було б зручно, аби зберігався його прогрес, він міг повернутись до раніше пройденого матеріалу для повторення та щоб матеріл був викладений зрозумілою мовою.

## Висновки до першого розділу

У цьому розділі було проведено всебічний та детальний аналіз наявних рішень, які широко використовуються в аналогічних проектах. Під час цього аналізу було знайдено ключові моменти, які є критично важливими для успішної реалізації проекту та забезпечення максимального комфорту користувачів. Особлива увага була приділена вивченню того, як можна оптимально реалізувати різні механіки, враховуючи вже наявний досвід та рішення, що були успішно застосовані в інших проектах. Це дозволяє не лише використати найкращі практики, але й уникнути можливих помилок та недоліків, з якими могли зіткнутися попередні розробники.

# РОЗДІЛ 2 ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## 2.1 Проектування загального алгоритму роботи програми

Алгоритм роботи додатку побудований таким чином, щоб забезпечити користувачеві зручний та інтуїтивно зрозумілий процес навчання, при цьому зберігаючи логіку послідовного засвоєння матеріалу. При першому запуску програми користувач автоматично направляється до вивчення першого модулю. Цей початковий етап є важливим, оскільки він дозволяє закласти базові знання, необхідні для подальшого просування в навчанні.

Користувач має можливість вільно переходити між модулями, і при кожному такому переході додаток фіксує, що користувач уже проходив відповідний урок або тест. Це означає, що при повторному поверненні до вже завершеного модулю користувач зможе безперешкодно переглядати пройдені уроки та тести, що створює комфортні умови для повторення і закріплення матеріалу. Важливо зазначити, що система збереження прогресу дозволяє користувачеві вільно оперувати навчальним контентом, знаючи, що вся його діяльність надійно зафіксована в додатку.

Однак, якщо користувач не завершив якийсь із уроків або тестів, програма в автоматичному режимі блокуватиме доступ до наступного етапу навчання в межах поточного модулю, доки не буде виконано всі необхідні завдання. Це гарантує, що користувач не зможе просуватися вперед, поки не засвоїть попередній матеріал, що забезпечує послідовність і логіку в навчальному процесі.

Крім того, користувачеві надано можливість у будь-який момент перейти на сторінку з прогресом навчання, де він зможе побачити свої досягнення та аналізувати результати пройдених тестів. Також передбачено спеціальну сторінку з інформацією про сам додаток, де можна знайти деталі щодо його створення та авторства. Це робить додаток не лише навчальним інструментом, але й зручним засобом для моніторингу свого прогресу та загального орієнтування в програмі.

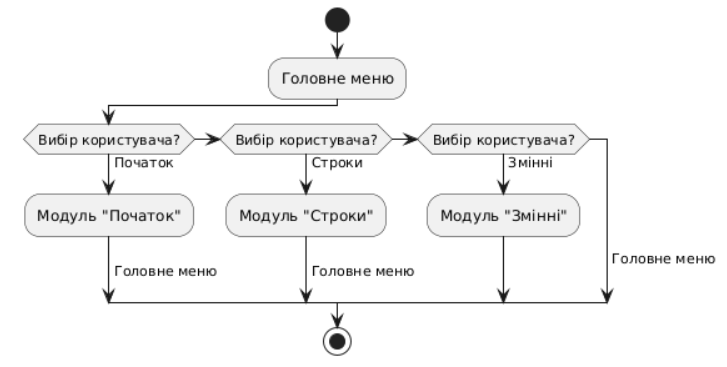


Рис. 2.1 Загальний алгоритм роботи програми

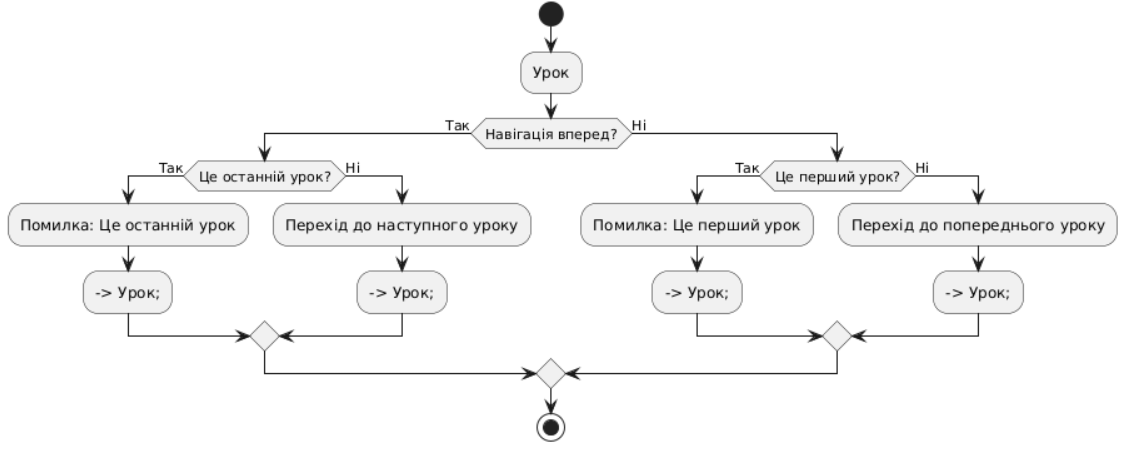


Рис. 2.2 Алгоритм навігації під час проходження уроків

## 2.2 Розбробка функціональних алгоритмів роботи програми

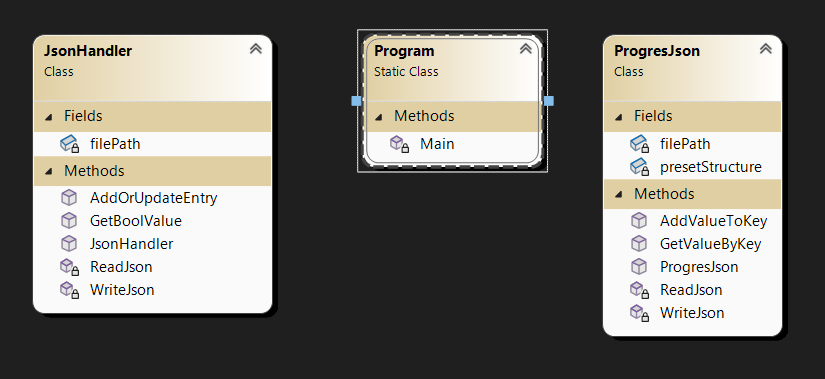


Рис. 2.3 Діаграма класів програми

На рисунку 12 зображена детальна діаграма класів, яка відображає основну архітектуру додатку, зокрема взаємодію між різними його компонентами. Центральне місце на цій діаграмі займає клас **Program**, який виконує роль базового класу для всієї програми. Саме з цього класу здійснюється запуск додатку, що означає, що він відповідає за початкову ініціалізацію всієї системи. Крім того, **Program** реалізує ключові аспекти користувацького інтерфейсу (UI), а також забезпечує коректну обробку подій, пов'язаних з віджетами, що взаємодіють з користувачем.

Далі, важливу роль у додатку відіграє клас **JsonHandler**, який призначений для управління збереженням інформації про навчальний прогрес користувача. Цей клас використовується для фіксації даних про те, чи проходив користувач той чи інший урок. Він надає можливість записувати ці дані у форматі JSON, що забезпечує зручність їх збереження та подальшого використання. У файлі, створеному за допомогою цього класу, фіксуються відомості про конкретний модуль та урок, а також булевий показник, який чітко вказує на те, чи був цей урок завершений користувачем. Таким чином, **JsonHandler** виконує важливу функцію, дозволяючи користувачеві повертатися до свого прогресу, не втрачаючи жодної важливої деталі.

Окремо варто виділити клас **ProgresJson**, який за своєю суттю схожий на **JsonHandler**, але його функціонал більш сфокусований на збереженні інформації про навчальні досягнення користувача, зокрема успішність проходження тестів. Цей клас фіксує всі результати тестувань, включаючи як кількість успішних, так і невдалих спроб. Ця інформація є вкрай важливою, оскільки в подальшому вона використовується для створення статистики, яка наочно демонструє успіхи користувача. Завдяки цьому, користувач може легко оцінити свій прогрес, побачити, де йому вдалося досягти високих результатів, а також визначити ті області, над якими ще варто попрацювати.

Таким чином, ця діаграма класів є ключовим елементом для розуміння внутрішньої структури додатку та того, як саме реалізовано управління даними в ньому.

## 2.3 Розробка програмного забезпечення

Застосунок включатиме в собі наступне:

Віджети TabControl: ці віджети використовуються для здійснення навігації в додатку. Можна розмістити UI у кожній вкладці та при натисненні користувачем кнопки переключати вміст на потрібну вкладку.

Віджет WebView: задля пришвидшення розробки використано віджет для показу зарані підготовлених HTML сторінок під час проходження уроків. Ось фрагмент коду, що завантажує потрібну сторінку:

private void loadHTMLInWebView(int tabIndex)

{

// якщо вибрано першу вкладку, де знаходиться модуль «Початок»

if (tabControl.SelectedIndex == 0)

{

// якщо вибрано перший урок

if (tabIndex == 0)

{

// віджету веб вʼю у віджеті Panel задаю властивість заповнити ввесь простір

webBrowser.Dock = DockStyle.Fill;

// додаю веб вʼю до конкретного Panel

startTabwebView.Controls.Add(webBrowser);

// прописую шлях до потрібного html файлу

string htmlFilePath = [$@"{baseDir}\beginning\_0.html](mailto:$@%22%7bbaseDir%7d\beginning_0.html)";

// відкриваю веб сторінку за шляхом

webBrowser.Navigate(htmlFilePath);

}

// якщо вибрано перший тест (друга вкладка у модулі «Початок»)

else if (tabIndex == 2)

{

webBrowser.Dock = DockStyle.Fill;

startTab2webView.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\beginning\_1.html";

webBrowser.Navigate(htmlFilePath);

}

else if (tabIndex == 4)

{

webBrowser.Dock = DockStyle.Fill;

startTab4webView.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\beginning\_2.html";

webBrowser.Navigate(htmlFilePath);

}

}

// модуль «Змінні»

else if (tabControl.SelectedIndex == 1)

{

if (tabIndex == 0)

{

webBrowser.Dock = DockStyle.Fill;

VariableWebViewPanel.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\variable\_0.html";

webBrowser.Navigate(htmlFilePath);

}

else if (tabIndex == 2)

{

webBrowser.Dock = DockStyle.Fill;

VariableWebViewPanel2.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\variable\_1.html";

webBrowser.Navigate(htmlFilePath);

}

else if (tabIndex == 4)

{

webBrowser.Dock = DockStyle.Fill;

VariableWebViewPanel4.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\variable\_2.html";

webBrowser.Navigate(htmlFilePath);

}

}

// модуль «Строки»

else if (tabControl.SelectedIndex == 2)

{

if (tabIndex == 0)

{

webBrowser.Dock = DockStyle.Fill;

StringWebViewPanel.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\string\_0.html";

webBrowser.Navigate(htmlFilePath);

}

else if (tabIndex == 2)

{

webBrowser.Dock = DockStyle.Fill;

StringWebViewPanel2.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\string\_1.html";

webBrowser.Navigate(htmlFilePath);

}

else if (tabIndex == 4)

{

webBrowser.Dock = DockStyle.Fill;

StringWebViewPanel4.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\string\_2.html";

webBrowser.Navigate(htmlFilePath);

}

}

}

Клас JsonHandler: клас, який використовується для збереження уже пройдених користувачем уроків. Вихідний код класу:

class JsonHandler

{

private string filePath;

public JsonHandler(string filePath)

{

this.filePath = filePath;

if (!File.Exists(filePath))

{

File.WriteAllText(filePath, "{}");

}

}

public void AddOrUpdateEntry(string key, bool value)

{

var jsonData = ReadJson();

jsonData[key] = value;

WriteJson(jsonData);

}

private Dictionary<string, bool> ReadJson() // метод для читання вмісту json файлу

{

var jsonContent = File.ReadAllText(filePath); // зберігаю у змінну вміст файлу

return JsonConvert.DeserializeObject<Dictionary<string, bool>>(jsonContent); // конвертую в обʼєкт Dictionary та повертаю

}

private void WriteJson(Dictionary<string, bool> jsonData) // записую Json у файл

{

var jsonContent = JsonConvert.SerializeObject(jsonData, Newtonsoft.Json.Formatting.Indented); // конвертую в Json обʼєкт

File.WriteAllText(filePath, jsonContent); // записую у файл

}

public bool GetBoolValue(string key) // у Json файлі отримую значення за ключем типу bool

{

var jsonData = ReadJson();

if (jsonData.ContainsKey(key)) // повертаю true лише якщо ключ існує та значення за ключем істина

{

return jsonData[key];

}

return false;

}

}

Клас ProgresJson: використовується для збереження прогресу під час проходження тестів (кількість успішних та хибних відповідей). Значна частина вихідного коду взята з попереднього класу, тому деякі пояснення упускаю. Вихідний код класу:

class ProgresJson

{

private string filePath;

private Dictionary<string, int> presetStructure = new Dictionary<string, int>

{

{ "beginning\_test\_0\_good", 0 },

{ "beginning\_test\_0\_bad", 0 },

{ "beginning\_test\_1\_good", 0 },

{ "beginning\_test\_1\_bad", 0 },

{ "string\_test\_0\_good", 0 },

{ "string\_test\_0\_bad", 0 },

{ "string\_test\_1\_good", 0 },

{ "string\_test\_1\_bad", 0 },

{ "variable\_test\_0\_good", 0 },

{ "variable\_test\_0\_bad", 0 },

{ "variable\_test\_1\_good", 0 },

{ "variable\_test\_1\_bad", 0 }

};

public ProgresJson(string filePath)

{

this.filePath = filePath;

if (!File.Exists(filePath))

{

WriteJson(presetStructure);

}

}

public int GetValueByKey(string key) // тут отримую кількість пройдених тестів (успішно або ні) за ключем

{

var jsonData = ReadJson();

if (jsonData.ContainsKey(key))

{

return jsonData[key];

}

throw new KeyNotFoundException($"Ключ '{key}' не знайдено у JSON файлі."); // викликаю виключення (у попередньому класі не реалізовував бо не було потреба, впринципі як і тут)

}

public void AddValueToKey(string key, int value) // тут я додаю до успішної чи не успішної кількості пройдених тестів одиницю

{

var jsonData = ReadJson();

if (jsonData.ContainsKey(key))

{

jsonData[key] += value;

WriteJson(jsonData);

}

else

{

throw new KeyNotFoundException($"Ключ '{key}' не знайдено у JSON файлі.");

}

}

private Dictionary<string, int> ReadJson()

{

var jsonContent = File.ReadAllText(filePath);

return JsonConvert.DeserializeObject<Dictionary<string, int>>(jsonContent);

}

private void WriteJson(Dictionary<string, int> jsonData)

{

var jsonContent = JsonConvert.SerializeObject(jsonData, Newtonsoft.Json.Formatting.Indented);

File.WriteAllText(filePath, jsonContent);

}

}

HTML та PNG файли: зарані підготовлений навчальний матеріал для показу користувачу за допомогою WebView під час проходження уроків.

## Висновки до другого розділу

У цьому розділі було детально розглянуто алгоритми роботи застосунку, а також власні класи, розроблені для роботи зі збереженням даних. Ці алгоритми є основою функціонування програми, забезпечуючи ефективність та надійність її роботи. Крім того, було проаналізовано компоненти, що використовуються в застосунку, зокрема їх роль і взаємодію в загальній архітектурі програмного забезпечення. Таке комплексне проектування дозволяє створити добре структуровану і функціональну програму, що відповідає поставленим завданням.

# РОЗДІЛ 3 ОПИС РОБОТИ З ПРОГРАМНИМ ДОДАТКОМ ТА ЙОГО ТЕСТУВАННЯ

## 3.1 Опис роботи з програмним додатком (Опис інтерфейсу)

Після відкриття застосунку користувача зустічає наступне вікно:

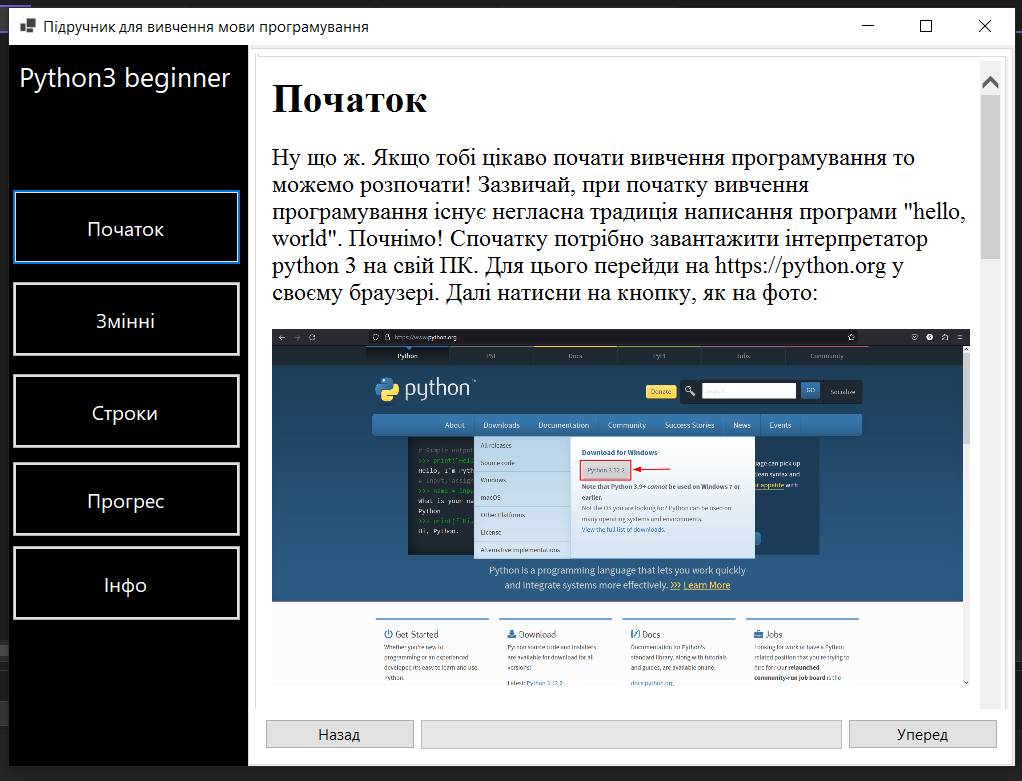


Рис. 3.1 Стартова сторінка застосунка

Користувач має можливість вивчати матеріал, поділений на три модулі: «Початок», «Змінні» та «Строки». Кожен модуль включає в себе три уроки і два тести, що дозволяють детально ознайомитися з темою та перевірити засвоєні знання.

Знизу кожного модуля розташовані кнопки «Назад» та «Уперед», які дозволяють зручно переміщатися між різними частинами матеріалу. Крім того, є прогрес-бар, що відображає поточний прогрес проходження модуля, надаючи користувачеві чітке уявлення про те, наскільки далеко він просунувся в навчанні.

Після того як користувач ознайомлюється з уроком, йому стає доступний тест, що дозволяє перевірити засвоєний матеріал. Успішне проходження тесту відкриває доступ до наступного уроку, що забезпечує послідовність навчання та поступове накопичення знань.

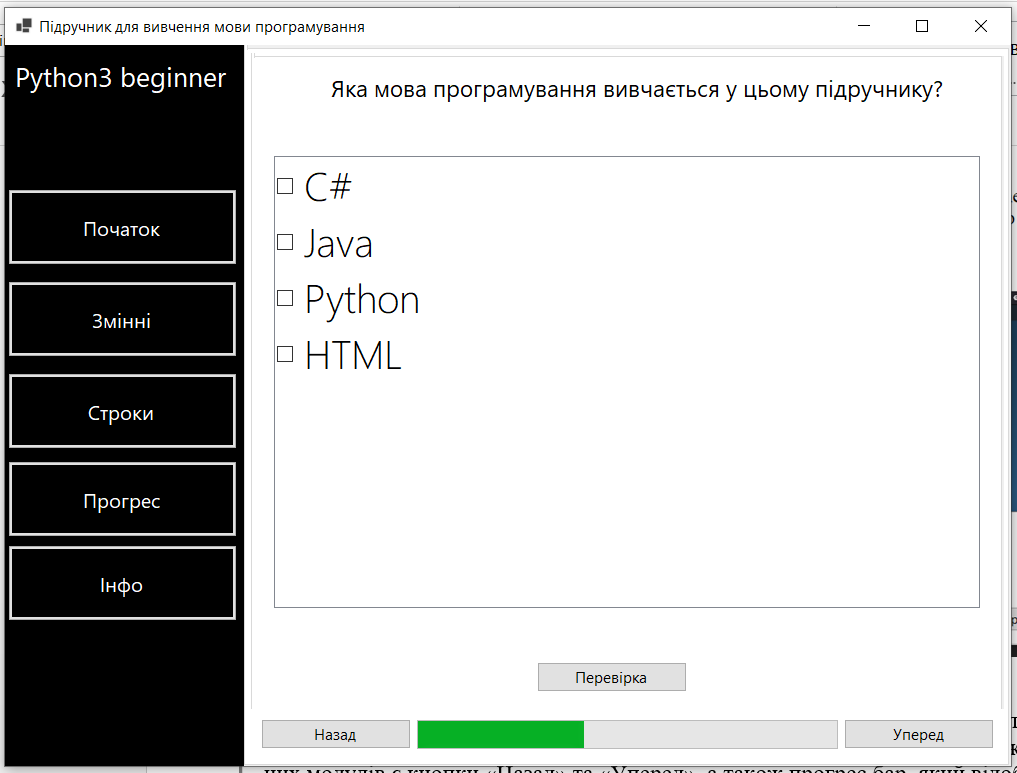


Рис. 3.2 Перший тест у модулі «Початок»

Якщо ж користувач натискає кнопку «Перевірка», програма розпочинає процес перевірки його відповіді. В цей момент здійснюється звірка введеного тексту з правильною відповіддю, що дозволяє оцінити точність і правильність відповіді користувача. У результаті цього процесу на екрані з'являється інформаційне вікно, яке містить чітке повідомлення про те, чи була відповідь правильною або неправильною. Це повідомлення допомагає користувачеві зрозуміти, чи відповів він на запитання вірно, що є важливим для подальшого навчання і корекції знань.

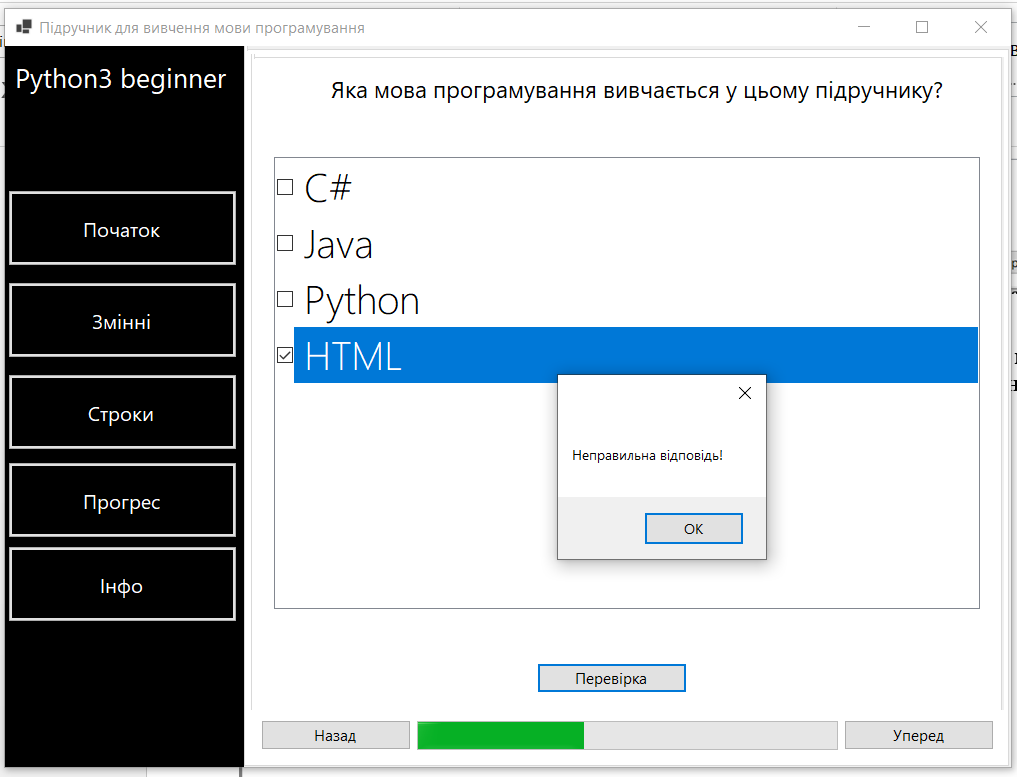


Рис. 3.3 Неправильна відповідь при приходженні першого тесту

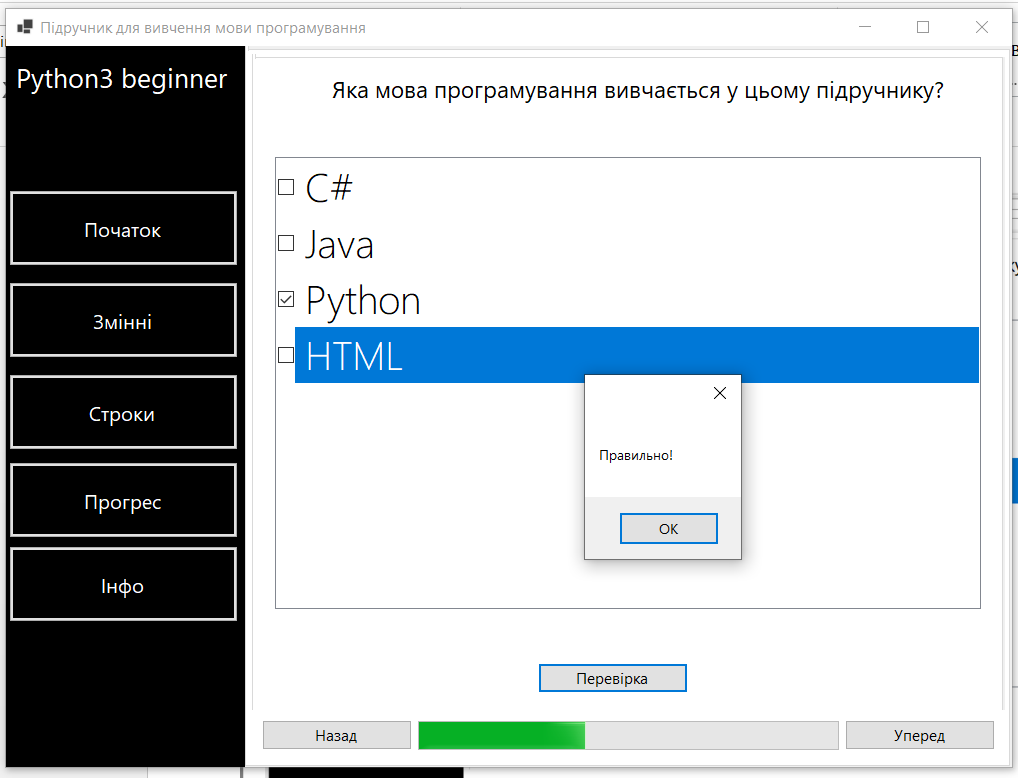


Рис. 3.4 Правильно пройдений перший тест

Користувач може вільно переключатися між різними модулями, однак при кожному переході відкривається перший урок нового модуля. Це дозволяє користувачу розпочати вивчення з базових основ, навіть якщо він переключається між модулями.

Якщо користувач успішно завершує перший урок і проходить відповідний тест, програма запам'ятовує його результати. Це означає, що при наступному відвідуванні модуля, користувач вже не буде змушений знову проходити цей урок і тест, що дозволяє уникнути повторення вже засвоєного матеріалу та забезпечує більш ефективний процес навчання.

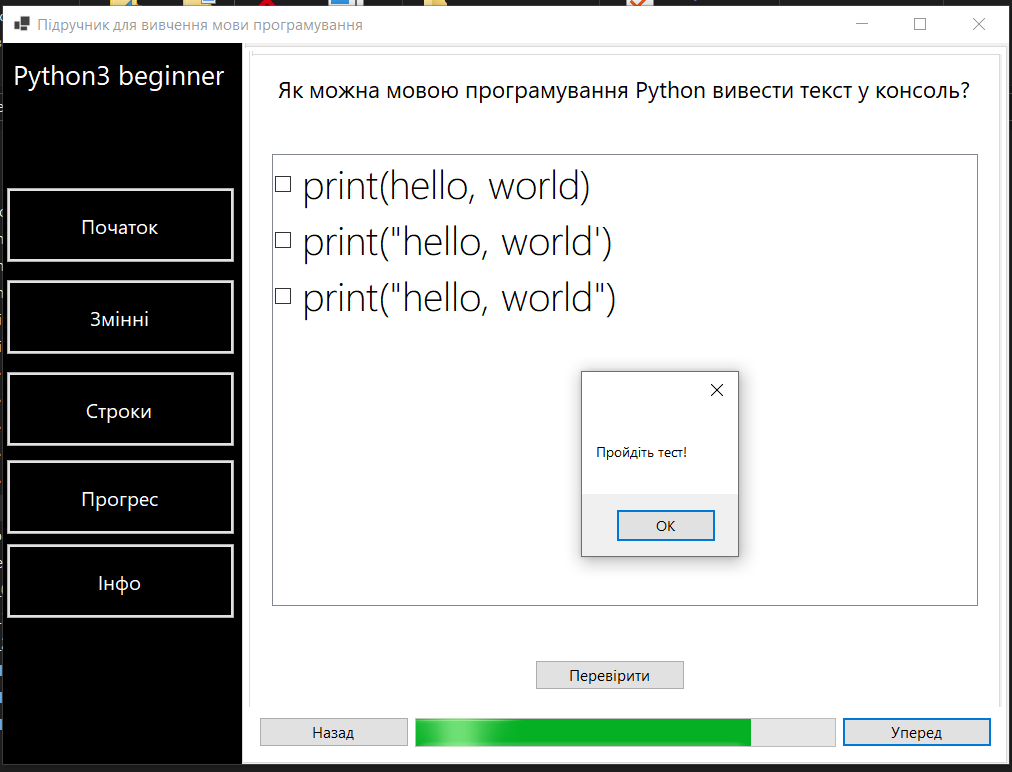


Рис. 3.5 Програма вимагає пройти тест для доступу до наступного уроку

Якщо користувач перейде до розділу «Прогрес», він отримає детальну інформацію про свої успіхи за допомогою віджетів прогрес-барів (progresBar). Ці віджети наочним чином відображають інформацію про вдалі спроби проходження тестів у кожному модулі.

Прогрес-бар кожного модуля показує, наскільки успішно користувач справився з тестами, на яких етапах навчання він досяг успіху, а також які тести ще потрібно пройти. Це дозволяє зручно відслідковувати загальний прогрес та візуально оцінювати результати навчання.

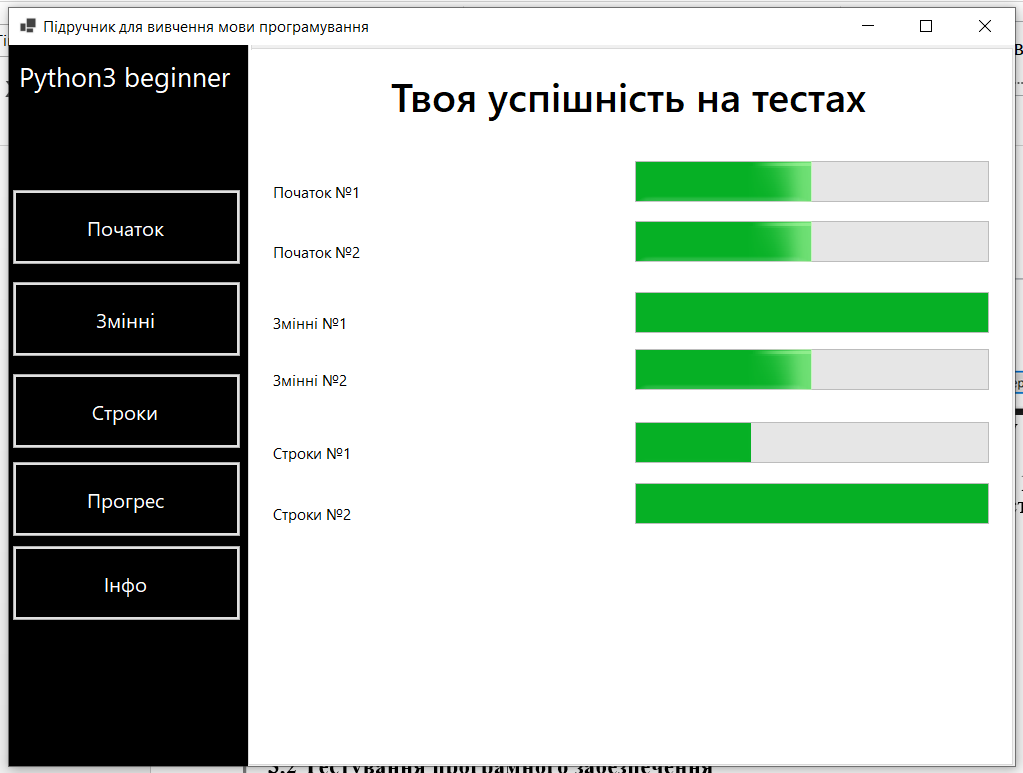


Рис. 3.6 Показ прогресу користувача

Також у додатку є спеціальний модуль, присвячений інформації про сам застосунок. У цьому модулі користувач знайде відомості про те, що даний додаток був створений під час виконання курсової роботи. Тут зазначено, що автором проекту є я, що дозволяє користувачам дізнатися більше про створення програми та її розробника.

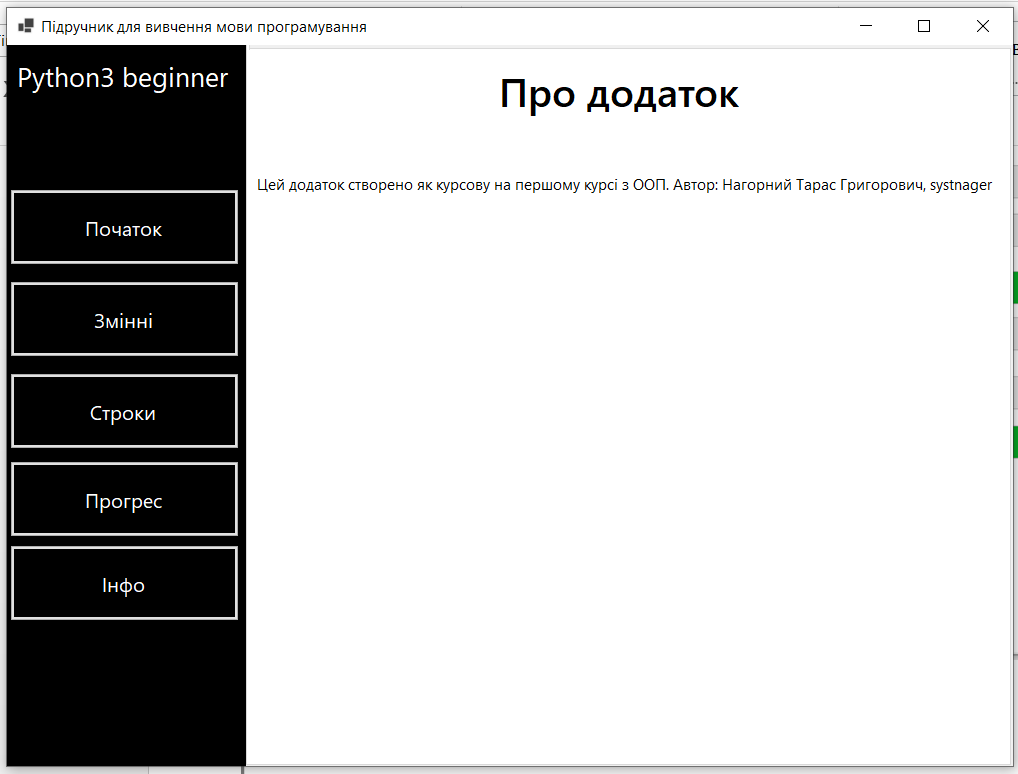


Рис. 3.7 Інформація про додаток

## 3.2 Тестування програмного забезпечення

Якщо користувач під час вивчення матеріалу знаходиться на першому уроці і натисне кнопку «Назад», програма сповістить його про помилку. В цьому випадку буде відображено повідомлення, яке інформує користувача, що він знаходиться на першому уроці і не може повернутися до попереднього етапу. Це допомагає уникнути непорозумінь і забезпечує зрозуміле сповіщення про неможливість переміщення назад у рамках поточного модулю.

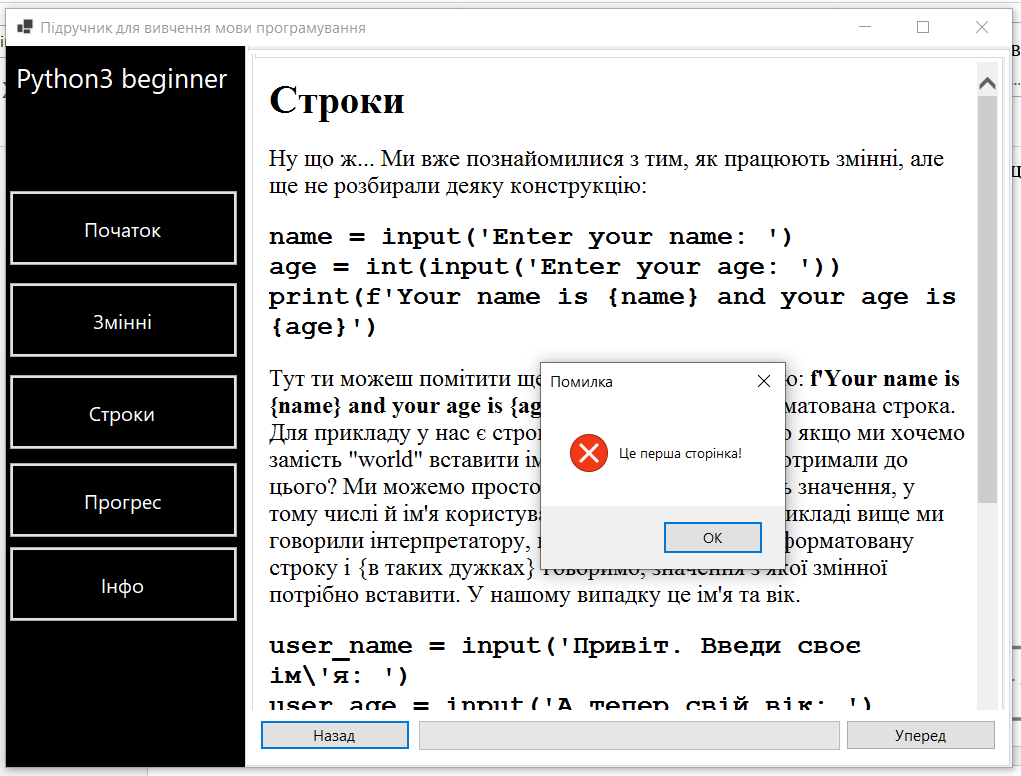


Рис. 3.8 Помилка при натиснені на кнопку назад

Аналогічно, якщо користувач знаходиться на останньому уроці і натискає кнопку «Уперед», програма сповістить його про помилку. У цьому випадку з’явиться повідомлення, яке інформує, що користувач досяг кінця уроків у поточному модулі і не може перейти вперед. Це забезпечує чіткість у навігації по матеріалу і попереджає користувача про те, що він вже завершив всі доступні уроки в цьому модулі.

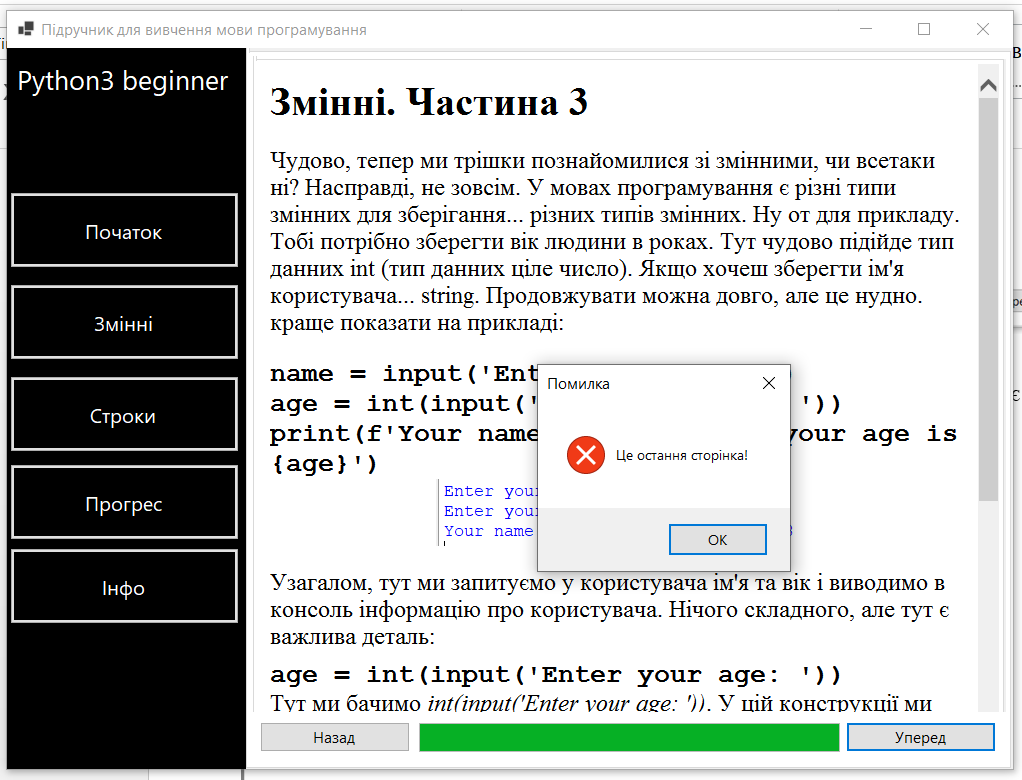


Рис. 3.9 Помилка при натиснені на кнопку «Уперед»

## Висновки до третього розділу

У цьому розділі було проведено всебічний і ґрунтовний аналіз роботи із застосунком, що включав детальний розгляд його інтерфейсу, який є ключовим елементом для забезпечення максимальної зручності та інтуїтивності для користувача. Значна увага була приділена тому, як користувач взаємодіє із програмою, включаючи всі основні функціональні елементи та можливості, які вона надає. Це дало змогу не лише оптимізувати дизайн інтерфейсу, але й переконатися, що всі аспекти роботи застосунку відповідають потребам і очікуванням користувачів.

Окрім цього, було ретельно розглянуто можливі помилки, що можуть виникнути під час використання застосунку, та способи їх запобігання. Завдяки проведеному аналізу вдалося впровадити відповідні механізми контролю та обробки помилок, що дозволило суттєво підвищити стабільність роботи програми. Ці заходи забезпечують безперебійну роботу застосунку навіть за наявності непередбачених обставин, що робить його надійним інструментом у руках користувачів. У підсумку, реалізовані підходи і рішення сприяли створенню стійкого та ефективного продукту, здатного витримувати різноманітні виклики під час експлуатації.

# ВИСНОВКИ

Під час виконання курсової роботи було проведено всебічний аналіз поставленої задачі, що включав детальне розглядання всіх аспектів і вимог, які необхідно врахувати для досягнення успіху. В рамках цього процесу було виявлено та проаналізовано існуючі аналоги, що дозволило зрозуміти, які функціональні можливості є критично важливими для кінцевого користувача. Цей аналіз також допоміг визначити, які саме технології та компоненти слід вибрати для ефективної реалізації рішення поставленої задачі.

Додатково, в ході виконання курсової роботи були розроблені та налагоджені алгоритми роботи додатку, що стали основою для його функціонування. Також було створено навчальний матеріал, що дозволяє наповнити додаток корисним контентом, необхідним для досягнення його основної мети.

Основною частиною виконаних завдань було застосування та закріплення всіх набутих знань, отриманих в рамках дисципліни об'єктно-орієнтованого програмування (ООП) з використанням мови програмування C#. Це дозволило не лише реалізувати поставлену задачу, але й поглибити розуміння принципів роботи застосунку, що суттєво вплинуло на якість і ефективність кінцевого продукту.

Результатом курсової роботи є створений електронний підручник, який є готовим і функціональним ресурсом для всіх бажаючих вивчити основи мови програмування Python 3.12. Цей підручник, що ґрунтується на детально розроблених матеріалах та алгоритмах, є надійним інструментом для освоєння Python, забезпечуючи користувачам якісний і доступний навчальний контент.

# СПИСОК ЛІТЕРАТУРИ

1. Об'єктно-орієнтоване програмування [Електронний ресурс] / Освітній портал ДУ «Житомирська політехніка» – 2020 – Режим доступу до ресурсу: https://learn.ztu.edu.ua/enrol/index.php?id=5512
2. Sololearn [Електронний ресурс] / Освітня платформа Sololearn - 2024 - Режим доступу до ресурсу: https://www.sololearn.com
3. Udemy [Електронний ресурс] / Освітня платформа Udemy - 2024 - Режим доступу до ресурсу: <https://www.udemy.com>
4. Open-source tool that uses simple textual descriptions to draw beautiful UML diagrams[Електронний ресурс] / Веб сервіс PlantUML - 2024 - Режим доступу до ресурсу: <https://plantuml.com>
5. Python doc [Електронний ресурс] / Документація мови програмування Python 3 - 2024 - Режим доступу до ресурсу: https://docs.python.org/3/
6. Stack overflow [Електронний ресурс] / Форум Stack overflow - 2024 - Режим доступу до ресурсу: <https://stackoverflow.com/questions/65226227/parse-json-using-newtonsoft>
7. Microsoft learn [Електронний ресурс] / Документація по C# - 2024 - Режим доступу до ресурсу: https://learn.microsoft.com/en-us/windows/winui/api/microsoft.ui.xaml.controls.tabview?view=winui-2.8
8. W3School [Електронний ресурс] / W3School HTML Style - CSS- 2024 - Режим доступу до ресурсу: https://www.w3schools.com/html/html\_css.asp
9. W3School [Електронний ресурс] / W3School Images - 2024 - Режим доступу до ресурсу: https://www.w3schools.com/html/html\_images.asp

# 

# ДОДАТКИ

## progresJson.cs

using System;

using System.Collections.Generic;

using System.IO;

using Newtonsoft.Json;

class ProgresJson

{

private string filePath;

private Dictionary<string, int> presetStructure = new Dictionary<string, int>

{

{ "beginning\_test\_0\_good", 0 },

{ "beginning\_test\_0\_bad", 0 },

{ "beginning\_test\_1\_good", 0 },

{ "beginning\_test\_1\_bad", 0 },

{ "string\_test\_0\_good", 0 },

{ "string\_test\_0\_bad", 0 },

{ "string\_test\_1\_good", 0 },

{ "string\_test\_1\_bad", 0 },

{ "variable\_test\_0\_good", 0 },

{ "variable\_test\_0\_bad", 0 },

{ "variable\_test\_1\_good", 0 },

{ "variable\_test\_1\_bad", 0 }

};

public ProgresJson(string filePath)

{

this.filePath = filePath;

if (!File.Exists(filePath))

{

WriteJson(presetStructure);

}

}

public int GetValueByKey(string key)

{

var jsonData = ReadJson();

if (jsonData.ContainsKey(key))

{

return jsonData[key];

}

throw new KeyNotFoundException($"Ключ '{key}' не знайдено у JSON файлі.");

}

public void AddValueToKey(string key, int value)

{

var jsonData = ReadJson();

if (jsonData.ContainsKey(key))

{

jsonData[key] += value;

WriteJson(jsonData);

}

else

{

throw new KeyNotFoundException($"Ключ '{key}' не знайдено у JSON файлі.");

}

}

private Dictionary<string, int> ReadJson()

{

var jsonContent = File.ReadAllText(filePath);

return JsonConvert.DeserializeObject<Dictionary<string, int>>(jsonContent);

}

private void WriteJson(Dictionary<string, int> jsonData)

{

var jsonContent = JsonConvert.SerializeObject(jsonData, Newtonsoft.Json.Formatting.Indented);

File.WriteAllText(filePath, jsonContent);

}

}

## Json.cs

class JsonHandler

{

private string filePath;

public JsonHandler(string filePath)

{

this.filePath = filePath;

if (!File.Exists(filePath))

{

File.WriteAllText(filePath, "{}");

}

}

public void AddOrUpdateEntry(string key, bool value)

{

var jsonData = ReadJson();

jsonData[key] = value;

WriteJson(jsonData);

}

private Dictionary<string, bool> ReadJson()

{

var jsonContent = File.ReadAllText(filePath);

return JsonConvert.DeserializeObject<Dictionary<string, bool>>(jsonContent);

}

private void WriteJson(Dictionary<string, bool> jsonData)

{

var jsonContent = JsonConvert.SerializeObject(jsonData, Newtonsoft.Json.Formatting.Indented);

File.WriteAllText(filePath, jsonContent);

}

public bool GetBoolValue(string key)

{

var jsonData = ReadJson();

if (jsonData.ContainsKey(key))

{

return jsonData[key];

}

return false;

}

}

## Form1.cs

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

using System.Xml.Linq;

namespace course\_work

{

public partial class Form1 : Form

{

WebBrowser webBrowser = new WebBrowser();

String baseDir = System.IO.Path.GetDirectoryName(System.Windows.Forms.Application.ExecutablePath);

int current\_tab\_index = 0;

List<TabPage> tabsList = new List<TabPage>();

List<int> uncompleteTestStartTabIndexList = new List<int>();

JsonHandler jsonHandler = new JsonHandler("data.json");

ProgresJson progresJson = new ProgresJson("progress.json");

public Form1()

{

InitializeComponent();

start\_Click(new object(), new EventArgs());

tabControl.ItemSize = new Size(0, 1);

startTabControl.ItemSize = new Size(0, 1);

VariableTabControl.ItemSize = new Size(0, 1);

StringTabControl.ItemSize = new Size(0, 1);

tabControl.SizeMode = TabSizeMode.Fixed;

startTabControl.SizeMode = TabSizeMode.Fixed;

VariableTabControl.SizeMode = TabSizeMode.Fixed;

StringTabControl.SizeMode = TabSizeMode.Fixed;

}

private void configureStartTab()

{

current\_tab\_index = 0;

startTabProgressBar.Value = 0;

addSTartTabListElements();

tabControl.SelectTab(0);

startTabControl.SelectTab(current\_tab\_index);

}

private void configureVariableTab()

{

current\_tab\_index = 0;

variable\_progressBar.Value = 0;

addVariableTabListElements();

tabControl.SelectTab(1);

VariableTabControl.SelectTab(current\_tab\_index);

}

private void configureStringTab()

{

current\_tab\_index = 0;

string\_progressBar.Value = 0;

addStringTabListElements();

tabControl.SelectTab(2);

StringTabControl.SelectTab(current\_tab\_index);

}

private void addStringTabListElements()

{

current\_tab\_index = 0;

tabsList = new List<TabPage>();

tabsList.Add(string\_tab\_0);

tabsList.Add(string\_tab\_1);

tabsList.Add(string\_tab\_2);

tabsList.Add(string\_tab\_3);

tabsList.Add(string\_tab\_4);

}

private void addVariableTabListElements()

{

current\_tab\_index = 0;

tabsList = new List<TabPage>();

tabsList.Add(variable\_tab\_0);

tabsList.Add(variable\_tab\_1);

tabsList.Add(variable\_tab\_2);

tabsList.Add(variable\_tab\_3);

tabsList.Add(variable\_tab\_4);

}

private void addSTartTabListElements()

{

current\_tab\_index = 0;

tabsList = new List<TabPage>();

tabsList.Add(startTab\_0);

tabsList.Add(startTab\_1);

tabsList.Add(startTab\_2);

tabsList.Add(startTab\_3);

tabsList.Add(startTab\_4);

}

private void uncompleteTestStartTabIndexListElements()

{

}

private void loadHTMLInWebView(int tabIndex)

{

if (tabControl.SelectedIndex == 0)

{

if (tabIndex == 0)

{

webBrowser.Dock = DockStyle.Fill;

startTabwebView.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\beginning\_0.html";

webBrowser.Navigate(htmlFilePath);

}

else if (tabIndex == 2)

{

webBrowser.Dock = DockStyle.Fill;

startTab2webView.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\beginning\_1.html";

webBrowser.Navigate(htmlFilePath);

}

else if (tabIndex == 4)

{

webBrowser.Dock = DockStyle.Fill;

startTab4webView.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\beginning\_2.html";

webBrowser.Navigate(htmlFilePath);

}

}

else if (tabControl.SelectedIndex == 1)

{

if (tabIndex == 0)

{

webBrowser.Dock = DockStyle.Fill;

VariableWebViewPanel.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\variable\_0.html";

webBrowser.Navigate(htmlFilePath);

}

else if (tabIndex == 2)

{

webBrowser.Dock = DockStyle.Fill;

VariableWebViewPanel2.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\variable\_1.html";

webBrowser.Navigate(htmlFilePath);

}

else if (tabIndex == 4)

{

webBrowser.Dock = DockStyle.Fill;

VariableWebViewPanel4.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\variable\_2.html";

webBrowser.Navigate(htmlFilePath);

}

}

else if (tabControl.SelectedIndex == 2)

{

if (tabIndex == 0)

{

webBrowser.Dock = DockStyle.Fill;

StringWebViewPanel.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\string\_0.html";

webBrowser.Navigate(htmlFilePath);

}

else if (tabIndex == 2)

{

webBrowser.Dock = DockStyle.Fill;

StringWebViewPanel2.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\string\_1.html";

webBrowser.Navigate(htmlFilePath);

}

else if (tabIndex == 4)

{

webBrowser.Dock = DockStyle.Fill;

StringWebViewPanel4.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\string\_2.html";

webBrowser.Navigate(htmlFilePath);

}

}

}

private void start\_Click(object sender, EventArgs e)

{

configureStartTab();

webBrowser.Dock = DockStyle.Fill;

startTabwebView.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\beginning\_0.html";

webBrowser.Navigate(htmlFilePath);

tabControl.SelectTab(0);

}

private void strings\_Click(object sender, EventArgs e)

{

configureStringTab();

webBrowser.Dock = DockStyle.Fill;

StringWebViewPanel.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\string\_0.html";

webBrowser.Navigate(htmlFilePath);

tabControl.SelectTab(2);

}

private void startTab\_next\_button\_Click(object sender, EventArgs e)

{

if (current\_tab\_index + 1 == 2)

{

if (jsonHandler.GetBoolValue("beginning\_test\_0") == false)

{

MessageBox.Show("Пройдіть тест!");

return;

}

}

else if (current\_tab\_index + 1 == 4)

{

if (jsonHandler.GetBoolValue("beginning\_test\_1") == false)

{

MessageBox.Show("Пройдіть тест!");

return;

}

}

if ((current\_tab\_index + 1) < tabsList.Count)

{

current\_tab\_index++;

int progress = ((current\_tab\_index + 1) \* 100 / tabsList.Count);

startTabProgressBar.Value = progress;

loadHTMLInWebView(current\_tab\_index);

startTabControl.SelectTab(current\_tab\_index);

}

else

{

MessageBox.Show("Це остання сторінка!", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void startTab\_back\_button\_Click(object sender, EventArgs e)

{

if (current\_tab\_index > 0)

{

current\_tab\_index--;

int progress = ((current\_tab\_index + 1) \* 100 / tabsList.Count);

startTabProgressBar.Value = progress;

loadHTMLInWebView(current\_tab\_index);

startTabControl.SelectTab(current\_tab\_index);

}

else

{

MessageBox.Show("Це перша сторінка!", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void label1\_Click(object sender, EventArgs e)

{

}

private void panel\_content\_Paint(object sender, PaintEventArgs e)

{

}

private void Form1\_Load(object sender, EventArgs e)

{

}

private void tabPage1\_Click(object sender, EventArgs e)

{

}

private void tabPage8\_Click(object sender, EventArgs e)

{

}

private void variable\_Click(object sender, EventArgs e)

{

configureVariableTab();

webBrowser.Dock = DockStyle.Fill;

VariableWebViewPanel.Controls.Add(webBrowser);

string htmlFilePath = $@"{baseDir}\variable\_0.html";

webBrowser.Navigate(htmlFilePath);

tabControl.SelectTab(1);

}

private void progressBar1\_Click(object sender, EventArgs e)

{

}

private void startTab1CheckButton\_Click(object sender, EventArgs e)

{

int pythonCount = 0;

foreach (object item in startTab1CheckedListBox.CheckedItems)

{

if (item.ToString() == "Python")

{

pythonCount++;

}

else

{

MessageBox.Show("Неправильна відповідь!");

progresJson.AddValueToKey("beginning\_test\_0\_bad", 1);

return;

}

}

if (pythonCount == 1)

{

MessageBox.Show("Правильно!");

jsonHandler.AddOrUpdateEntry("beginning\_test\_0", true);

progresJson.AddValueToKey("beginning\_test\_0\_good", 1);

}

else

{

MessageBox.Show("Неправильна відповідь!");

progresJson.AddValueToKey("beginning\_test\_0\_bad", 1);

}

}

private void button1\_Click(object sender, EventArgs e)

{

int pythonCount = 0;

foreach (object item in checkedListBox1.CheckedItems)

{

if (item.ToString() == "print(\"hello, world\")")

{

pythonCount++;

}

else

{

MessageBox.Show("Неправильна відповідь!");

progresJson.AddValueToKey("beginning\_test\_1\_bad", 1);

return;

}

}

if (pythonCount == 1)

{

jsonHandler.AddOrUpdateEntry("beginning\_test\_1", true);

MessageBox.Show("Правильно!");

progresJson.AddValueToKey("beginning\_test\_1\_good", 1);

}

else

{

MessageBox.Show("Неправильна відповідь!");

progresJson.AddValueToKey("beginning\_test\_1\_bad", 1);

}

}

private void checkedListBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void startTab2webView\_Paint(object sender, PaintEventArgs e)

{

}

private void tabPage6\_Click(object sender, EventArgs e)

{

}

private void panel5\_Paint(object sender, PaintEventArgs e)

{

}

private void string\_next\_Click(object sender, EventArgs e)

{

if (current\_tab\_index + 1 == 2)

{

if (jsonHandler.GetBoolValue("string\_test\_0") == false)

{

MessageBox.Show("Пройдіть тест!");

return;

}

}

else if (current\_tab\_index + 1 == 4)

{

if (jsonHandler.GetBoolValue("string\_test\_1") == false)

{

MessageBox.Show("Пройдіть тест!");

return;

}

}

if ((current\_tab\_index + 1) < tabsList.Count)

{

current\_tab\_index++;

int progress = ((current\_tab\_index + 1) \* 100 / tabsList.Count);

string\_progressBar.Value = progress;

loadHTMLInWebView(current\_tab\_index);

StringTabControl.SelectTab(current\_tab\_index);

}

else

{

MessageBox.Show("Це остання сторінка!", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void string\_back\_Click(object sender, EventArgs e)

{

if (current\_tab\_index > 0)

{

current\_tab\_index--;

int progress = ((current\_tab\_index + 1) \* 100 / tabsList.Count);

string\_progressBar.Value = progress;

loadHTMLInWebView(current\_tab\_index);

StringTabControl.SelectTab(current\_tab\_index);

}

else

{

MessageBox.Show("Це перша сторінка!", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void variable\_back\_Click(object sender, EventArgs e)

{

if (current\_tab\_index > 0)

{

current\_tab\_index--;

int progress = ((current\_tab\_index + 1) \* 100 / tabsList.Count);

variable\_progressBar.Value = progress;

loadHTMLInWebView(current\_tab\_index);

VariableTabControl.SelectTab(current\_tab\_index);

}

else

{

MessageBox.Show("Це перша сторінка!", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void variable\_next\_Click(object sender, EventArgs e)

{

if (current\_tab\_index + 1 == 2)

{

if (jsonHandler.GetBoolValue("variable\_test\_0") == false)

{

MessageBox.Show("Пройдіть тест!");

return;

}

}

else if (current\_tab\_index + 1 == 4)

{

if (jsonHandler.GetBoolValue("variable\_test\_1") == false)

{

MessageBox.Show("Пройдіть тест!");

return;

}

}

if ((current\_tab\_index + 1) < tabsList.Count)

{

current\_tab\_index++;

int progress = ((current\_tab\_index + 1) \* 100 / tabsList.Count);

variable\_progressBar.Value = progress;

loadHTMLInWebView(current\_tab\_index);

VariableTabControl.SelectTab(current\_tab\_index);

}

else

{

MessageBox.Show("Це остання сторінка!", "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void label4\_Click(object sender, EventArgs e)

{

}

private void variable\_check\_answer\_button\_Click(object sender, EventArgs e)

{

int right\_answer\_count = 0;

foreach (object item in VariablecheckedListBox.CheckedItems)

{

if (item.ToString() == "input()")

{

right\_answer\_count++;

}

else

{

MessageBox.Show("Неправильна відповідь!");

progresJson.AddValueToKey("variable\_test\_0\_bad", 1);

return;

}

}

if (right\_answer\_count == 1)

{

jsonHandler.AddOrUpdateEntry("variable\_test\_0", true);

progresJson.AddValueToKey("variable\_test\_0\_good", 1);

MessageBox.Show("Правильно!");

}

else

{

MessageBox.Show("Неправильна відповідь!");

progresJson.AddValueToKey("variable\_test\_0\_bad", 1);

}

}

private void variable\_check\_answer\_button2\_Click(object sender, EventArgs e)

{

int right\_answer\_count = 0;

foreach (object item in VariablecheckedListBox2.CheckedItems)

{

if (item.ToString() == "Виведе Hello, Petro")

{

right\_answer\_count++;

}

else

{

MessageBox.Show("Неправильна відповідь!");

progresJson.AddValueToKey("variable\_test\_1\_bad", 1);

return;

}

}

if (right\_answer\_count == 1)

{

jsonHandler.AddOrUpdateEntry("variable\_test\_1", true);

progresJson.AddValueToKey("variable\_test\_1\_good", 1);

MessageBox.Show("Правильно!");

}

else

{

MessageBox.Show("Неправильна відповідь!");

progresJson.AddValueToKey("variable\_test\_1\_bad", 1);

}

}

private void button10\_Click(object sender, EventArgs e)

{

int right\_answer\_count = 0;

foreach (object item in StringcheckedListBox.CheckedItems)

{

if (item.ToString() == "Виведе Petro, how are you?")

{

right\_answer\_count++;

}

else

{

MessageBox.Show("Неправильна відповідь!");

progresJson.AddValueToKey("string\_test\_0\_bad", 1);

return;

}

}

if (right\_answer\_count == 1)

{

jsonHandler.AddOrUpdateEntry("string\_test\_0", true);

progresJson.AddValueToKey("string\_test\_0\_good", 1);

MessageBox.Show("Правильно!");

}

else

{

MessageBox.Show("Неправильна відповідь!");

progresJson.AddValueToKey("string\_test\_0\_bad", 1);

}

}

private void checkedListBox4\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void string\_is\_answer\_right\_check\_button\_2\_Click(object sender, EventArgs e)

{

int right\_answer\_count = 0;

foreach (object item in StringcheckedListBox2.CheckedItems)

{

if (item.ToString() == "Буде помилка")

{

right\_answer\_count++;

}

else

{

MessageBox.Show("Неправильна відповідь!");

progresJson.AddValueToKey("string\_test\_1\_bad", 1);

return;

}

}

if (right\_answer\_count == 1)

{

jsonHandler.AddOrUpdateEntry("string\_test\_1", true);

progresJson.AddValueToKey("string\_test\_1\_good", 1);

MessageBox.Show("Правильно!");

}

else

{

MessageBox.Show("Неправильна відповідь!");

progresJson.AddValueToKey("string\_test\_1\_bad", 1);

}

}

private void button8\_Click(object sender, EventArgs e)

{

tabControl.SelectTab(3);

beginning\_statistic\_progressBar.Value = progresJson.GetValueByKey("beginning\_test\_0\_good") \* 100 / (progresJson.GetValueByKey("beginning\_test\_0\_good") + progresJson.GetValueByKey("beginning\_test\_0\_bad"));

beginning\_statistic\_progressBar2.Value = progresJson.GetValueByKey("beginning\_test\_1\_good") \* 100 / (progresJson.GetValueByKey("beginning\_test\_1\_good") + progresJson.GetValueByKey("beginning\_test\_1\_bad"));

variable\_statistic\_progressBar.Value = progresJson.GetValueByKey("variable\_test\_0\_good") \* 100 / (progresJson.GetValueByKey("variable\_test\_0\_good") + progresJson.GetValueByKey("variable\_test\_0\_bad"));

variable\_statistic\_progressBar2.Value = progresJson.GetValueByKey("variable\_test\_1\_good") \* 100 / (progresJson.GetValueByKey("variable\_test\_1\_good") + progresJson.GetValueByKey("variable\_test\_1\_bad"));

string\_statistic\_progressBar.Value = progresJson.GetValueByKey("string\_test\_0\_good") \* 100 / (progresJson.GetValueByKey("string\_test\_0\_good") + progresJson.GetValueByKey("string\_test\_0\_bad"));

string\_statistic\_progressBar2.Value = progresJson.GetValueByKey("string\_test\_1\_good") \* 100 / (progresJson.GetValueByKey("string\_test\_1\_good") + progresJson.GetValueByKey("string\_test\_1\_bad"));

}

private void show\_info\_button\_Click(object sender, EventArgs e)

{

tabControl.SelectTab(4);

}

}

}