Український державний університет імені Михайла Драгоманова

Факультет математики, інформатики та фізики

**Звіт про виконану роботу до лабораторного заняття № 3**

З курсу **Архітектура програмної платформи .NET**

***Виконала:*** студентка групи41ІПЗ

спеціальності «Інженерія програмного

забезпечення»

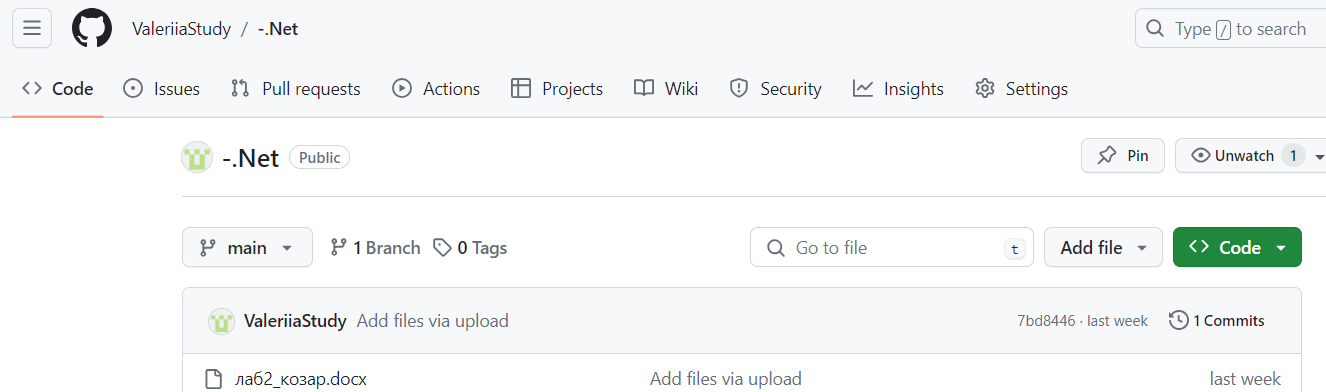
***Козар В.М.***

***Викладач:  [Січкаренко](https://moodle.fmif.udu.edu.ua/user/view.php?id=21135&course=1)***

***[Володимир Олександрович](https://moodle.fmif.udu.edu.ua/user/view.php?id=21135&course=1)***

1. Ознайомилась з теоретичною частиною.

2. Обрала тему - Текстовий редактор

3. Перейти в локальний репозиторій з лабораторними роботами;

4. Створити окрему гілку (git checkout -b) в системі контролю версій Git для поточної лабораторної роботи;

5. Створити новий консольний проект на мові програмування C#;

6. Реалізувати 3 класи (на вибір студента) з обраної теми;

7. Під час реалізації програми використовувати типи даних визначені в просторі імен System;

8. В функції Main продемонструвати використання створених класів;

9. Продемонструвати роботу програми;

|  |
| --- |
| using System;  // Клас для представлення документу  public class Document  {  public string Title { get; set; }  public string Content { get; set; }  public void Display()  {  Console.WriteLine($"Document: {Title}");  Console.WriteLine("Content:");  Console.WriteLine(Content);  Console.WriteLine();  }  }  // Клас для текстового редактора  public class Editor  {  public void EditDocument(Document document, string newContent)  {  Console.WriteLine($"Editing document: {document.Title}");  document.Content = newContent;  Console.WriteLine("Document edited successfully.");  }  public void DisplayDocument(Document document)  {  document.Display();  }  }  // Клас для управління файлами  public class FileManager  {  public void SaveDocument(Document document, string filePath)  {  Console.WriteLine($"Saving document to: {filePath}");  // Логіка для збереження документу в файл  Console.WriteLine("Document saved successfully.");  }  }  class Program  {  static void Main()  {  // Приклад використання створених класів  // Створення документу  Document myDocument = new Document  {  Title = "My First Document",  Content = "This is the content of my document."  };  // Використання редактора  Editor textEditor = new Editor();  textEditor.EditDocument(myDocument, "Updated content of my document");  textEditor.DisplayDocument(myDocument);  // Управління файлами  FileManager fileManager = new FileManager();  fileManager.SaveDocument(myDocument, "C:\\Documents\\MyDocument.txt");  }  } |
| 3 |

1. Зафіксувати зміни (git commit);

4

11. Надіслати зміни у віддалений репозиторій (git push);

5

**Відповіді на запитання**

1. Клас:

Клас є основною одиницею організації коду в об'єктно-орієнтованому програмуванні.

Основні складові класу включають поля (члени класу) і методи (функції члена класу).

Об'єкт - це конкретний екземпляр класу, створений на основі його опису.

1. CTS (Common Type System) та CLS (Common Language Specification):

CTS: Це система, яка визначає, як типи даних представлені та взаємодіють в середовищі CLR (Common Language Runtime).

CLS: Це набір правил та вимог, які гарантують сумісність мов програмування на платформі .NET, дозволяючи взаємодіяти між різними мовами.

1. Типи даних представлені в CTS:

* Цілі числа (int, long, short)
* Дійсні числа (float, double)
* Символьні типи (char)
* Булеві значення (bool)
* Спеціальні типи (наприклад, строки, масиви)

1. Типи значень та типи посилань:

* Типи значень: Зберігають значення напряму в пам'яті. Приклади: цілі числа, символи.
* Типи посилань: Зберігають адресу пам'яті, де знаходиться об'єкт. Приклади: класи, інтерфейси. Робота з ними включає роботу з вказівниками та може включати механізми керування пам'яттю.

1. "Магічні числа":

* Це жорстко задані константи у коді, які важко розуміти.
* Їх слід уникати, оголошуючи константи або використовуючи їхні значення як змінні з поняттям.

1. Оголошення та використання констант:

* const використовується для оголошення констант у C#.
* Константи повинні бути оголошені з поняттями та назвами, щоб зробити код зрозумілішим.

1. Тип даних описаний за допомогою struct:

* struct є значимим типом даних, який може містити поля та методи.
* Структури передаються за значенням, і їхні екземпляри зазвичай використовуються для представлення простих об'єктів.

1. Тип даних описаний за допомогою class:

class є посиланням на тип даних, який може містити поля, методи, властивості та інші члени.

Екземпляри класу передаються за посиланням.

1. Різниця між struct та class:

Структури є значимими типами, класи - посиланнями.

Структури передаються за значенням, класи - за посиланням.

Структури підтримують наслідування тільки від інтерфейсів, в той час як класи підтримують наслідування від інших класів.

1. Специфікатори доступу до класів, структур та їх частин:

public: Доступно з будь-якого місця.

private: Доступно тільки всередині того самого класу.

protected: Доступно всередині того самого класу та його підкласів.

internal: Доступно тільки в межах того самого збірника.

protected internal: Доступно всередині того самого збірника або в підкласах навіть за його межами.