МІНІСТЕРСТВО ВНУТРІШНІХ СПРАВ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ВНУТРІШНІХ СПРАВ
Кафедра кібербезпеки та DATA-технологій, факультет №6

МЕТОДИЧНІ МАТЕРІАЛИ до

лабораторних робіт із навчальної дисципліни

" Об’єктно-орієнтоване програмування "

обов’язкових компонент освітньої програми першого (бакалаврського) рівня вищої освіти

Спеціальність: 125 "Кібербезпека"
(«Безпека інформаційних та комунікаційних систем»)

Харків 2023

ЗАТВЕРДЖЕНО
Науково-методичною радою
Харківського національного
університету внутрішніх справ
Протокол від 30.01.2023 № 1

СХВАЛЕНО
Вченою радою факультету № 6
Протокол від 18.01.2023 № 1

ПОГОДЖЕНО
Секцією Науково-методичної ради
ХНУВС з технічних дисциплін
Протокол від 27.01.2023 № 1

Розглянуто на засіданні кафедри кібербезпеки та DATA-технологій факультету № 6 *(протокол від 13.01.2023 № 1)*

Розробники:

Професор кафедри, к.т.н., доцент; Струков В. М.;

Старший викладач, Цуранов М.В.;

Рецензенти:

1. *Пєвнєв В.Я., д.т.н., доцент, професор кафедри комп'ютерних систем, мереж та кібербезпеки факультету радіоелектроніки, комп'ютерних систем та інфокомунікацій НАУ «ХАІ» ім. М.Є. Жуковського;*
2. *Світличний В.А., к.т.н., доцент, доцент кафедри протидії кіберзлочинності факультету №4 ХНУВС.*

1. Розподіл часу навчальної дисципліни за темами (денна форма навчання)

Номер та найменування теми	Кількість годин відведених на вивчення навчальної дисципліни						Вид контролю
	Всього	з них:					
		лекції	Семінарські заняття	Практичні заняття	Лабораторні заняття	Самостійна робота	
Семестр 4							
Тема № 1. Основні поняття об'єктно-орієнтованого програмування.	**14**	**2**		**2**		**10**	
Тема № 2. Основи уніфікованої мови моделювання UML.	**14**	**2**		**2**		**10**	
Тема № 3. Основи моделювання поведінки системи в UML.	**14**	**2**		**2**		**10**	
Тема № 4. Моделювання класів в UML.	**16**	**2**		**2**	**2**	**10**	
Тема № 5. Архітектура платформи .NET та програмування для .NET Framework	**22**	**4**		**2**	**6**	**10**	
Тема № 6. Синтаксис мови C#. Основні операції, оператори та типи даних C#	**25**	**4**		**2**	**6**	**13**	
Тема № 7. Класи та об'єкти. Перетворення типів даних	**20**	**4**			**6**	**10**	
Тема № 8. Спадкування. Інтерфейси та їх використання.	**25**	**4**			**8**	**13**	
Всього за семестр № 2:	**150**	**24**		**12**	**28**	**86**	**залік**

Розподіл часу навчальної дисципліни за темами
(заочна форма навчання)

Номер та найменування теми	Кількість годин відведених на вивчення навчальної дисципліни						Вид контролю
	Всього	з них:					
		лекції	Семінарські заняття	Практичні заняття	Лабораторні заняття	Самостійна робота	
Семестр 4							
Тема № 1. Основні поняття об'єктно-орієнтованого програмування.	**17**	**2**				**15**	
Тема № 2. Основи уніфікованої мови моделювання UML.	**17**			**2**		**15**	
Тема № 3. Основи моделювання поведінки системи в UML.	**15**					**15**	
Тема № 4. Моделювання класів в UML.	**15**					**15**	
Тема № 5. Архітектура платформи .NET та програмування для .NET Framework	**26**	**2**			**4**	**20**	
Тема № 6. Синтаксис мови C#. Основні операції, оператори та типи даних C#	**20**					**20**	
Тема № 7. Класи та об'єкти. Перетворення типів даних	**20**					**20**	
Тема № 8. Спадкування. Інтерфейси та їх використання.	**20**					**20**	
Всього за семестр № 2:	**150**	**6**	**4**			**140**	**залік**

2. МЕТОДИЧНІ ВКАЗІВКИ ДО ЛАБОРАТОРНИХ РОБІТ

ЛАБОРАТОРНА РОБОТА №1 ВИВЧЕННЯ СЕРЕДОВИЩА РОЗРОБКИ MICROSOFT VISUAL STUDIO.

Мета роботи: Ознайомитись з базовими особливостями мови програмування C#, освоїти використання середовища Visual Studio для створення консольних програм мовою C#. Вивчити принципи об'єднання кількох проектів у вирішення. Вивчити можливості NClass зі створення діаграм класів та генерації коду. Отримати базові відомості про класи та об'єкти. **Час проведення**: 4 години. **Місце проведення:** комп'ютерний клас.

Навчальні питання:

1. Створення рішення у Visual Studio.

Література:

Основна література.

1. Коноваленко І.В. Програмування мовою C# 6.0. Тернопіль, ТНТУ, 2016. 227 с.
2. H. Schildt, "C# 4.0 The Complete Reference McGraw Hill Education; 1st edition" – 984 p., 2017.
3. J. Sharp, "Microsoft Visual C# Step by Step", Pearson Education – 878p., 2018.

Допоміжна література.

1. A. Troelsen, "Pro C# 7 With .NET and .NET Core" - Apress, — 1372 p., 2017.
2. J. Albahari, B. Albahari, "C# 7.0 in a Nutshell", O'Reilly Media— 1090 p., 2017.
3. Інструментальні програмні засоби розробки ІУС. Методичні вказівки до виконання лабораторних робіт. / уклад.: К.І. Київська–Київ: КНУБА, 2018. – 40 с

Інформаційні ресурси в Інтернеті

1. C# Programming. Yellow Book [Ел. ресурс]. URL: https://www.robmiles.com/c-yellow-book
2. Secure Coding Guidelines for .NET [Ел. ресурс]. URL: https://docs.microsoft.com/en-us/dotnet/standard/security/secure-coding-guidelines
3. C# Reference, сайт розробників MSDN. [Електронний ресурс] – Режим доступу: https://docs.microsoft.com/uk-ua/dotnet/csharp/languagereference/index
4. Об'єктно-орієнтоване програмування. Частина 1. Основи об'єктно-орієнтованого програмування на мові C#.: Навчальний посібник. / Д.В. Настенко, А. Б. Нестерко.

– К.: НТУУ «КПІ», 2016. - 76с. [Електронний ресурс]. – Режим доступу https://ela.kpi.ua/bitstream/123456789/16671/1/OOP_manual.pdf

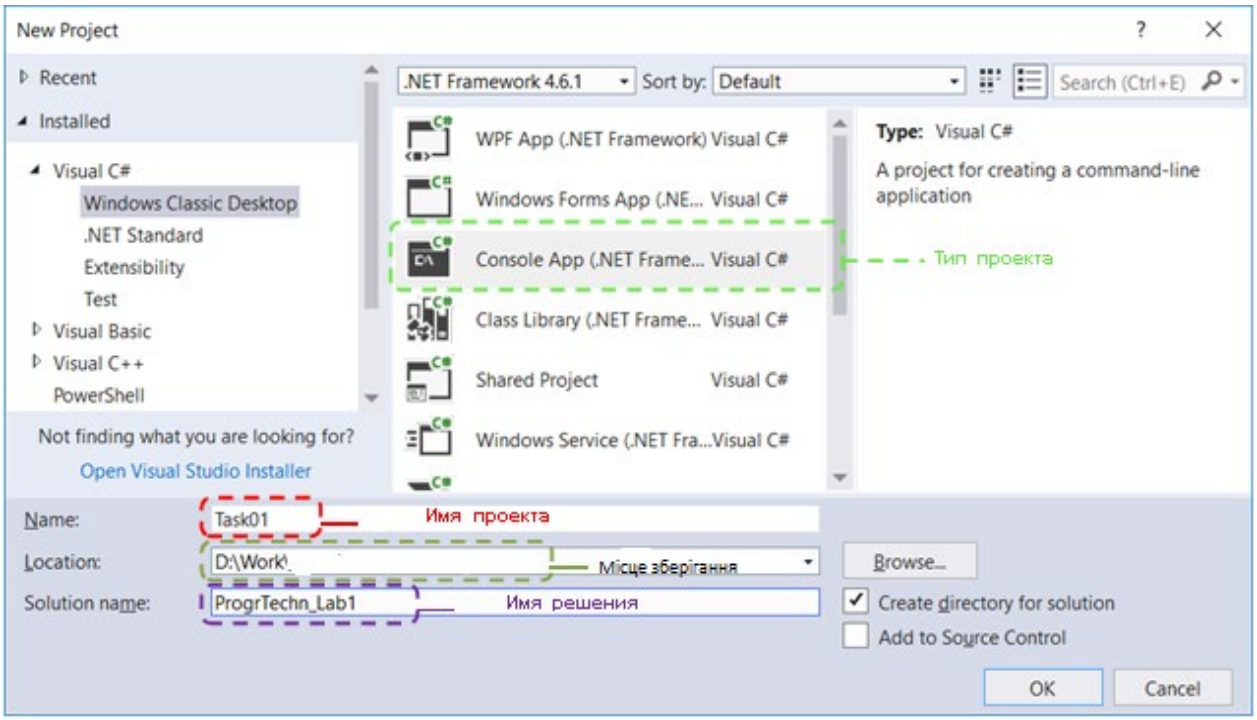
План проведення заняття:

Теоретичні відомості

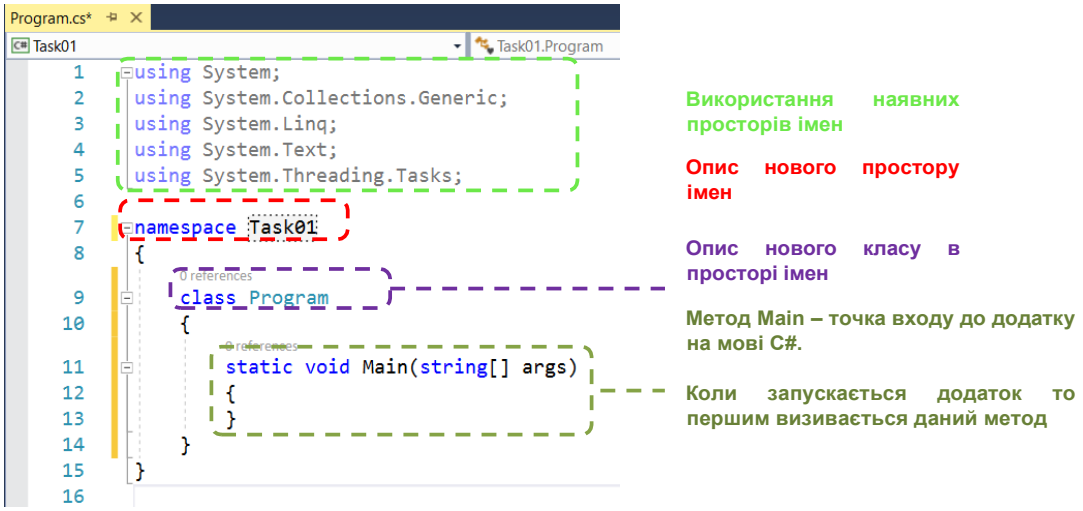
Рішення в Visual Studio - це контейнер високого рівня для інших елементів, таких як проекти, readme файли, діаграми і т.д.

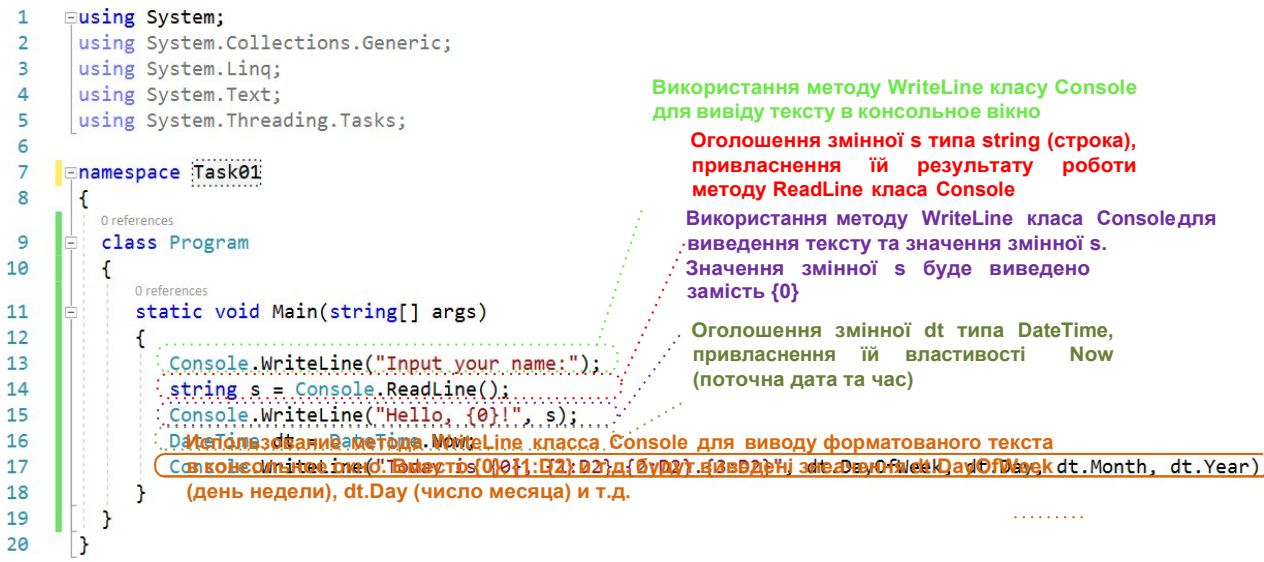
Послідовність виконання завдання

Створіть за допомогою майстра (File - New - Project) рішення з ім'ям ProgrTech_Lab1 і додайте до нього проект консольної програми на C # з ім'ям Task01:



В результаті середовищем буде згенеровано наступний код:





Потім запустіть програму двічі: спочатку за допомогою комбінації Ctrl+F5, а потім за допомогою F5. Зверніть увагу на різницю у способі виконання програми.

– Ознайомтеся з можливостями класу Console у MSDN (https://docs.microsoft.com/en-us/dotnet/api/system.console), додайте до розробленої раніше програми:
– □ зміна кольору тексту;
– □ зміна кольору фону;
– □ зміна тексту в заголовку консольного вікна;
– □ зміна ширини/висоти консольного вікна;
– □ видачу звукового сигналу.
– Ознайомтеся з можливостями структури DateTime у MSDN (https://docs.microsoft.com/en-us/dotnet/api/system.datetime), додайте до розробленої раніше програми:
– □ виведення поточного часу;
– □ виведення решти днів до нового року.

Вимоги до звіту

Звіт з цієї лабораторної роботи повинен включати:

1. титульний лист із зазначенням номера та назви лабораторної роботи;
2. мета роботи;
3. тексти завдань та результати їх виконання;
4. вихідні тексти розроблених програм;
5. результати роботи розроблених програм;
6. висновки (що було зроблено, за допомогою яких засобів, що було вивчено тощо).

Контрольні питання

1. Якими мовами програмування можна розробляти .NET програми в Microsoft Visual Studio?
2. Що таке рішення у Visual Studio?
3. Що являє собою проект у Visual Studio?
4. У чому різниця між запусками програми клавішею F5 та комбінацією клавіш Ctrl-F5?
5. Що потрібно зробити, щоб при запуску на виконання запускався певний проект?
6. Для чого призначено діаграму класів?
7. Що таке клас?
8. Яке ім'я має бути у методу, який є точкою входу до програми C#?
9. Вкажіть синтаксис виклику методу, наведіть приклад.
10. За допомогою якого класу / яких методів реалізується виведення в консоль C#?

ЛАБОРАТОРНА РОБОТА №2 АБСТРАКТНІ СУТНОСТІ ТА ЗВ'ЯЗКИ МІЖ НИМИ. РОБОТА ІЗ БІБЛІОТЕКАМИ КЛАСІВ.

Мета роботи: Навчитися описувати об'єкти реального світу як абстрактних сутностей. Отримати практичні навички роботи з класами та об'єктами. Освоїти створення бібліотек класів. **Час проведення**: 4 години. **Місце проведення:** комп’ютерний клас.

Навчальні питання:

1. Необхідно розробити клас, що емулює роботу побутового приладу, а також програму, яка демонструє роботу з цим класом.

Література:

Основна література.

4. Коноваленко І.В. Програмування мовою C# 6.0. Тернопіль, ТНТУ, 2016. 227 с.
5. H. Schildt, “C# 4.0 The Complete Reference McGraw Hill Education; 1st edition” – 984 p., 2017.
6. J. Sharp, “Microsoft Visual C# Step by Step”, Pearson Education – 878p., 2018.

Допоміжна література.

1. A. Troelsen, “Pro C# 7 With .NET and .NET Core” - Apress, — 1372 p., 2017.
2. J. Albahari, B. Albahari, “C# 7.0 in a Nutshell”, O'Reilly Media— 1090 p., 2017.
3. Інструментальні програмні засоби розробки ІУС. Методичні вказівки до виконання лабораторних робіт. / уклад.: К.І. Київська–Київ: КНУБА, 2018. – 40 с

Інформаційні ресурси в Інтернеті

1. C# Programming. Yellow Book [Ел. ресурс]. URL: https://www.robmiles.com/c-yellow-book
2. Secure Coding Guidelines for .NET [Ел. ресурс]. URL: https://docs.microsoft.com/en-us/dotnet/standard/security/secure-coding-guidelines
3. C# Reference, сайт розробників MSDN. [Електронний ресурс] – Режим доступу: https://docs.microsoft.com/uk-ua/dotnet/csharp/languagereference/index
4. Об’єктно-орієнтоване програмування. Частина 1. Основи об’єктно-орієнтованого програмування на мові C#.: Навчальний посібник. / Д.В. Настенко, А. Б. Нестерко. – К.: НТУУ «КПІ», 2016. - 76с. [Електронний ресурс]. – Режим доступу https://ela.kpi.ua/bitstream/123456789/16671/1/OOP_manual.pdf

План проведення заняття:

Постановка задачі

Необхідно розробити клас, що емулює роботу побутового приладу, а також програму, яка демонструє роботу з цим класом.

Клас має бути реалізований в окремій бібліотеці класів. Програма повинна дозволяти створювати кілька приладів (не більше 5), виконувати через меню в консольній програмі ті чи інші дії з ними, наприклад, включати та

вимикати вибраний прилад, змінювати режим роботи тощо. При цьому після кожної дії користувача програма повинна відображати на екрані стан вибраного приладу та меню всіх можливих дій.

Завдання 1

Ознайомитись із постановкою завдання, за індивідуальним варіантом завдання (див. додаток А), проаналізувати предметну область та описати функціонування абстрактного побутового приладу відповідного типу, виділивши найбільш важливі функції, режими роботи та обмеження (деякі функції можуть бути недоступні у певних режимах роботи: наприклад, неможливо увімкнути мікрохвильову піч, не закривши дверцята).

Завдання 2

За описом, підготовленим у завданні 1, розробити необхідні класи, перерахування тощо, використовуючи на першому етапі діаграму класів (рисунок 1), а потім редактор коду.

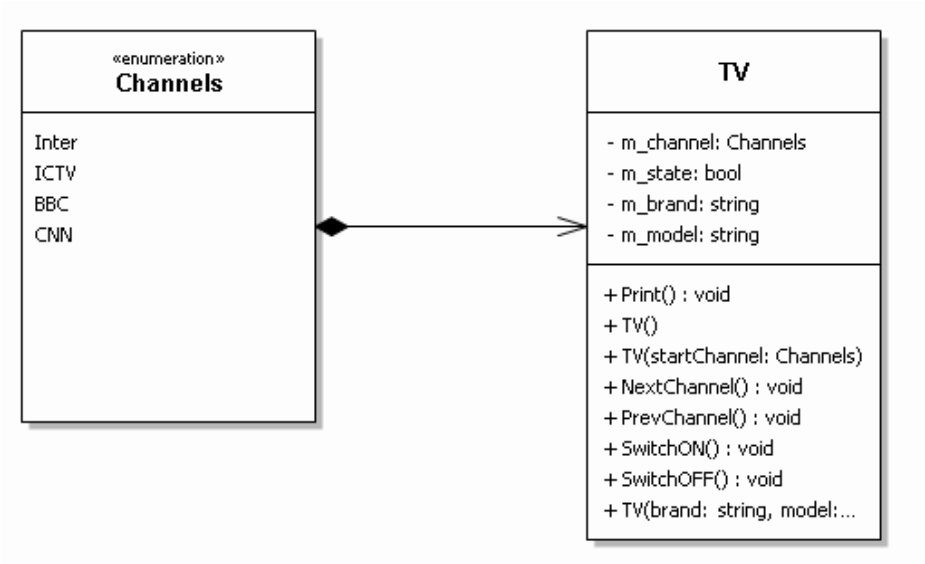


Рисунок 1 – Діаграма класів

Режими роботи побутового приладу мають бути реалізовані у вигляді перерахувань (enum).

Клас побутового приладу повинен мати метод, що повертає його поточний стан у вигляді рядка, а також методи, що дозволяють змінювати цей стан.

Клас повинен містити поля для зберігання бренду та моделі побутового приладу, оголошені з ключовим словом readonly (їх значення мають задаватися при створенні екземпляра), а також методи доступу до них.

```
/// <summary>
/// Каналы
/// </summary>
public enum Channels
{
    inter,
    ictv,
    tet,
    hromadske
}

/// <summary>
/// Класс описания телевизора
/// </summary>
public class TVset
{
    /// <summary>
    /// Текущий канал
    /// </summary>
    private Channels channel = Channels.inter;
    /// <summary>
    /// Признак включения питания
    /// </summary>
    private bool power = false;

    /// <summary>
    /// Переключение на заданный канал
    /// </summary>
    /// <param name="newChannel">Канал</param>
    public void SetChannel(Channels newChannel)
    {
        channel = newChannel;
```

Рисунок 3 – Фрагмент програми з документуючими коментарями

У класу має бути визначено кілька конструкторів, що дозволяють створити об'єкт із значеннями за промовчанням (конструктор без параметрів) або об'єкт із заданими значеннями (конструктор із параметрами). Як мінімум, мають бути реалізовані такі варіанти:

 конструктор без параметрів (створює екземпляр із брендом "Noname" і моделлю "Unknown");

 конструктор з параметром «бренд» (рядок) (створює екземпляр із заданим брендом та моделлю «Unknown»);

 конструктор з параметрами «бренд» (рядок) та «модель» (рядок) (створює екземпляр із заданим брендом та моделлю).

 конструктор з параметрами «бренд» (рядок), «модель» (рядок) та «Початковий стан» (перерахування) (створює екземпляр із заданим брендом і моделлю, переводить його в заданий початковий стан).

Клас повинен містити статичне поле (поле, оголошене з ключовим словом static), яке має збільшуватися при створенні кожного екземпляра та

використовуватися для створення унікального серійного номера побутового приладу.
Опис кожного класу має бути поміщений окремий файл.
Головний метод програми (Main) повинен виконувати завдання відображення меню та передачі команд користувача класу побутового приладу.
Усередині класу побутового приладу не повинно бути звернень до класу Console – стан приладу повинен повертатися у вигляді string та виводитися у консольне вікно вже у методі Main класу Program.

Кожен елемент класу ОБОВ'ЯЗКОВО має містити документуючий коментар, що описує його призначення (рисунок 3).
Текст програми повинен починатися з коментаря, в якому вказується номер та назва лабораторної роботи, номер завдання, а також прізвище студента:

Порядок виконання роботи

1. В Visual Studio створіть рішення (Solution) та додайте в нього 2 проекти – перший проект типу «Class Library» (рисунок 4), другий проект – типу «Console Application».

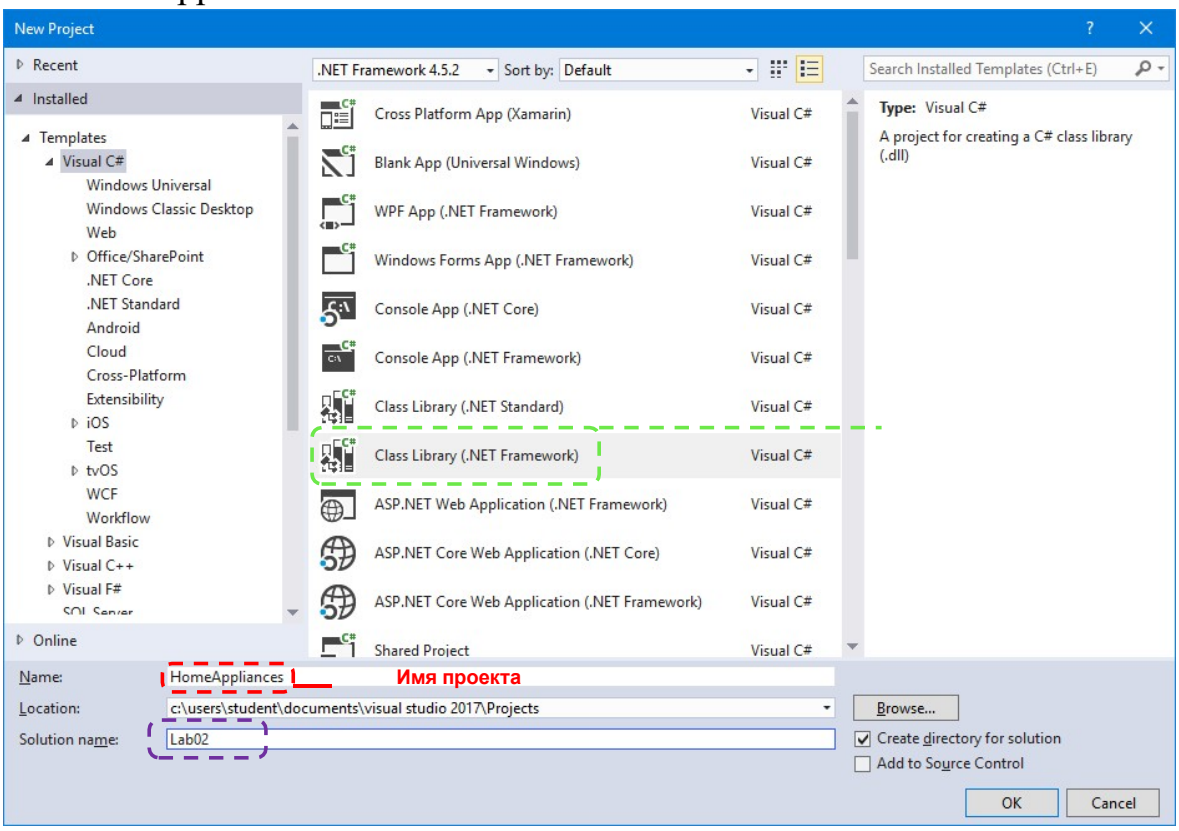
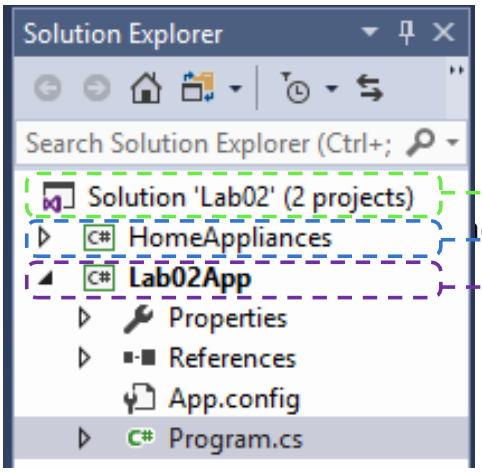


Рисунок 4 – Створення проекту типу «Class Library»

2. 2. У разі успішного виконання кроку 1 повинен бути одержаний результат, показаний на малюнку 5.



асів) Проект 2 (консольний додаток)

Рисунок 5 – Рішення з двома проектами

3. 3. Задайте створеному за умовчанням файлу Class1.cs ім'я, яке відповідає побутовому приладу за варіантом завдання, вибравши пункт Rename у контекстному меню (рисунок 6).

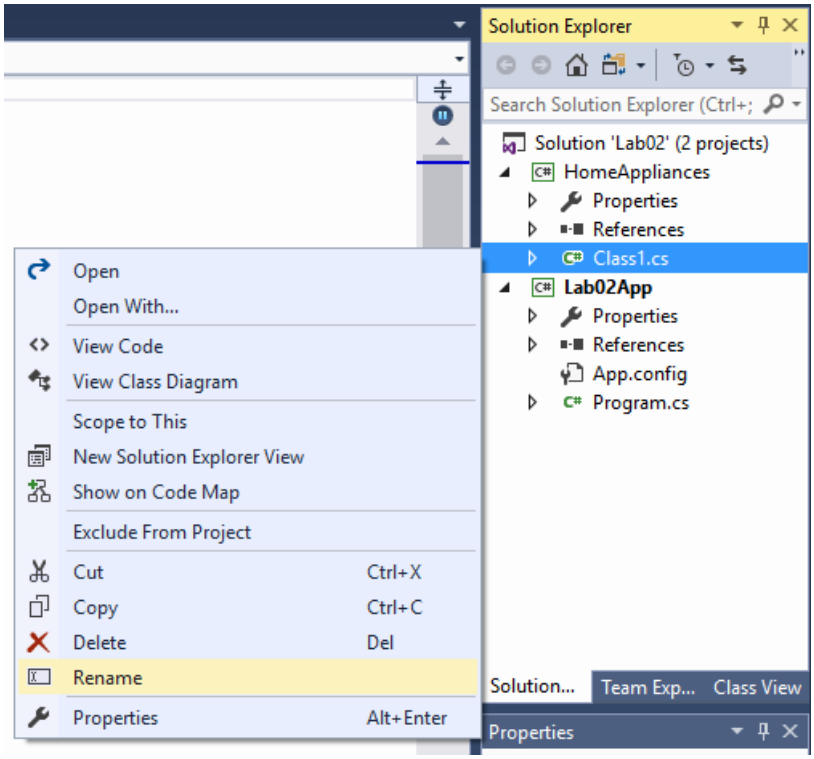


Рисунок 6 – Перейменування файла проекта

4. 4. Перейдіть до діаграми класів, вибравши відповідний пункт у контекстному меню (рисунок 7).

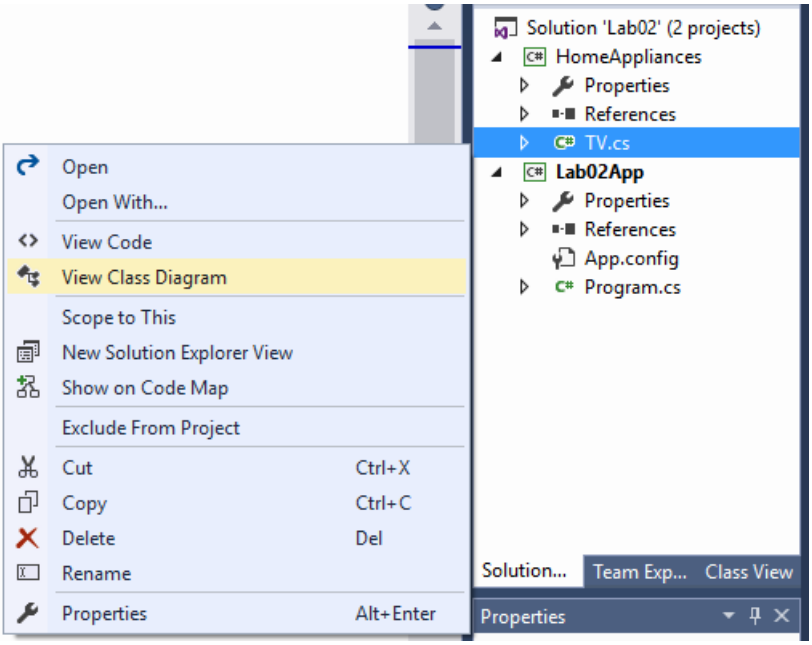


Рисунок 7 – Перехід до просмотру діаграми класів

5. 5. Використовуючи візуальний редактор, створіть необхідні поля та методи класу (рисунок 8).

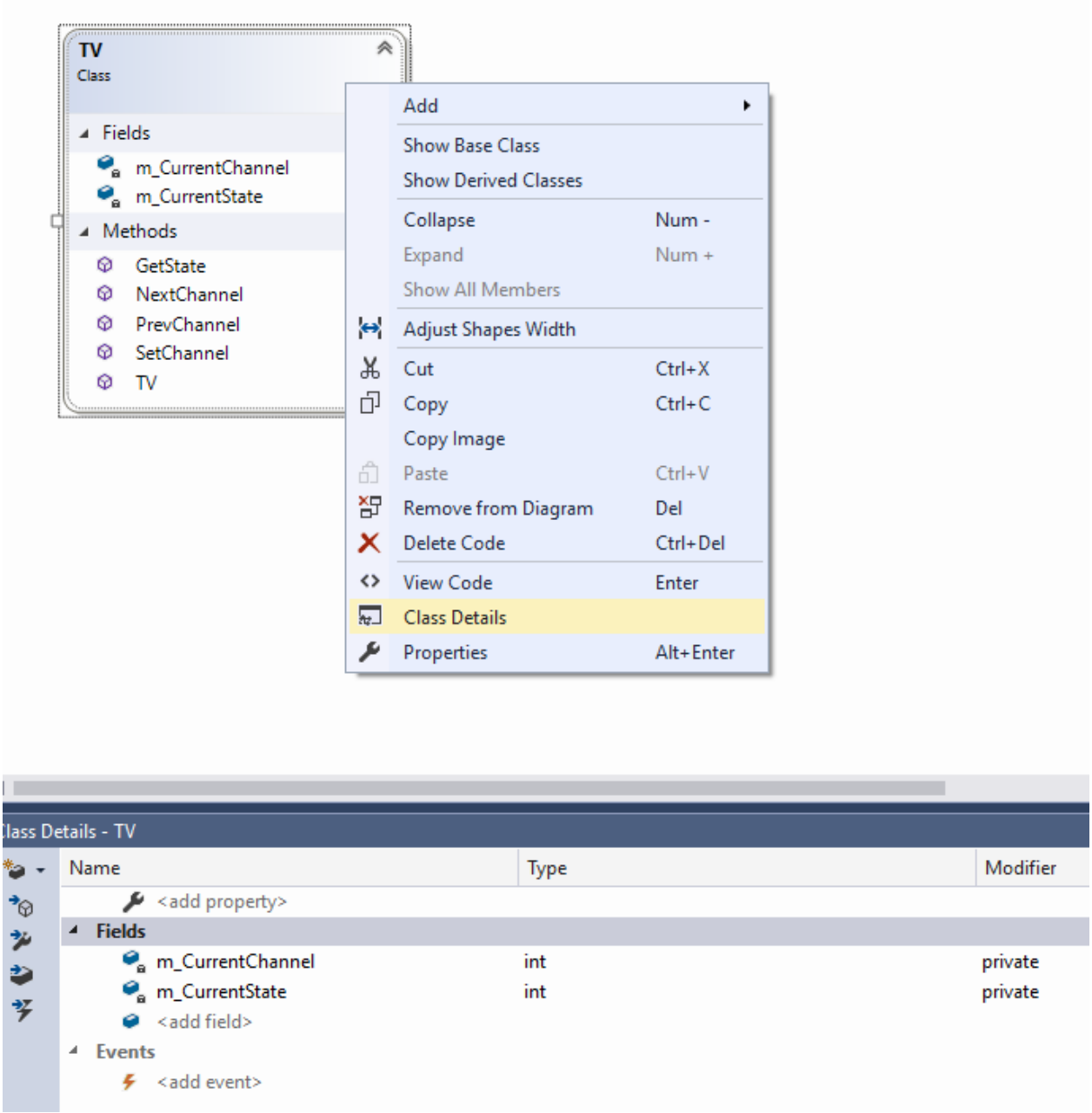


Рисунок 8 – Редагування членів класу на діаграмі класів

6. 6. Додайте на діаграму класів нову сутність – перерахування, що міститиме можливі режими роботи побутового приладу (рисунок 9).

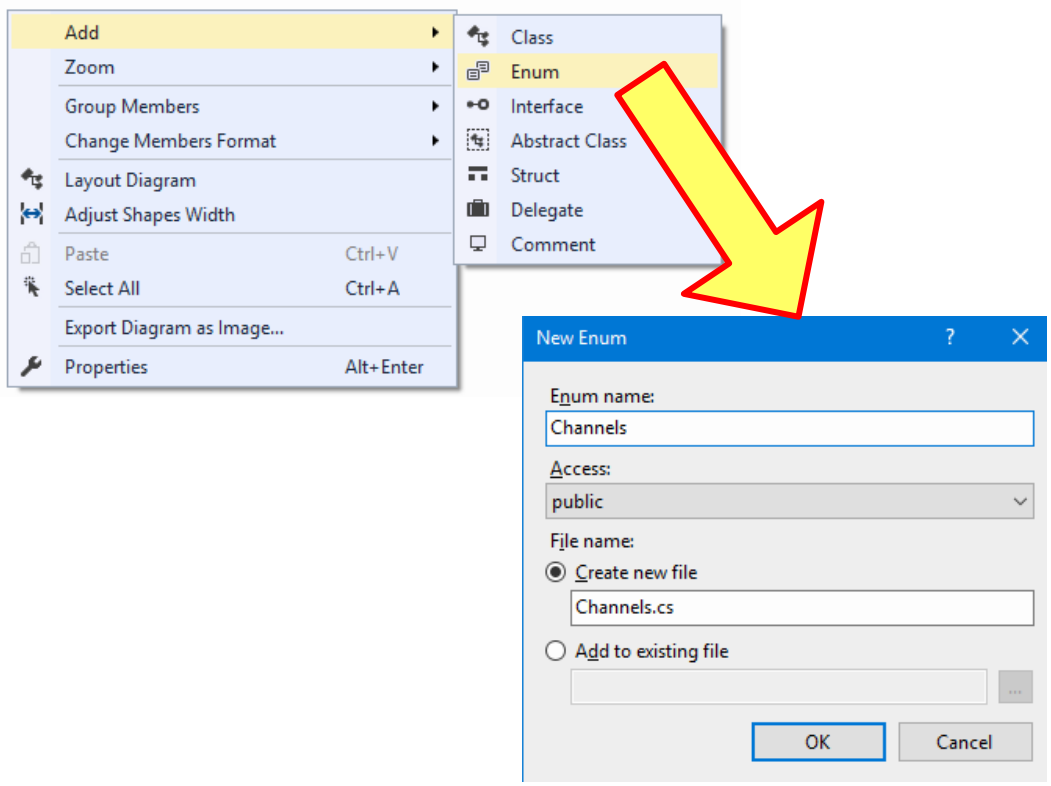


Рисунок 9 – Додавання нової сутності на діаграму класів

7. Додавання нової сутності на діаграму класів (рисунок 10).

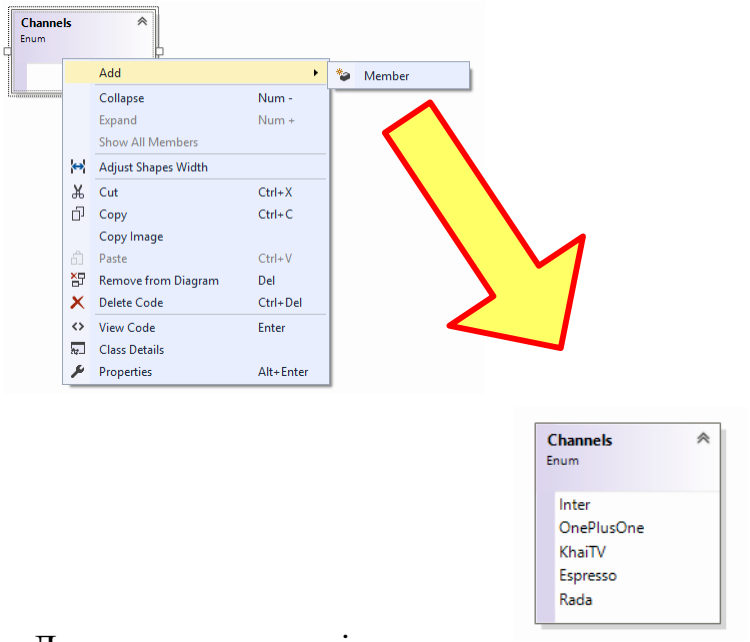


Рисунок 10 – Додавання елементів перерахування

8. Біля поля класу, яке призначене для зберігання стану побутового приладу, задайте як тип даних створене на попередньому кроці перерахування (рисунок 11).

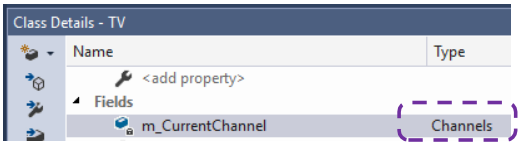


Рисунок 11 – Завдання типу даних біля поля класу

9. 9. Натисніть праву кнопку миші на полі класу, яке призначене для збереження стану побутового приладу, у контекстному меню виберіть «Show as Association» (рисунок 12).

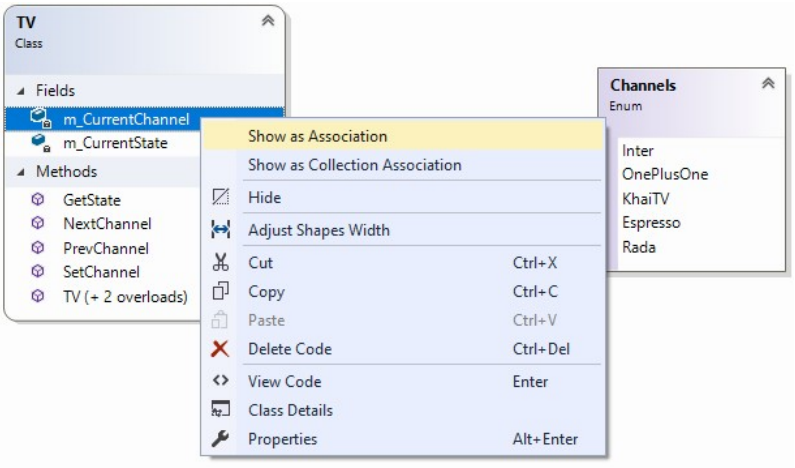


Рисунок 12 – Вибір пункта «Show as Association»

10. У результаті між створеними сутностями має з'явитися зв'язок «асоціація» (рис. 13). Асоціація означає відношення між сутностями, що дозволяє екземпляру однієї сутності викликати екземпляр іншої сутності.

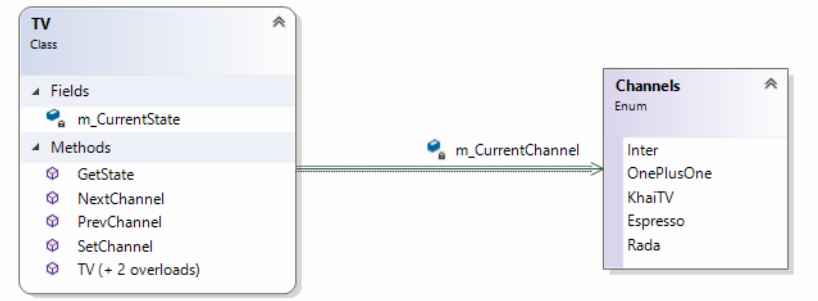


Рисунок 13 – Зв'язок «асоціація» між сутностями

11. Перейдіть до редактора коду проекту з бібліотекою класів, реалізуйте логіку роботи всіх методів/конструкторів класу побутового приладу.
12. Перейдіть до редактора коду проекту з консольною програмою. Для того, щоб у даному проекті можна було використовувати клас, описаний у бібліотеці класів, необхідно додати посилання на цю бібліотеку, вибравши пункт «Add Reference», далі вибравши «Project – Solution» та відзначивши галочкою потрібну бібліотеку (Рисунок 14).

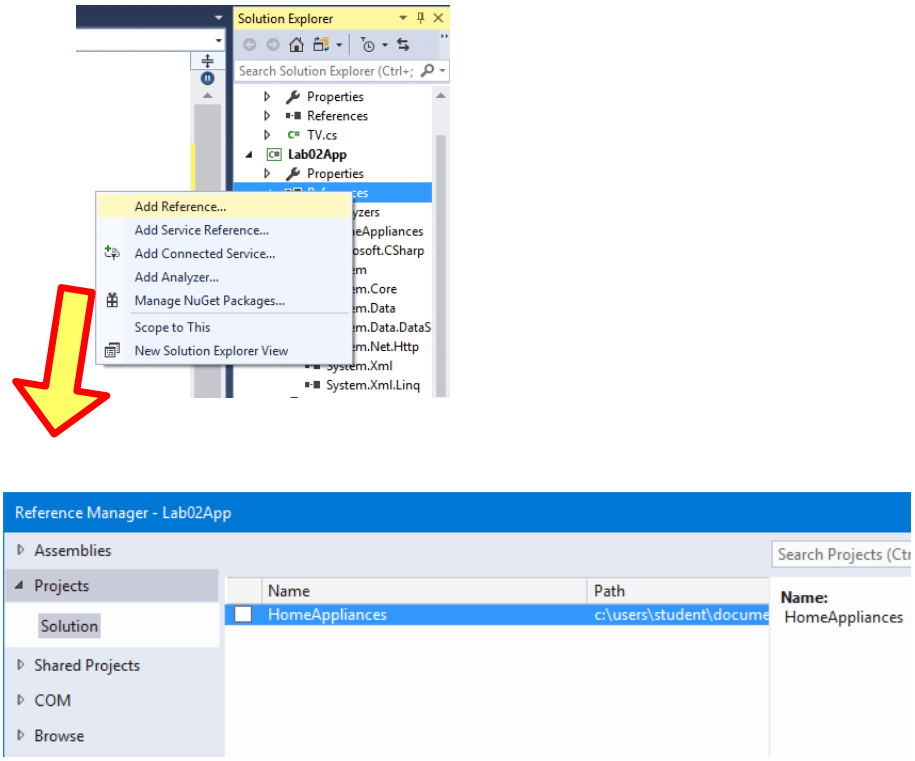


Рисунок 14 – Додавання посилання на бібліотеку класів

13. За допомогою ключового слова using підключіть простір імен, зазначений у бібліотеці класів. Наприклад, якщо простір імен називається HomeAppliances, необхідно вказати using HomeAppliances (Рисунок 15).

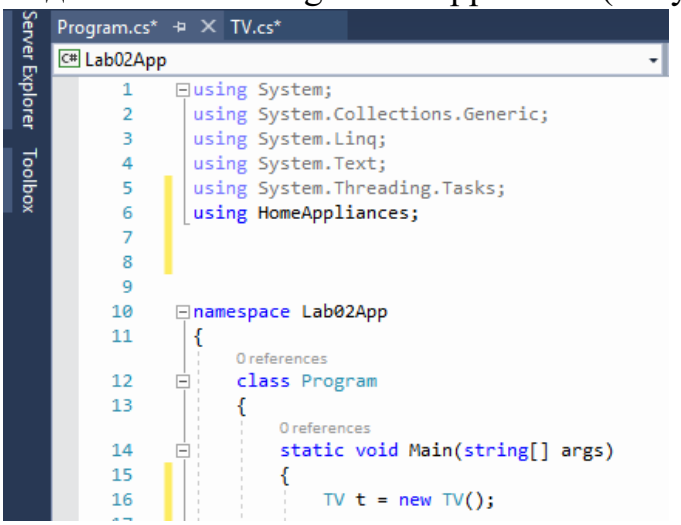


Рисунок 15 – Підключення простору імен

14. Якщо все було зроблено правильно, у методі Main класу Program можна створювати екземпляри класу побутового приладу.
15. Реалізуйте програму, яка показує роботу з класом побутового приладу. Повинна бути передбачена можливість створення кількох приладів (не більше 5) та роботи з ними.

Вимоги до звіту

Звіт з цієї лабораторної роботи повинен включати:

1. титульний лист із зазначенням номера та назви лабораторної роботи;
2. мета роботи;
3. тексти завдань та результати їх виконання;
4. текстове опис функціонування побутового приладу;
5. діаграму класів;
6. вихідні тексти розробленої програми;
7. результати роботи розробленої програми (скриншоти);
8. Висновки.

Контрольні питання

1. Що таке клас?
2. Що таке об'єкт?
3. Що таке поле?
4. Що таке метод?
5. Як пов'язані між собою класи та об'єкти у програмі?
6. Що таке перерахування?
7. Які члени класу вам відомі?
8. Які члени класу містять код?
9. Які члени класу містять дані?
10. У чому призначення конструктора?
11. Скільки конструкторів може містити клас?
12. У чому полягає призначення ключового слова readonly?
13. У чому призначення ключового слова static?
14. Наведіть синтаксис опису класу у загальному вигляді.
15. Які модифікатори доступу до членів класу Вам відомі?
16. Яке ключове слово мови C# використовується для створення екземпляра?
17. Що таке зв'язок «асоціація»?
18. Як додати посилання на бібліотеку класів? Які дії необхідно виконати, щоб можна було створювати екземпляри класів із підключеної бібліотеки?

Додаток А. Варіанти завдань

1. Пральна машина
2. ТБ
3. Холодильник
4. Мікрохвильова піч
5. Кавоварка
6. Хлібопічка
7. Радіоприймач
8. Електронний годинник

9. Кухонний комбайн
10. Духовка
11. Сушильна машина
12. Морозильна камера
13. Йогуртниця
14. Посудомийна машина
15. Мультиварка
16. Кондиціонер
17. Фен
18. Блендер
19. Обігрівач
20. Чайник
21. Метеостанція
22. Бойлер
23. М'ясорубка
24. Швейна машинка
25. Тостер
26. Фрітюрниця

ЛАБОРАТОРНА РОБОТА №3 УСПАДКУВАННЯ ТА ПОЛІМОРФІЗМ.

Мета роботи: Вивчити парадигми «успадкування» та «поліморфізм». Освоїти можливості мови C#, призначені для реалізації даних парадигм. **Час проведення**: 4 години. **Місце проведення:** комп'ютерний клас.

Навчальні питання:

1. Розробка бібліотеки класів.

Література:

Основна література.

7. Коноваленко І.В. Програмування мовою C# 6.0. Тернопіль, ТНТУ, 2016. 227 с.
8. H. Schildt, "C# 4.0 The Complete Reference McGraw Hill Education; 1st edition" – 984 p., 2017.
9. J. Sharp, "Microsoft Visual C# Step by Step", Pearson Education – 878p., 2018.

Допоміжна література.

1. A. Troelsen, “Pro C# 7 With .NET and .NET Core” - Apress, — 1372 p., 2017.
2. J. Albahari, B. Albahari, “C# 7.0 in a Nutshell”, O'Reilly Media— 1090 p., 2017.
3. Інструментальні програмні засоби розробки ІУС. Методичні вказівки до виконання лабораторних робіт. / уклад.: К.І. Київська–Київ: КНУБА, 2018. – 40 с

Інформаційні ресурси в Інтернеті

1. C# Programming. Yellow Book [Ел. ресурс]. URL: https://www.robmiles.com/c-yellow-book
2. Secure Coding Guidelines for .NET [Ел. ресурс]. URL: https://docs.microsoft.com/en-us/dotnet/standard/security/secure-coding-guidelines
3. C# Reference, сайт розробників MSDN. [Електронний ресурс] – Режим доступу: https://docs.microsoft.com/uk-ua/dotnet/csharp/languagereference/index
4. Об’єктно-орієнтоване програмування. Частина 1. Основи об’єктно-орієнтованого програмування на мові C#.: Навчальний посібник. / Д.В. Настенко, А. Б. Нестерко. – К.: НТУУ «КПІ», 2016. - 76с. [Електронний ресурс]. – Режим доступу https://ela.kpi.ua/bitstream/123456789/16671/1/OOP_manual.pdf

План проведення заняття:

Особливості лабораторної роботи

Лабораторна робота виконується командою із двох осіб. Кожен член команди реалізує завдання 1 для свого варіанту та передає результати другому члену команди для виконання ним завдання 2.

Завдання 1.

Відповідно до варіанта завдання (додаток А) створіть бібліотеку класів, в якій побудуйте ієрархію класів, враховуючи такі вимоги:

 базовий клас, що знаходиться на вершині ієрархії, має бути реалізований у вигляді абстрактного класу;

 у кожного класу має бути конструктор за замовчуванням та кілька конструкторів з параметрами;

 конструктори з параметрами повинні перевіряти коректність переданих параметрів та генерувати виняток, якщо параметри некоректні;

 конструктори похідних класів повинні викликати конструктори базових;

 у кожного класу має бути реалізований метод, який повертає максимум інформації про нього (значення всіх полів) у вигляді рядка; даний метод має бути визначений як virtual та перевизначений у похідних класах;

 у кожного класу має бути не менше 5 полів;

 всі поля повинні мати модифікатор доступу private або protected; якщо потрібна можливість зміни/набуття значення, повинні бути реалізовані властивості;

 властивості повинні перевіряти значення, що присвоюються їм, при спробі привласнити некоректне значення має бути згенеровано виняток;
 кожен елемент класу обов'язково має містити документуючий коментар, який описує його призначення (це необхідно для подальшої генерації документації).

За допомогою docfx (https://dotnet.github.io/docfx/) згенеруйте документацію з розробленої бібліотеки класів.

Передайте розроблену бібліотеку класів (.dll-файл) разом із документацією іншому члену команди.

Завдання 2.

Отримайте від іншого члена команди розроблену ним бібліотеку класів (.dll-файл) та документацію на неї.

Створіть консольну програму, додайте посилання на отриману бібліотеку класів.

Створіть масив типу «базовий клас», помістіть до масиву екземпляри всіх похідних класів ієрархії, використовуючи upcast. Дані для створення екземплярів зверніться до користувача. У циклі продемонструйте можливість виклику всіх методів/звернення до властивостей кожного елемента цього масиву, використовуючи downcast. Передбачте можливість обробки всіх можливих винятків (як генерованих у бібліотеці класів, так і стандартних).

Вимоги до звіту

Звіт з цієї лабораторної роботи повинен включати:

1. титульний лист із зазначенням номера та назви лабораторної роботи;
2. мета роботи;
3. текст завдання та результати його виконання;
4. вихідні тексти розробленої програми;
5. результати роботи розробленої програми;
6. діаграму класів;
7. документацію на розроблену бібліотеку класів;
8. документацію на бібліотеку класів, отриману з іншого члена команди;
9. тестові набори;
10. висновки.

Контрольні питання

1. Чим відрізняється приховування методів перевизначення?
2. Члени класів, з якими модифікаторами доступу доступні у похідних класах?
3. Що таке UpCast та DownCast?
4. Навіщо призначений конструктор?
5. Чи автоматично викликається конструктор базового класу при виклику конструктора похідного?

6. Чи дозволено множинне наслідування в C#?
7. Чи можна заборонити спадкування від класу? Яким чином?
8. Чи можна заборонити перевизначення методу? Яким чином?
9. Що повертає оператор is, якщо об'єкт може бути наведений до заданого типу?
10. Навіщо призначений оператор as? 11. Що таке абстрактний клас?
12. Чи може абстрактний клас містити поля/методи? 13. Чи можна створювати екземпляри абстрактного класу? 14. Що таке виняток?
15.Для чого призначена обробка винятків? 16.Як можна обробити винятки всіх типів?
17.Як можна обробити лише певний тип винятків?

Додаток А. Варіанти завдань

1. Електронний компонент, ПЛІС, мікропроцесор
2. Лікарський засіб, пігулки, сироп
3. Автоелектроніка, відеореєстратор, GPS-навігатор
4. Друковане видання, книга, журнал
5. Населений пункт, місто, селище міського типу
6. Товар, ваговий товар, поштучний товар
7. Документ, довідка, звіт
8. Транспортний засіб, автомобіль, мотоцикл
9. Мережеве обладнання, маршрутизатор, комутатор
10. Накопичувач, SSD-диск, мережевий накопичувач (NAS)
11. Геометрична фігура, коло, квадрат
12. Меблі, крісло, шафа
13. Будівля, житловий будинок, офісний центр
14. Навчальний заклад, школа, університет
15. Торговий заклад, супермаркет, ларьок
16. Програмне забезпечення, антивірус, архіватор
17. Побутовий прилад, праска, телевізор
18. Нерухомість, квартира, будинок
19. Рахунок у банку, кредитний рахунок, поточний рахунок
20. Громадський транспорт, трамвай, маршрутне таксі
21. Літак, вантажний літак, пасажирський літак
22. Портативний пристрій, ноутбук, планшет
23. Флеш-пам'ять, картка microSD, USB-накопичувач
24. Касове обладнання, лічильник банкнот, детектор валют
25. Аксесуар для планшета, чохол, захисна плівка
26. Персона, студент, викладач

ЛАБОРАТОРНА РОБОТА №4 ПЕРЕВАНТАЖЕННЯ МЕТОДІВ ТА ОПЕРАТОРІВ. ІНТЕРФЕЙСИ.

Мета роботи: Вивчити можливості перевантаження методів і операторів, що надаються мовою C#. Вивчити призначення інтерфейсів. **Час проведення**: 4 години. **Місце проведення:** комп'ютерний клас.

Навчальні питання:

1. Вивчити призначення інтерфейсів.
2. Навчитися описувати класи та інтерфейси

Література:

Основна література.

10.Коноваленко І.В. Програмування мовою C# 6.0. Тернопіль, ТНТУ, 2016. 227 с.

11.H. Schildt, "C# 4.0 The Complete Reference McGraw Hill Education; 1st edition" – 984 p., 2017.

12.J. Sharp, "Microsoft Visual C# Step by Step", Pearson Education – 878p., 2018.

Допоміжна література.

1. A. Troelsen, "Pro C# 7 With .NET and .NET Core" - Apress, — 1372 p., 2017.

2. J. Albahari, B. Albahari, "C# 7.0 in a Nutshell", O'Reilly Media— 1090 p., 2017.

3. Інструментальні програмні засоби розробки ІУС. Методичні вказівки до виконання лабораторних робіт. / уклад.: К.І. Київська–Київ: КНУБА, 2018. – 40 с

Інформаційні ресурси в Інтернеті

1. C# Programming. Yellow Book [Ел. ресурс]. URL: https://www.robmiles.com/c-yellow-book
2. Secure Coding Guidelines for .NET [Ел. ресурс]. URL: https://docs.microsoft.com/en-us/dotnet/standard/security/secure-coding-guidelines
3. C# Reference, сайт розробників MSDN. [Електронний ресурс] – Режим доступу: https://docs.microsoft.com/uk-ua/dotnet/csharp/languagereference/index
4. Об'єктно-орієнтоване програмування. Частина 1. Основи об'єктно-орієнтованого програмування на мові C#.: Навчальний посібник. / Д.В.

Настенко, А. Б. Нестерко. – К.: НТУУ «КПІ», 2016. - 76с. [Електронний ресурс]. – Режим доступу https://ela.kpi.ua/bitstream/123456789/16671/1/OOP_manual.pdf

План проведення заняття:

Завдання 1

Розробити бібліотеку класів із реалізованим у ній класом «математична абстракція» згідно з варіантом завдання (див. Додаток А), враховуючи такі вимоги:

- клас повинен містити не менше трьох конструкторів (без параметрів, з різним набором параметрів тощо), а також усі методи та оператори, наведені на UML-діаграмі у завданні;
- у разі передачі неприпустимих значень мають бути згенеровані винятки;
- у класі має бути реалізований оператор неявного перетворення на рядок, завдання якого – перетворити на рядок поточний стан екземпляра;
- у класі має бути реалізований оператор явного перетворення на тип даних double.

Розробити консольну програму, яка повинна забезпечувати введення вихідних даних, створення як мінімум двох екземплярів розробленого класу бібліотеки класів, вибір та виконання допустимих для цих екземплярів операцій, виведення на екран результатів виконання операцій.

Реалізувати обробку винятків у основній програмі.

Скласти тестові набори для перевірки правильності функціонування всіх методів та операторів розробленого класу "математичної абстракції".

Кожен елемент класу повинен містити документуючий коментар (///), який описує його призначення.

Завдання 2

Виходячи з постановки задачі, згідно з варіантом (див. Додаток Б), описати всі класи та інтерфейси (дати назви для класів/інтерфейсів, скласти перелік членів класів/інтерфейсів надавши їм імена, встановити зв'язки між класами та інтерфейсами). Скласти діаграму класів у NClass або Visual Studio.

Розробити програму з реалізованими в ній класами відповідно до завдання (кількість класів та інтерфейсів не повинна бути меншою від визначеного в заданні числа, не повинна порушуватися функціональність інтерфейсів та класів).

Вимоги до звіту

Звіт з цієї лабораторної роботи повинен включати:

1. титульний лист із зазначенням номера та назви лабораторної роботи;
2. мета роботи;
3. тексти завдань та результати їх виконання;
4. діаграми класів;

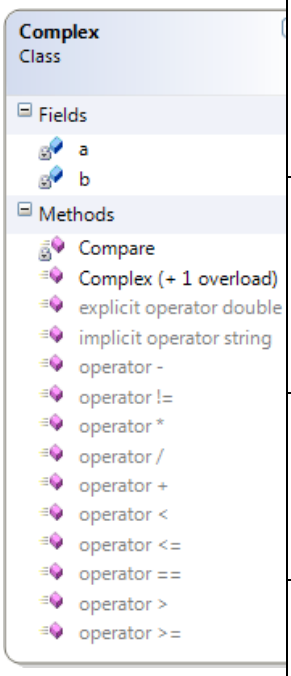
5. вихідні тексти розроблених програм;
6. результати роботи розроблених програм;
7. тестові набори;
8. Висновки.

Контрольні питання

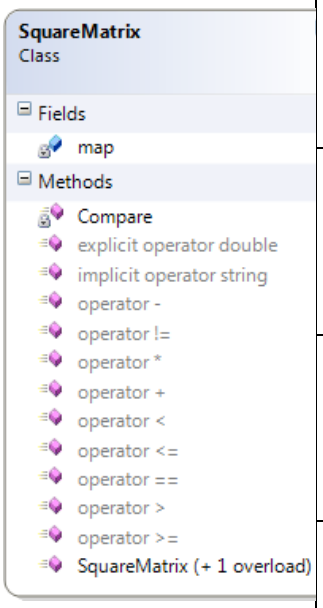
14. 1. Що таке конструктор?
15. 2. Чи може клас мати кілька конструкторів?
16. 3. Що таке конструктор за замовчуванням?
17. 4. Чи можна викликати з одного конструктора інші?
18. 5. Що таке деструктор?
19. 6. Навіщо призначений статичний конструктор?
20. 7. Чи може клас мати два конструктори з однаковим набором параметрів?
21. 8. Для чого призначене навантаження методів?
22. 9. Для чого призначене навантаження операторів? 10. Чи можна перевантажити оператор + кілька разів?
23. 11. Чи можна перевантажити оператор ==, а оператор != не перевантажувати? 12. Які різновиди операторів перетворення типів існують? 13. Що таке інтерфейс?
24. 14. Що може містити інтерфейс?
25. 15. У чому відмінності інтерфейсів та абстрактних класів?

Приложение А. Варианты заданий для задания 1

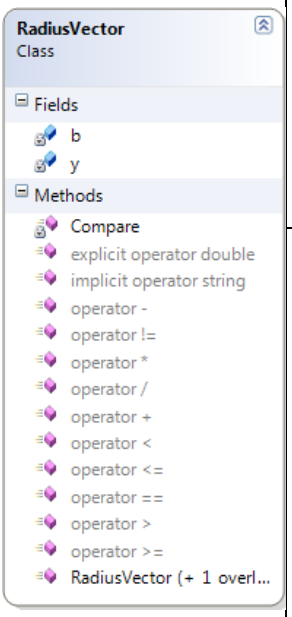
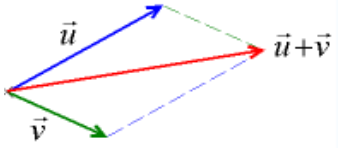
Варіант 1

Математична абстракція	Клас в UML	Логіка роботи операторів
Комплексне число (пара дійсних чисел, що позначають реальну A і уявну частину B) виду A + Bi, де i - уявна одиниця, тобто одне з чисел, що задовольняють рівняння i2 = -1		***Додавання*** (a + bi) + (c + di) = (a + c) + (b + d)i ***Віднімання*** (a + bi) − (c + di) = (a − c) + (b − d)i ***Множення*** (a + bi)(c + di) = ac + bci + adi + bdi^2 = (ac − bd) + (bc + ad)i ***Ділення*** (a □ bi) □ □ ac □ bd □ □ □ bc □ ad □ (c □ di) □ c2 □ d2 □ □ c2 □ d2 □ □ □ □ □ ***Порівняння***: числа (a + bi) и (c + di) рівні, если a2 □ b2 □ c2 □ d2 (a + bi) > (c + di), если a2 □ b2 □ c2 □ d2 и т.д. ***Перетворення в double*** реалізувати у вигляді повернення дійсної частини без уявної

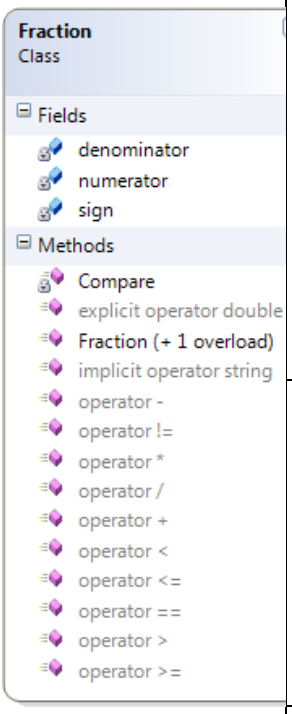
Варіант 2

Математична абстракція	Клас в UML	Логіка роботи операторів
Квадратна матриця порядку 3 (матриця з кількістю рядків 3 та кількістю стовпців 3) – реалізує зберігання даних у двовимірному масиві дійсних чисел		***Додавання*** сума відповідних елементів кожної з матриць аргументів
		Віднімання різницю відповідних елементів кожної з матриць аргументів
		Множення множення відповідних елементів кожної з матриць аргументів
		Порівняння матриць провадиться за значенням визначника. Більше та матриця, у якої він більший.
		Перетворення в double реалізувати у вигляді обчислення визначника

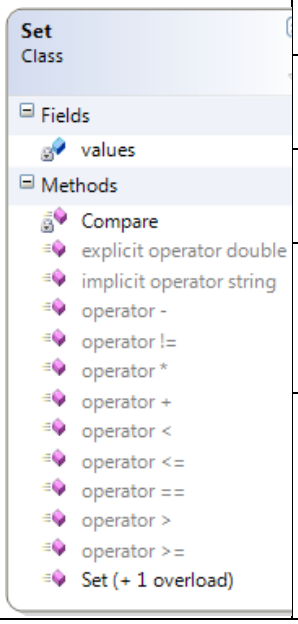
Варіант 3

Математична абстракція	Клас в UML	Логіка роботи операторів
Радіус-вектор – вектор, що йде з початку координат у точку з координатами X та Y, заданими речовими числами		***Додавання*** векторів за правилом паралелограма 
		Віднімання векторів зводиться до суми u □ □□v□ . Другий доданок □□v□ є вектором, протилежним вектору v за напрямом, але рівним йому за довжина.
		Множення вектора на число v k за правилом скалярного твору
		Ділення вектора на число зводиться до множення v 1 k
		Порівняння векторів виконується за довжиною. Більше той вектор, довжина якого більша.
		Перетворення в double реалізувати у вигляді обчислення довжини вектора

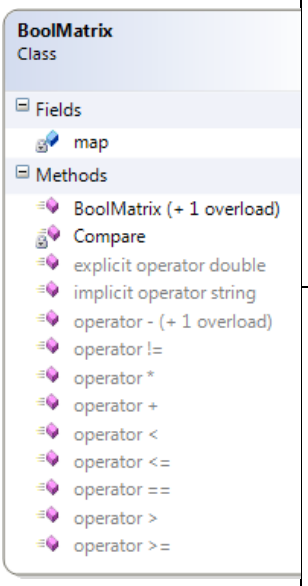
Варіант 4

Математична абстракція	Клас в UML	Логіка роботи операторів
Раціональне дробом		***Додавання*** сума відповідних раціональних чисел Додатково реалізувати можливість складання з цілими числами
		Віднімання різницю відповідних раціональних чисел Додатково реалізувати можливість віднімання з цілими числами
		Мнження твір відповідних раціональних чисел
		Ділення відношення відповідних раціональних чисел
		Порівняння раціональних чисел реалізувати шляхом порівняння відповідних їм дійсних чисел (чисел з плаваючою точкою)
		Перетворення в double реалізувати у вигляді обчислення відношення чисельника до знаменника

Варіант 5

<table>
<tr><th>Математична абстракція</th><th>Клас в UML</th><th>Логіка роботи операторів</th></tr>
<tr><td rowspan="5">Множество – совокупность элементов – реализовать в виде одномерного массива целых чисел</td><td rowspan="5"> 
</td><td>Додавання
об'єднання множин</td></tr>
<tr><td>Віднімання
різниця множин</td></tr>
<tr><td>Множення
перетин множин</td></tr>
<tr><td>Порівнянн множин реалізувати шляхом порівняння всіх їх елементів</td></tr>
<tr><td>Пеетвреня в double
реалізувати у вигляді обчислення суми всіх елементів множини</td></tr>
</table>

Варіант 6

Математична абстракція	Клас в UML	Логіка роботи операторів
Булева матриця		***Додавання*** логічне додавання (диз'юнкція) відповідних елементів кожної з булевих матриць (якщо розміри матриць різні, генерувати виняток)
		Віднімання логічне множення (кон'юнкція) відповідних елементів кожної з булевих матриць (якщо розміри матриць різні, генерувати виняток)
		Унарний мінус інверсія всіх елементів матриці
		Порівняння булевих матриць реалізувати шляхом порівняння всіх їх елементів (якщо розміри матриць різні, генерувати виняток)
		Переторення в double реалізувати як обчислення суми всіх елементів

Примітка

Закритий метод Compare у всіх варіантах завдань має реалізувати порівняння двох екземплярів:

- першого, для якого викликається метод;
- другого, який задано аргументом.

Таким чином, метод повинен мати такий вигляд:

```
private int Compare(имя_вашего_класса val)
{
// здесь сравниваются данные текущего экземпляра и
```

```
// экземпляра переданного через параметр val
// функция возвращает: 0 в случае равенства экземпляров,
// отрицательно число – если текущий экземпляр меньше val,
// положительное число – если текущий экземпляр больше val
}
```

Далі цей метод має викликатись методами операцій порівняння (operator!=(), operator==(), operator>() і т.д.), які перетворюють ціле значення, що повертається методом Compare, у логічне значення (bool), наприклад:

```
public static bool operator<(SquareMatrix first, SquareMatrix second)
{
return first.Compare(second) < 0;
}
```

Додаток Б. Варіанти завдань для завдання 2

Варіант №1

Розробити програму, що емулює роботу персональних комп'ютерів (клас №1) різних типів (класи, похідні від класу)
№1): настільний комп'ютер – desktop (клас №2), переносний комп'ютер – laptop (клас №3), планшет – tablet (клас №4).
Будь-який із пристроїв, незалежно від типу, потрібно створити (оператор new у тексті програми), визначивши йому назву (для цього потрібно реалізувати конструктор з параметром типу рядок). Створений пристрій можна включати (метод класу №1), вимикати (метод класу №1) та використовувати у включеному стані (метод класу №1). При емуляції роботи (виклику методу «використовувати») перевіряється, чи увімкнено комп'ютер, і якщо увімкнено, то використання призводить до витрати енергії та виведення на екран поточного стану пристрою (назва пристрою та стан його елементів – батареї та монітора).
Для комп'ютерів різного типу визначено різноманітні можливості. У настільного та переносного комп'ютерів можна змінити стан дисплея (інтерфейс №1), тобто. включати (перший метод в інтерфейсі №1) та вимикати (другий метод в інтерфейсі №1). Для планшета і переносного комп'ютера є можливість використання вбудованого джерела живлення (інтерфейс №2), тобто. заряджати батарею (перший метод в інтерфейсі №2) та перевіряти її стан (другий метод в інтерфейсі №2 – функція, що повертає ціле значення, що характеризують відсоток зарядки).
У програмі має бути не менше 4 класів та двох інтерфейсів, а також клас з функцій Main, що реалізує демонстраційні функції.

Після запуску програми має бути створений масив не менше ніж із п'яти елементів типу «клас №1», при ініціалізації елементів цього масиву повинні створюватися екземпляри типів «клас №2», «клас №3», "Клас №4".
Користувачеві програми за допомогою меню повинні бути доступні функції перегляду стану всіх пристроїв, увімкнення та вимкнення окремих пристроїв за їх номером у масиві, увімкнення та вимкнення дисплеїв (для пристроїв, що підтримують інтерфейс №1), підзарядки батарей (для пристроїв, що підтримують інтерфейс №2), а також емуляція часових відліків – «експлуатація» (при виборі цього пункту меню, всі включені комп'ютери витрачають енергію, для мобільних пристроїв це відображається у зменшенні зарядки батарей, тобто зміні поля, що зберігає поточний відсоток зарядки).

Варіант №2

Розробити програму, що керує банківськими рахунками (клас №1) різних типів (класи, які успадковуються від класу №1): депозитний рахунок (клас №2), поточний рахунок (клас №3), картковий рахунок (клас №4). Кожен банківський рахунок характеризується поточним балансом (поле у класі №1) та власником (поле у класі №1), ці дані встановлюються при відкритті рахунку (метод у класі №1 – конструктор з параметрами) та використовуються іншими операціями, у тому числі переглядом поточного стану (метод у класі №1, що повертає поточний баланс), визначення власника (метод у класі №1) та операцією закриття рахунку (метод у класі №1 – метод, що повертає суму, зняту з рахунку та обнулюючий баланс). Рахунками кожного типу визначено свій перелік можливих операцій. Для депозитних та карткових рахунків забезпечується нарахування відсотків (інтерфейс №1), тобто. реалізовано функції встановлення відсоткової ставки (метод в інтерфейсі №1, що змінює спеціальні поля у класах №2 та №4) та нарахування відсотків (метод в інтерфейсі №1 – збільшення балансу відповідно до поточного балансу та поточної відсоткової ставки). Для поточних та карткових рахунків доступні операції вільного доступу до засобів (інтерфейс №2): покласти на рахунок (метод в інтерфейсі №2) та зняти з рахунку (метод в інтерфейсі №2).
Таким чином, у програмі має бути реалізовано не менше 4 класів (три з яких є похідними від одного) та два інтерфейси. Крім того, необхідно створити клас з таким методом Main, який дозволяв би за допомогою системи меню створювати рахунки (елементи в масиві екземплярів) та керувати ними. Необхідно забезпечити можливість створення щонайменше 5 (наприклад, масив елементів типу клас №1) рахунків різних типів (оператор вибирає тип рахунки за його створенні).

Передбачити можливість перегляду стану всіх рахунків та автоматичного нарахування відсотків за всіма рахунками, які підтримують таку можливість.

Варіант №3

Розробити програму – спрощений емулятор побутових приладів. Побутовий прилад у програмі має бути реалізований у вигляді класу (клас №1), що дозволяє виводити на екран назву приладу та його поточний стан (метод класу №1).
Реалізувати класи з мінімальною функціональністю для приладів трьох типів: електролампи з регулюванням яскравості (клас №2), телевізор (клас №3) та радіоприймач (клас №4).
Будь-який прилад можна включати (метод класу №1) та вимикати (метод класу №1). Однак, призначення приладу визначає інші функції. Зокрема, для і телевізора необхідно забезпечити можливість регулювання гучності звуку (інтерфейс №1) , тобто. реалізувати функції підвищення гучності (метод в інтерфейсі №1) та зниження гучності (другий спосіб в інтерфейсі №1). А електролампи і телевізора має бути реалізована можливість зміни яскравості (метод в інтерфейсі №2), тобто. функції збільшення (метод в інтерфейсі №2) та зменшення яскравості (другий метод в інтерфейсі №2).
Для збереження поточних налаштувань яскравості та гучності можна використовувати поля цілого типу, значення яких можна виводити на екран як стан приладу разом із назвою приладу (рядком, що містить назву, задану при створенні екземпляра оператором new – для цього потрібно забезпечити конструктор з параметром).
Таким чином, необхідно реалізувати 4 класи для побутових приладів (базовий та три похідних від нього) та два інтерфейси, а також клас з методом Main для створення масиву з п'яти приладів (тип елементів – клас №1) та організації меню, що дозволяє працювати з приладами: включати та вимикати окремі прилади, регулювати гучність та яскравість окремих приладів за номером у масиві або всіх приладів, що підтримують ці функції.

ЛАБОРАТОРНА РОБОТА №5 КОЛЕКЦІЇ.

Мета роботи: Вивчити різновиди та призначення колекцій, дослідити особливості додавання, видалення, сортування значень у колекціях. Освоїти створення власних колекцій. **Час проведення**: 4 години. **Місце проведення:** комп’ютерний клас.

Навчальні питання:

1. Навчитися створювати власні коллекції.

Література:

Основна література.

13.Коноваленко І.В. Програмування мовою C# 6.0. Тернопіль, ТНТУ, 2016. 227 с.

14.H. Schildt, “C# 4.0 The Complete Reference McGraw Hill Education; 1st edition” – 984 p., 2017.

15.J. Sharp, “Microsoft Visual C# Step by Step”, Pearson Education – 878p., 2018.

Допоміжна література.

1. A. Troelsen, “Pro C# 7 With .NET and .NET Core” - Apress, — 1372 p., 2017.

2. J. Albahari, B. Albahari, “C# 7.0 in a Nutshell”, O'Reilly Media— 1090 p., 2017.

3. Інструментальні програмні засоби розробки ІУС. Методичні вказівки до виконання лабораторних робіт. / уклад.: К.І. Київська–Київ: КНУБА, 2018. – 40 с

Інформаційні ресурси в Інтернеті

1. C# Programming. Yellow Book [Ел. ресурс]. URL: https://www.robmiles.com/c-yellow-book
2. Secure Coding Guidelines for .NET [Ел. ресурс]. URL: https://docs.microsoft.com/en-us/dotnet/standard/security/secure-coding-guidelines
3. C# Reference, сайт розробників MSDN. [Електронний ресурс] – Режим доступу: https://docs.microsoft.com/uk-ua/dotnet/csharp/languagereference/index
4. Об’єктно-орієнтоване програмування. Частина 1. Основи об’єктно-орієнтованого програмування на мові C#.: Навчальний посібник. / Д.В. Настенко, А. Б. Нестерко. – К.: НТУУ «КПІ», 2016. - 76с. [Електронний ресурс]. – Режим доступу https://ela.kpi.ua/bitstream/123456789/16671/1/OOP_manual.pdf

План проведення заняття:

Завдання 1

Реалізувати власну колекцію у вигляді узагальненого типу (параметризованого класу) CollectionType<T>. Визначити конструктори, методи додавання та видалення елементів, інші необхідні методи та, якщо потрібно, перевантажені операції. Визначити індексатори та властивості. CollectionType можна реалізувати на основі стандартних колекцій (ArrayList, List, Stack, Array тощо). Передбачити обробку можливих винятків.

Завдання 2

Реалізувати у класах, розроблених у лабораторній роботі №3, інтерфейс IComparable<T>. Реалізація даного інтерфейсу дозволить сортувати об'єкти в колекціях за заданими Вами правилами.

Завдання 3

Розробити програму, у якій реалізувати необхідну функціональність відповідно до варіанта завдання (додаток А).

Завдання 4

Розробити програму, в якій оголосити масив, стандартні типізовану та нетипізовану колекції, а також користувальницьку колекцію об'єктів класів з лабораторної роботи №3:

```
int amount; // кількість елементів
... // запросити кількість елементів у користувача
// колекція нетипизированная
ArrayList alis = new ArrayList(amount);
// колекція типизированная
List<YourClass> lis = new List<YourClass>(amount);
// масив
YourClass[] arr = new YourClass[amount];
// колекція користувача
CollectionType<YourClass>     col     =     new     CollectionType
<YourClass>(amount);
```

Заповнити всі колекції випадковими даними. Провести експеримент із сортуванням одних і тих самих даних, поміщених у звичайний масив та три варіанти колекцій.

Для підрахунку часу, що витрачається на операцію сортування, можна використовувати один із таких способів:

Спосіб 1 (більш простий)	Спосіб 2 (більш точний)

`// засікти час початку`
`DateTime start = DateTime.Now;`
`// виконати операцію`
`DoSomething();`
`// засікти час закінчення`
`DateTime finish =`
`DateTime.Now;`
`// обчислити часовий інтервал`
`TimeSpan duration = finish - start;`
	`using System.Diagnostics;`
	`...`
	`Stopwatch sw = new Stopwatch();`
	`// засікти час початку`
	`sw.Start();`
	`// виконати операцію`
	`DoSomething();`
	`// засікти час закінчення`
	`sw.Stop();`
	`// обчислити часовий інтервал`
	`TimeSpan duration = sw.Elapsed;`

Включити у звіт результати порівняння виконання сортування для 1000, 1000000, 2000000 та 4000000 елементів. Зробити висновки.

Кожен елемент кожного класу повинен містити документуючий коментар (///), який описує його призначення.

Вимоги до звіту

Звіт з цієї лабораторної роботи повинен включати:

1. титульний лист із зазначенням номера та назви лабораторної роботи;

2. мета роботи;

3. діаграму класів;

4. вихідні тексти розроблених програм;

5. результати роботи розроблених програм (скриншоти або тексти, що виводяться програмами);

6. тестові набори (обов'язково включити тестові набори, які призводять до появи винятків);

7. висновки, у яких необхідно відобразити призначення всіх розроблених класів, реалізованих інтерфейсів та використаних колекцій.

Контрольні запитання

1. Що таке колекція?
2. Які колекції доступні у .NET Framework?
3. У чому полягає відмінність колекцій ArrayList та List<T>?
4. Які інтерфейси реалізують класи колекцій?

5. Елементи якого типу можна поміщати до колекції ArrayList?
6. Що таке упаковка та розпакування?
7. Які елементи містять інтерфейс IComparable?
8. Які елементи містять інтерфейс IComparer?
9. В якому порядку розміщуються в пам'яті елементи, які розміщені в Hashtable?
10. В якому порядку розміщуються в пам'яті елементи, які розміщуються в SortedList?

Додаток А. Варіанти завдань для завдання 3

№	***Коллекция***
1.	Створити масив об'єктів CollectionType. Реалізувати методи: знаходження колекції розміру n; – знаходження максимальної та мінімальної колекції в масиві за кількістю елементів
2.	Створити масив об'єктів CollectionType. Реалізувати методи: знаходження колекції з негативними елементами (вибрати будь-яке поле об'єкта); – знаходження максимальної та мінімальної колекції в масиві, що містять зазначений елемент.
3.	Створити масив об'єктів CollectionType. Реалізувати методи: знаходження кількості колекцій рівних заданому розміру; – знаходження максимальної та мінімальної колекції в масиві.
4.	Створити масив об'єктів CollectionType. Реалізувати методи: знаходження кількості колекцій, що містять лише 2 елементи; – знаходження максимальної та мінімальної колекції в масиві за заданим значенням поля об'єкта (можна вибрати будь-яке поле).
5.	Створити масив об'єктів CollectionType. Реалізувати методи: знаходження кількості колекцій, що містять вказаний елемент; – знаходження максимальної колекції, що містить вказаний елемент.

6.	Створити масив об'єктів CollectionType. Реалізувати методи: знаходження кількості колекцій, що містять задане значення (вибрати будь-яке поле об'єкта); – знаходження максимальної та мінімальної колекції в масиві.
7.	Створити масив об'єктів CollectionType. Реалізувати методи: – знаходження кількості колекцій, сума яких більша за вказане значення (для підсумовування вибрати будь-яке поле об'єкта);

	– знаходження максимальної та мінімальної колекції в масиві.
8.	Створити масив об'єктів CollectionType. Реалізувати методи: знаходження кількості колекцій, що містять вказаний елемент; – знаходження мінімальної колекції, що містить вказаний елемент.
9.	Створити масив об'єктів CollectionType. Реалізувати методи: знаходження кількості колекцій, що містять лише 5 елементів; знаходження максимальної та мінімальної колекції в масиві за заданим значенням поля об'єкта (можна вибрати будь-яке поле).
10.	Створити масив об'єктів CollectionType. Реалізувати методи: знаходження кількості колекцій, що містять вказаний елемент; – знаходження максимальної та мінімальної колекції в масиві.

ЛАБОРАТОРНА РОБОТА №6 РОБОТА З ФАЙЛАМИ ТА СЕРІАЛІЗАЦІЯ.

Мета роботи: Вивчити можливості роботи з файлами, що надаються мовою C#. Освоїти практично переваги використання серіалізації. **Час проведення**: 4 години. **Місце проведення:** комп'ютерний клас.

Навчальні питання:

1. Навчитися роботі з файлами.

Література:

Основна література.

16.Коноваленко І.В. Програмування мовою C# 6.0. Тернопіль, ТНТУ, 2016. 227 с.

17.H. Schildt, “C# 4.0 The Complete Reference McGraw Hill Education; 1st edition” – 984 p., 2017.

18.J. Sharp, “Microsoft Visual C# Step by Step”, Pearson Education – 878p., 2018.

Допоміжна література.

1. A. Troelsen, “Pro C# 7 With .NET and .NET Core” - Apress, — 1372 p., 2017.

2. J. Albahari, B. Albahari, “C# 7.0 in a Nutshell”, O'Reilly Media— 1090 p., 2017.

3. Інструментальні програмні засоби розробки ІУС. Методичні вказівки до виконання лабораторних робіт. / уклад.: К.І. Київська–Київ: КНУБА, 2018. – 40 с

Інформаційні ресурси в Інтернеті

1. C# Programming. Yellow Book [Ел. ресурс]. URL: https://www.robmiles.com/c-yellow-book
2. Secure Coding Guidelines for .NET [Ел. ресурс]. URL: https://docs.microsoft.com/en-us/dotnet/standard/security/secure-coding-guidelines
3. C# Reference, сайт розробників MSDN. [Електронний ресурс] – Режим доступу: https://docs.microsoft.com/uk-ua/dotnet/csharp/languagereference/index
4. Об’єктно-орієнтоване програмування. Частина 1. Основи об’єктно-орієнтованого програмування на мові C#.: Навчальний посібник. / Д.В. Настенко, А. Б. Нестерко. – К.: НТУУ «КПІ», 2016. - 76с. [Електронний ресурс]. – Режим доступу https://ela.kpi.ua/bitstream/123456789/16671/1/OOP_manual.pdf

План проведення заняття:

Завдання 1

Вивчити роботу з файлами за матеріалами лекцій та підручника (глава 20, Троєлсен Е. Мова програмування C# 5.0 та платформа .NET 4.5. 6-е вид.).

Завдання 2

Проаналізувати інформацію, яку можна отримати за допомогою API (див. Додаток А).

Реалізувати тип даних (клас), що відповідає сутностям (об'єктам реального світу), про які надається інформація через API. Клас повинен включати поля/властивості для зберігання всієї інформації, що надається.

Розробити консольну програму, що дозволяє десеріалізувати інформацію в колекцію об'єктів створеного типу даних.

Реалізувати такі функції:

 додавання нового запису до колекції;

 видалення запису з колекції;

 перегляд усіх записів;

 зміна будь-якого запису;

 пошук записів по кожному з полів;

 експорт будь-якого запису до текстового файлу.

Програма повинна зберігати всі введені/змінені дані при виході з програми шляхом вибору відповідного пункту меню

"Вихід". При запуску програма повинна перевіряти наявність файлу з даними та завантажувати їх автоматично, якщо файл, створений у попередньому сеансі роботи, доступний та коректний. Збереження даних має бути реалізоване з використанням механізму серіалізації, завантаження – з використанням механізму десеріалізації.

Також у програмі необхідно використовувати обробку можливих винятків (наприклад, файл з даними відсутній, пошкоджений, введені некоректні дані користувачем тощо). Кожен елемент класу повинен містити документуючий коментар (///), який описує його призначення.

Вимоги до звіту

Звіт з цієї лабораторної роботи повинен включати:

1. титульний лист із зазначенням номера та назви лабораторної роботи;

2. мета роботи;

3. діаграму класів;

4. вихідні тексти розроблених програм;

5. результати роботи розроблених програм (скриншоти або тексти, що виводяться програмами);

6. тестові набори (обов'язково включити тестові набори, які призводять до появи винятків);

7. висновки, у яких необхідно відобразити призначення всіх розроблених класів.

Контрольні питання

1. 1. Які класи .NET Framework призначені для роботи з файлами?
2. 2. Що таке серіалізація?
3. 3. Що таке десеріалізація?
4. 4. Що таке маршаллінг/демаршаллінг?
5. 5. Які види серіалізації існують? У чому полягають їхні переваги/недоліки?
6. 6. Що таке виняток?
7. 7. Навіщо призначена обробка винятків?
8. 8. Як можна обробити винятки всіх типів?
9. 9. Як можна обробити лише певний тип винятків?

Додаток А. Варіанти завдань

№	*Дані*	*Формат*	*Посилання*
1.	Курс валют НБУ на поточну дату	JSON	https://bank.gov.ua/NBUStatService/v1/statdirectory/exchange?json
2.	Курс валют НБУ на поточну дату	XML	https://bank.gov.ua/NBUStatService/v1/statdirectory/exchange?xml
3.	Прогноз погоди по Харкову	XML	http://www.yr.no/place/Ukraine/Kharkiv/Kharkiv/forecast_hour_by_hour.xml
4.	Курс валют ЕЦБ на поточну дату	XML	https://api.fixer.io/latest
5.	Курс валют НБРБ на поточну дату	JSON	http://www.nbrb.by/API/ExRates/Currencies
6.	Факультети НУРЕ	JSON	http://cist.nure.ua/ias/app/tt/get_faculties
7.	Групи факультету комп'ютерної інженерії та управління НУРЕ	JSON	http://cist.nure.ua/ias/app/tt/get_groups?faculty_id=114

8.	Оператори послуг платіжної інфраструктури	JSON	http://data.gov.ua/sites/default/files/media//320/26.04.2017/13_Operatory_platignoj_infrastruktury_json.json
9.	Інформація про власників банків України	JSON	http://data.gov.ua/file/135694/download?token=4CptNdVC
10.	Український індекс міжбанківських ставок	JSON	https://bank.gov.ua/NBUStatService/v1/statdirectory/exchange?json
11.	Український індекс міжбанківських ставок	XML	https://bank.gov.ua/NBUStatService/v1/statdirectory/exchange
12.	Небанківські фінансові установи, яким видано ліцензії	JSON	http://data.gov.ua/sites/default/files/media//320/26.07.2017/08_Informacija_pro_generalni_licenzij_NFU_LIC_in_json.json
13.	Список виступаючих на конференції	JSON	https://gist.github.com/planetoftheweb/2c2f3b03b72a7f2ae923
14.	Підрозділи поліції	JSON	https://data.police.uk/api/forces
15.	Інформація про людину	JSON	https://randomuser.me/api
16.	Час сходу та заходу сонця	JSON	https://api.sunrise-sunset.org/json?lat=50.0437003&lng=36.2176341
17.	Інформація про країни	JSON	https://country.register.gov.uk/records.json
18.	Курси валют Національного банку Чехії	XML	https://www.cnb.cz/cs/financni_trhy/devizovy_trh/kurzy_devizoveho_trhu/denni_kurz.xml
19.	Жарти	JSON	https://08ad1pao69.execute-api.us-east-1.amazonaws.com/dev/random_ten
20.	Університети світу	JSON	https://raw.githubusercontent.com/Hipo/university-domains-list/master/world_universities_and_domains.json

ЛАБОРАТОРНА РОБОТА №7 РЕГУЛЯРНІ ВИРАЗИ. ЮНІТ-ТЕСТУВАННЯ.

Мета роботи: Вивчити можливості роботи з рядками, що надаються мовою C#. Вивчити обробку параметрів командного рядка. Освоїти використання класів StreamReader та StreamWriter. Навчитися працювати з регулярними виразами у мові C#. Вивчити використання юніт-тестів для перевірки

розробленої логіки..

Час проведення: 4 години. **Місце проведення:** комп'ютерний клас.

Навчальні питання:

1. Навчитися роботі з регулярними виразами
2. Навчитися проводити юніт-тестування.

Література:

Основна література.

19.Коноваленко І.В. Програмування мовою C# 6.0. Тернопіль, ТНТУ, 2016. 227 с.

20.H. Schildt, "C# 4.0 The Complete Reference McGraw Hill Education; 1st edition" – 984 p., 2017.

21.J. Sharp, "Microsoft Visual C# Step by Step", Pearson Education – 878p., 2018.

Допоміжна література.

1. A. Troelsen, "Pro C# 7 With .NET and .NET Core" - Apress, — 1372 p., 2017.
2. J. Albahari, B. Albahari, "C# 7.0 in a Nutshell", O'Reilly Media— 1090 p., 2017.
3. Інструментальні програмні засоби розробки ІУС. Методичні вказівки до виконання лабораторних робіт. / уклад.: К.І. Київська–Київ: КНУБА, 2018. – 40 с

Інформаційні ресурси в Інтернеті

1. C# Programming. Yellow Book [Ел. ресурс]. URL: https://www.robmiles.com/c-yellow-book
2. Secure Coding Guidelines for .NET [Ел. ресурс]. URL: https://docs.microsoft.com/en-us/dotnet/standard/security/secure-coding-guidelines
3. C# Reference, сайт розробників MSDN. [Електронний ресурс] – Режим доступу: https://docs.microsoft.com/uk-ua/dotnet/csharp/languagereference/index
4. Об'єктно-орієнтоване програмування. Частина 1. Основи об'єктно-орієнтованого програмування на мові C#.: Навчальний посібник. / Д.В. Настенко, А. Б. Нестерко. – К.: НТУУ «КПІ», 2016. - 76с. [Електронний ресурс]. – Режим доступу https://ela.kpi.ua/bitstream/123456789/16671/1/OOP_manual.pdf

План проведення заняття:

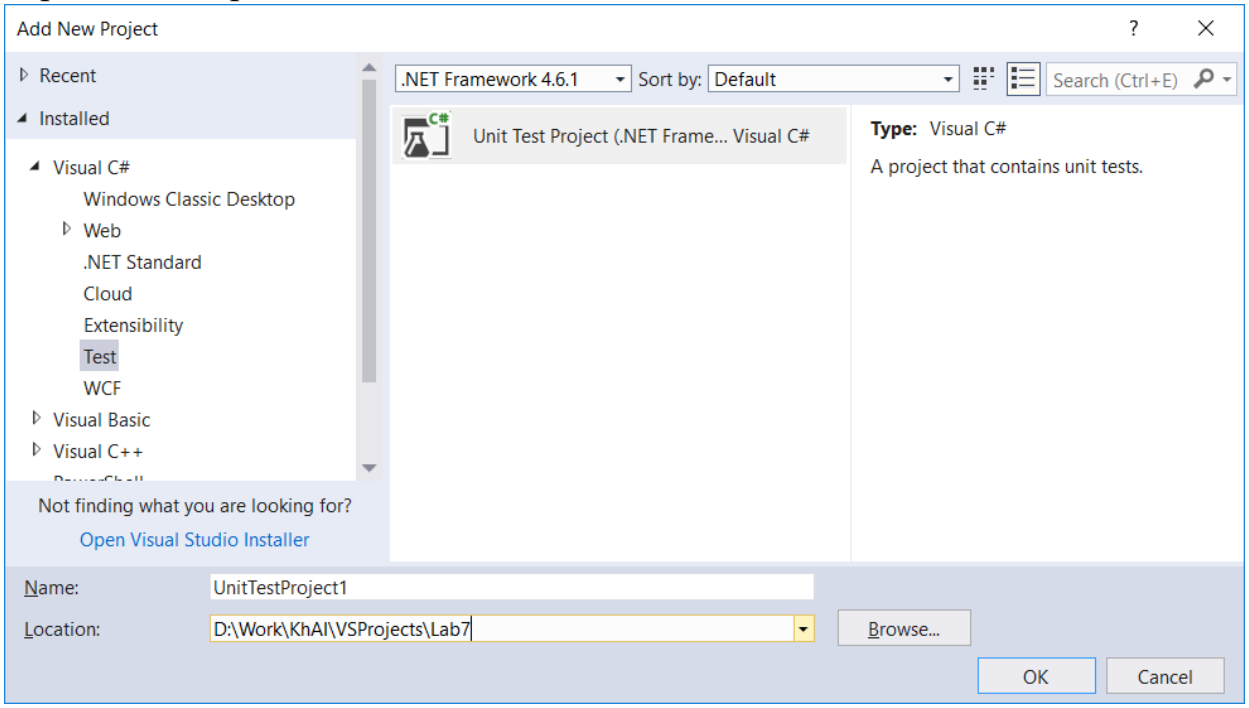
Завдання 1

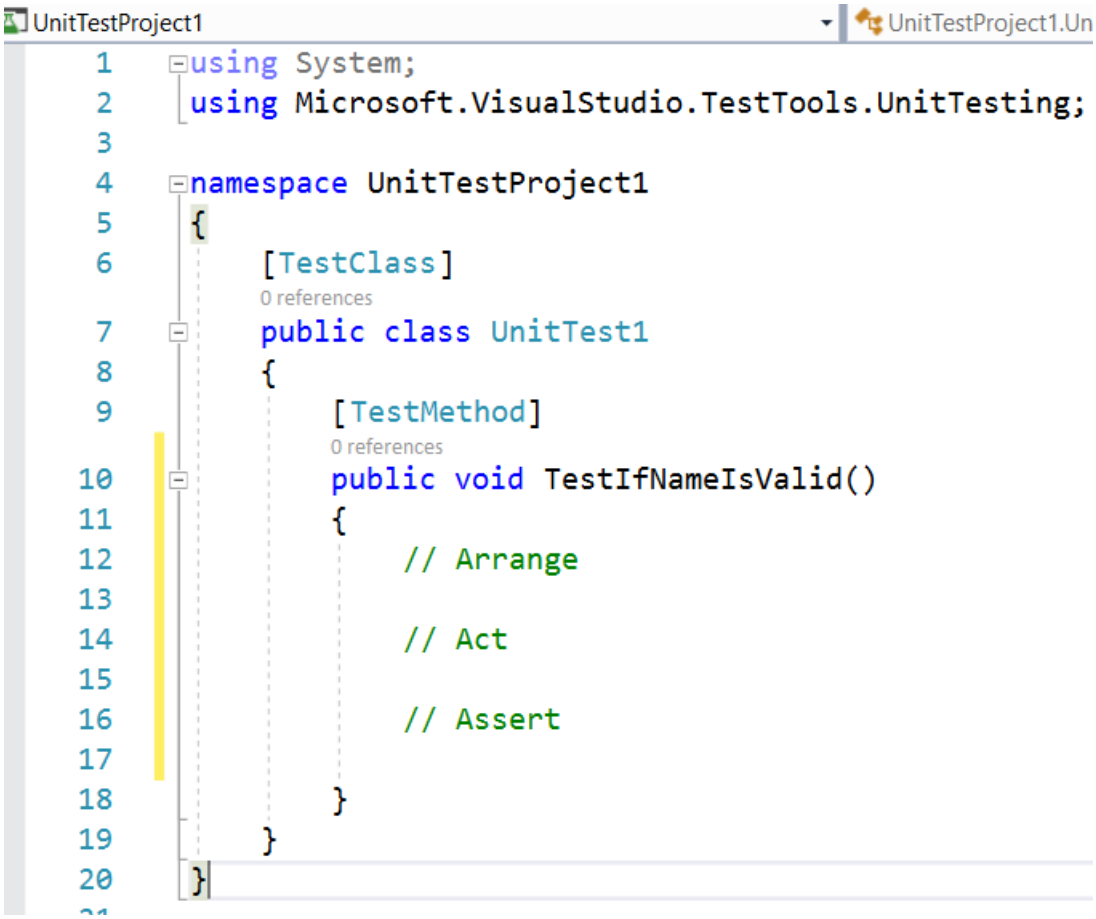
Розробити консольну програму, призначену для перевірки рядків на відповідність заданому шаблону. Результатом роботи програми має бути виведення інформації про те, які рядки відповідають заданому шаблону, а які – ні.
Якщо користувач вказав ключ /s, програма повинна перевірити всі слова, передані після ключа в командному рядку.
Якщо користувач вказав ключ /f та ім'я файлу, програма повинна перевірити всі слова у вказаному файлі.
Якщо користувач вказав ключ /? або /h, програма повинна вивести довідкову інформацію про її призначення та опис усіх ключів.
Якщо користувач запустив програму без ключів, вона повинна вимагати рядки у користувача та виводити результати перевірки; введення рядків завершується, якщо користувач ввів рядок "end".
Шаблон повинен бути поставлений за допомогою регулярного виразу.
Варіанти індивідуальних завдань наведено у додатку А.

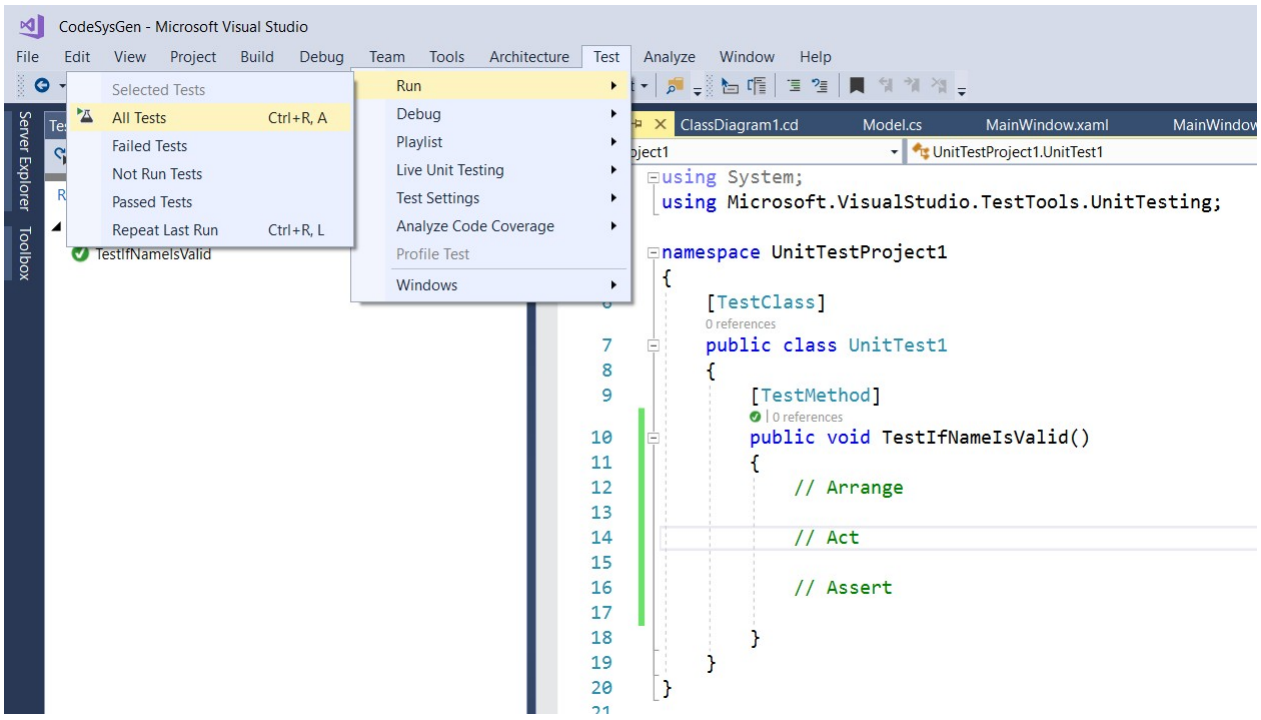
Завдання 2

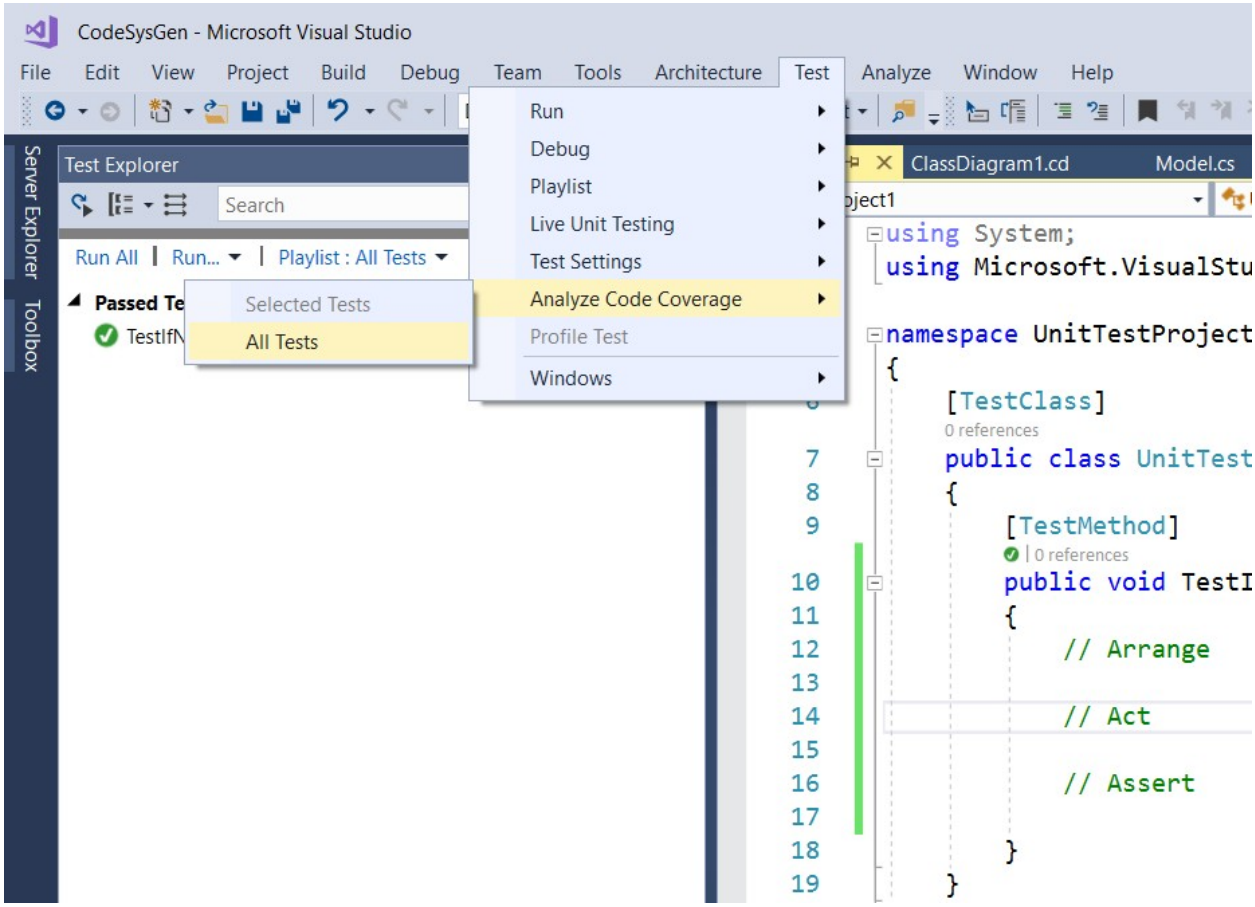
Розробити unit-тести для логіки, розробленої у завданні 1. Забезпечити рівень покриття тестами щонайменше 50%.

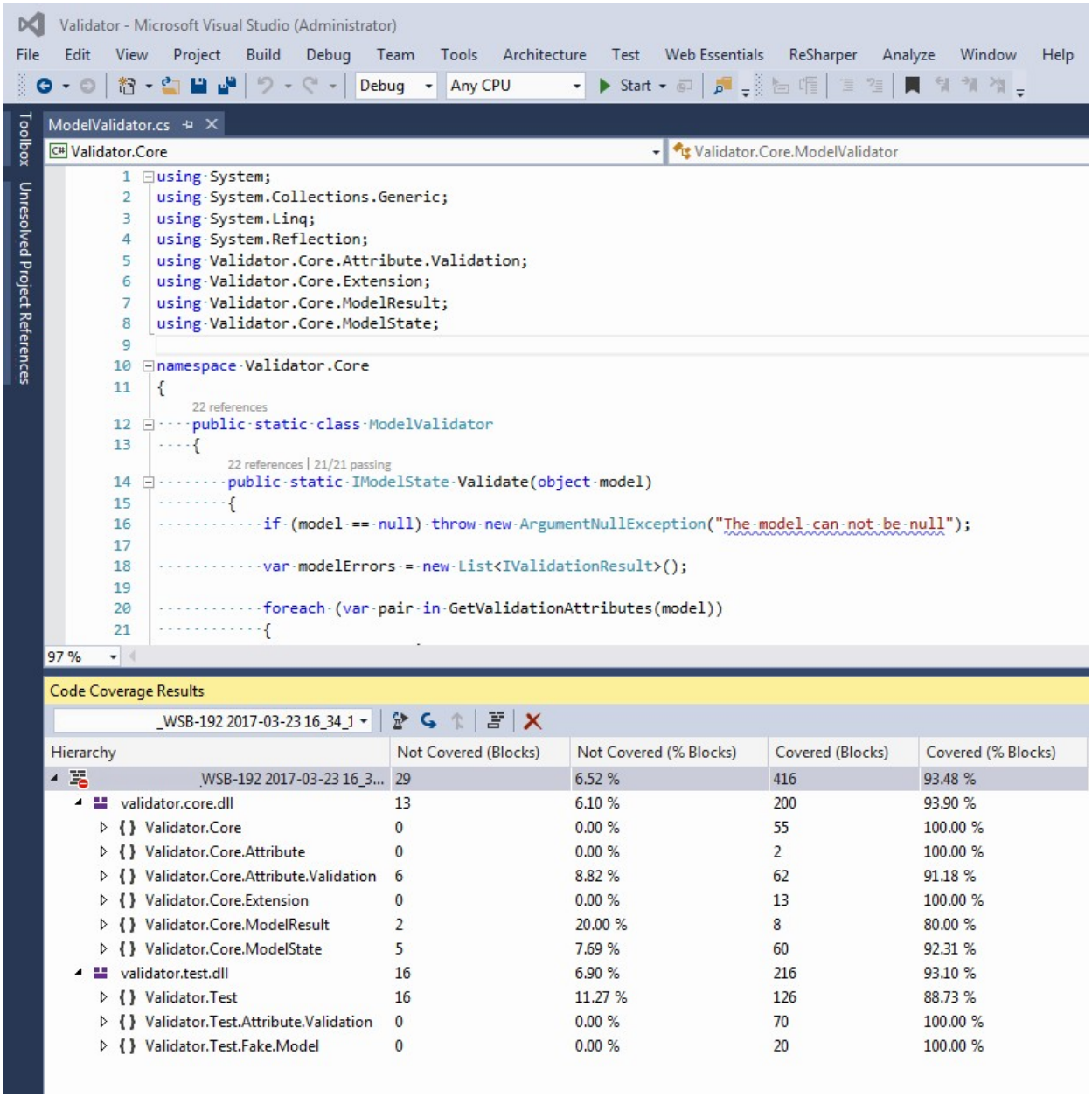
Приклад створення unit-тестів:




UnitTestProject1
UnitTestProject1.Un
using System;
using Microsoft.VisualStudio.TestTools.UnitTesting;
namespace UnitTestProject1
{
[TestClass]
0 references
public class UnitTest1
{
[TestMethod]
0 references
public void TestIfNameIsValid()
{
// Arrange
// Act
// Assert
}
}
}


CodeSysGen - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Architecture Test Analyze Window Help
Selected Tests
All Tests Ctrl+R, A
Failed Tests
Not Run Tests
Passed Tests
Repeat Last Run Ctrl+R, L
Run
Debug
Playlist
Live Unit Testing
Test Settings
Analyze Code Coverage
Profile Test
Windows
TestIfNameIsValid
Server Explorer
Toolbox
ClassDiagram1.cd
Model.cs
MainWindow.xaml
UnitTestProject1.UnitTest1
using System;
using Microsoft.VisualStudio.TestTools.UnitTesting;
namespace UnitTestProject1
{
[TestClass]
0 references
public class UnitTest1
{
[TestMethod]
0 references
public void TestIfNameIsValid()
{
// Arrange
// Act
// Assert
}
}
}


CodeSysGen - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Architecture Test Analyze Window Help
Run
Debug
Playlist
Live Unit Testing
Test Settings
Analyze Code Coverage
Profile Test
Windows
Selected Tests
All Tests
Test Explorer
Search
Run All | Run... | Playlist : All Tests
Server Explorer
Toolbox
ClassDiagram1.cd
Model.cs
using System;
using Microsoft.VisualStu
namespace UnitTestProject
{
[TestClass]
0 references
public class UnitTest
{
[TestMethod]
0 references
public void TestI
{
// Arrange
// Act
// Assert
}
}



Рекомендовані посилання:
http://vscode.ua/prog-lessons/modulnoe-testirovanie-v-visual-studio.html
http://streletzcoder.ua/osnovyi-modulnogo-unit-testirovaniya-v-visual-studio/

http://vscode.ru/prog-lessons/modulnoe-testirovanie-v-visual-studio.html
http://streletzcoder.ru/osnovyi-modulnogo-unit-testirovaniya-v-visual-studio/

Вимоги до звіту

Звіт з цієї лабораторної роботи повинен включати:

1. титульний лист із зазначенням номера та назви лабораторної роботи;
2. мета роботи;
3. діаграму класів;

4. вихідні тексти розроблених програм;
5. результати unit-тестування;
6. висновки з описом можливостей класів String, StreamReader і StreamWriter, і навіть Group, Capture, Match, RegEx простору імен System.Text.RegularExpressions.

Контрольні запитання

1. Як отримати доступ до параметрів командного рядка в консольній програмі?
2. Навіщо призначений клас StreamReader?
3. Навіщо призначений клас StreamWriter?
4. Які конструктори мають клас String?
5. Які методи є у класу String?
6. Які виняткові ситуації можливі під час роботи з файлами?
7. Які існують режими відкриття файлів?
8. Що таке регулярні вирази? Навіщо вони призначені?
9. Для чого призначено клас Group?
10. Для чого призначений клас Capture?
11. Для чого призначений клас Match?
12. Для чого призначений клас RegEx?
13. Що таке unit-тестування?

Додаток А. Варіанти завдань для завдання 1

Вариант	***Задание***
1.	Рядки, що містять як слово ціле число. Приклад рядків, які підходять: Year is 2018., 1 is a number, «3.1415 matches because. is not a word char», "Lab 7 is very simple". Приклад рядків, які не підходять: Not2Bad, No digits here, 2Good4You.
2.	Рядки, що містять коректні ідентифікаційні номери транспортних засобів (VIN-коди). Приклади рядків, які підходять: WP1ZZZ92ZBLA03352, WBSGZ01010L587108, "Y6DTF698KE0332004", "XTE110270R0282981". Приклади рядків, які не підходять: «VINVIN123456», "12345678987654321".

3.	Рядки, що містять «cat» як підрядок двічі. Приклад рядків, які підходять: catcat, cat and cat. Приклад рядків, які не підходять: "catac", "cat", "Caatt".
4.	Рядки, що містять коректні номерні знаки транспортних засобів України типу 1 згідно з ДСТУ 4278. Приклади рядків, що підходять: «AX0200BA», «KX7777PP», «BI5419EC». Приклади рядків, які не підходять: "AXAXAX", "ZZ0901ZZ", "EC1234AA".
5.	Рядки, що містять коректні номери телефонів мобільних операторів України. Приклади рядків, які підходять: "(050) 310-20-89", "0971125678", "073-123-45-76". Приклади рядків, які не підходять: «0112345678», «(077) 212-45-08», «09548567 нуль».
6.	Рядки, що містять двійковий запис числа, кратного 3. Приклад рядків, які підходять: "0", "10010". Приклад рядків, які не підходять: "00101", "Not a number", "1 1", "0 0".
7.	Рядки, що є смайликами: □ першим символом є або; або: один раз; □ далі може йти символ – скільки завгодно раз (у тому числі нуль разів); □ в кінці обов'язково йде деяка кількість (не менше однієї) однакових дужок з наступного набору: (,), [,]; □ усередині смайлика не може зустрічатися жодних інших символів. Приклад рядків, які підходять: ":)", "; [[[[[]]. Приклад рядків, які не підходять: ":-)]", ";--".
8.	Рядки, що містять слово всередині довільного тексту, що не містить дужок, у дужках. Приклад рядків, які підходять: "good (excellent) phrase", "good (too bad) phrase", "good ((recursive)) phrase". Приклад рядків, які не підходять: "word () is not () in brackets", "bad (() recursive) phrase", "no brackets here".
9.	Рядки, що містять «cat» як слово. Приклад рядків, які підходять: "cat", "catapult and cat", "catapult and cat and concatenate". Приклад рядків, які не підходять: catcat, concat, Cat.

10.	Рядки, що не містять провідних або кінцевих символів. Приклад рядків, які підходять: "Good string", "". Приклад рядків, які не підходять: "bad string", "bad string", "very bad string".
11.	Рядки, що містять правильні MAC-адреси. Приклад рядків, які підходять: "01:32:54:67:89:AB", «aE:dC:cA:56:76:54». Приклад рядків, які не підходять: "01:33:47:65:89:ab:cd", "01:23:45:67:89:Az".
12.	Рядки, що відповідають шаблону «YYYY/MM/DD» та містять коректну дату в діапазоні від 1990 до 2018 року. Приклад рядків, які підходять: «2012/05/02», "1990/11/05". Приклади рядків, які не підходять: "2019/03/02", "2018/25/25".
13.	Рядки, що містять коректні IPv4 адреси в десятковому поданні з точками. Приклади рядків, які підходять: "195.88.72.125", "127.0.0.1". Приклад рядка, який не підходить: "256.256.256.256".
14.	Рядки, що містять доменні імена для протоколів http і https, з необов'язковим слешем наприкінці. Приклади рядків, які підходять: http://khai.edu, https://amazon.com/. Приклади рядків, які не підходять: «univd.edu.ua», "http:// univd.edu.ua".
15.	Рядки, що містять правильні hex ідентифікатори кольору HTML. Приклади рядків, які підходять: #FFFFFF, "#FF3421", "#00ff00". Приклади рядків, які не підходять: "232323", "f#fddee", "#fd2".
16.	Рядки, що містять коректні номерні знаки транспортних засобів України стандарту 1995 року. Приклади рядків, які підходять: "777-77ХА", "12345ХР", "229-06ТС". Приклади рядків, які не підходять: "33-222ХК", "321-04ХХ", "89012МІ".
17.	Рядки, що містять коректні номери кредитних карток Visa, Mastercard або Простір. Приклади рядків, які підходять: "4012888888881881", "5105 1051 0510 5100", "9804 1286 0012 7543". Приклади рядків, які не підходять: «1234567812345678», «0000 0000 1111 2222», "401278684828183".

18.	Рядки, що містять час у форматі "ЧЧ:ММ:СС". Приклади рядків, які підходять: 12:20:45, 23:00:54. Приклади рядків, які не підходять: 12:00, 14:11:67.
19.	Рядки, які містять коректні номери внутрішнього паспорта громадянина України, паспорти громадянина України для виїзду за кордон або коректний реєстраційний номер облікової картки платника податків. Приклади рядків, які підходять: "ММ450190", "FE126018", "000111222", "2005820864". Приклади рядків, які не підходять: "123456ЕЕ", "235768".
20.	Рядки, що містять коректні номери студентських квитків. Приклади рядків, що підходять: «ВК№10330615», "ХА № 10450718", "ХА 10320416". Приклади рядків, які не підходять: "ХАІ12345678", "1234567890".