**МЕТОДИЧНІ ВКАЗІВКИ ДЛЯ ЛАБОРАТОРНИХ РОБІТ**

**З ОБ'ЄКТНО-ОРІЄНТОВАНОГО ПРОГРАМУВАННЯ**



**ВСТУП**

В посібнику наведено 19 лабораторних робіт з дисципліни «Об'єктно-орієнтоване програмування». Кожна робота містить короткі теоретичні відомості, приклад виконання роботи та індивідуальні завдання. В якості мови програмування використовується C #.

Об'єктно-орієнтоване програмування - це методологія програмування, заснована на представленні програми у вигляді сукупності об'єктів, кожен з яких є екземпляром певного класу, а класи утворюють ієрархію спадкування.

Клас - це множина об'єктів, що володіють загальною структурою, поведінкою і семантикою. Окремий об'єкт - це екземпляр класу. Клас представляє лише абстракцію істотних властивостей об'єкта.

Стан об'єкта характеризується переліком (зазвичай статичним) всіх властивостей даного об'єкта і поточними (зазвичай динамічними) знаннями кожної з цих властивостей.

Поведінка об'єкта - це те, як об'єкт діє і реагує; поведінка виражається в термінах стану об'єкта і передачі повідомлень.

Об'єктно-орієнтоване програмування будується на чотирьох принципах: абстракція, інкапсуляція, поліморфізм і успадкування.

**Лабораторна робота №1**

**Інтегроване середовище розробки Visual Studio**

**Мета роботи:**

*1. Ознайомлення з середовищем розробки Visual Studio 2017, основними типами проектів*

*2. Створення простого консольного проекту. Консольне введення-виведення. Клас Console і його методи*

*3. Дослідження структури консольного проекту*

*4. Способи запуску проектів на виконання*

*5. Створення простого Windows проекту*

**6. Варіанти завдань для Лабораторної роботи № 1**

Лабораторні роботи виконуються в середовищі Visual Studio 2017 (будь-якої редакції).

**Методичні рекомендації до виконання лабораторної роботи**

1. Прочитайте цю лабораторну роботу та виконайте наведені в ній приклади (вони всі працездатні)

3. Зверніть увагу на методи призначені для введення:

*Console.ReadKey();* – обробки натиснення клавіатури

*string str = Console.ReadLine()* – введення рядка.

*Console.Read()* – введення одного символу.

Методи призначені для виведення:

*Console.WriteLine(“Hello, World!”);*

*Console.Write(“Hello, World!”);*

Метод Console.**WriteLine** відрізняється від методу Console.Write тим, що завершує свою роботу обов'язковим виведенням Escape-послідовності line feed/carriage return (перехід на новий рядок).

**Порядок виконання роботи**

1. Створити директорію Lab1, в якій буде розміщуватися проект цієї лабораторної роботи.

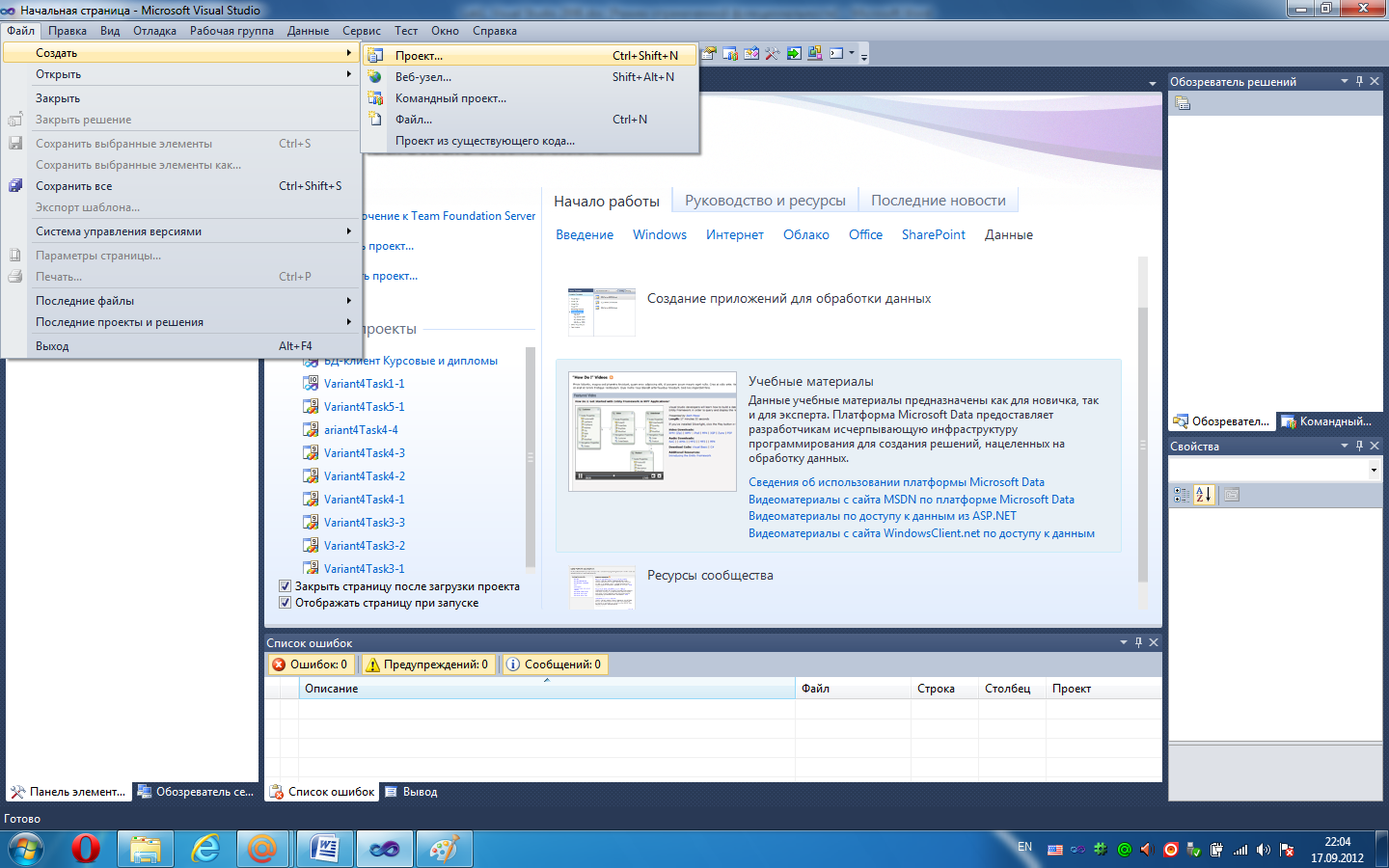
2. Виконати завдання свого варіанту у вигляді одного проекту

3. Зафіксувати результати у Лабораторній роботі 1.

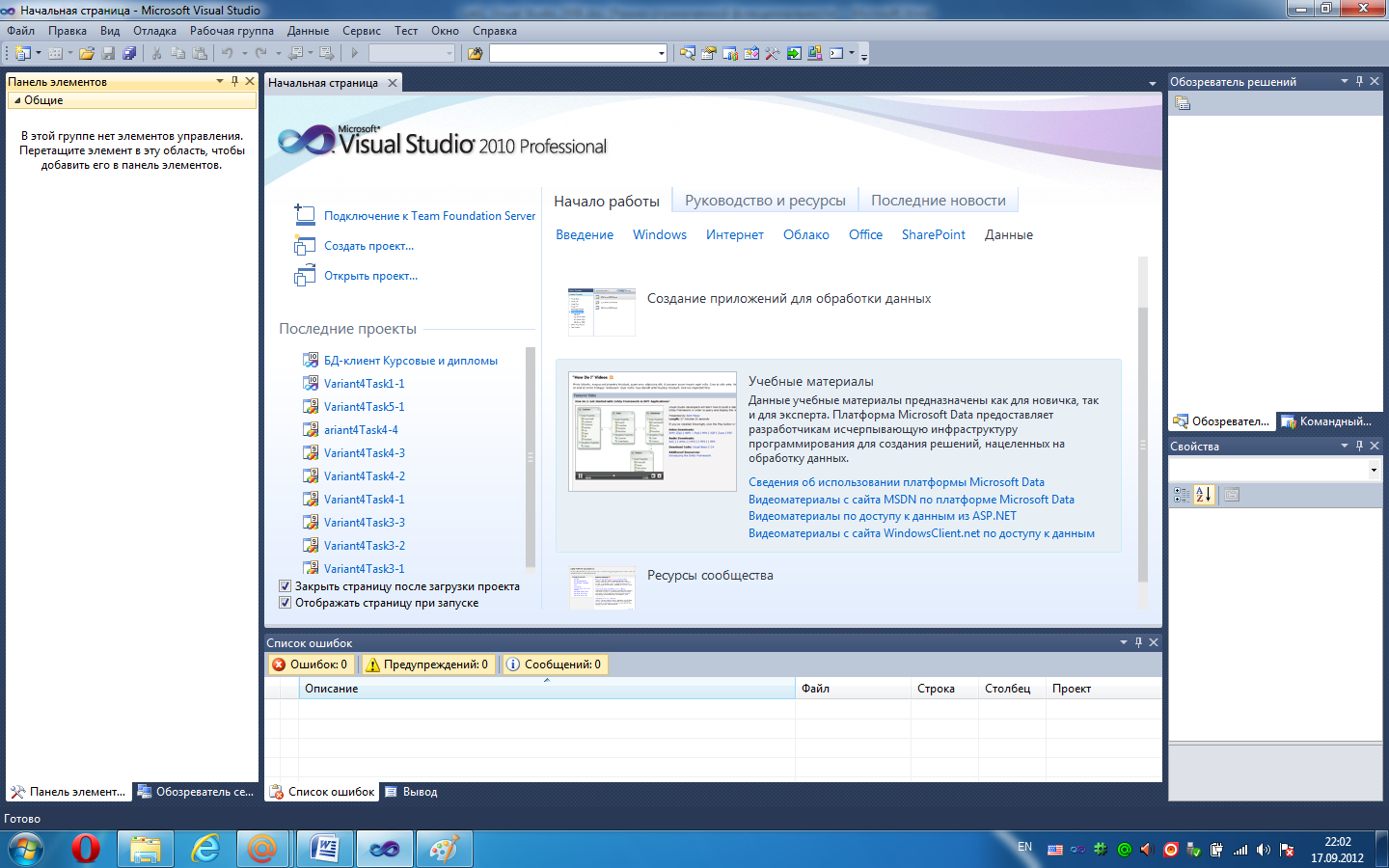
**1. Ознайомлення з середовищем розробки Visual Studio 2017, основними типами проектів**

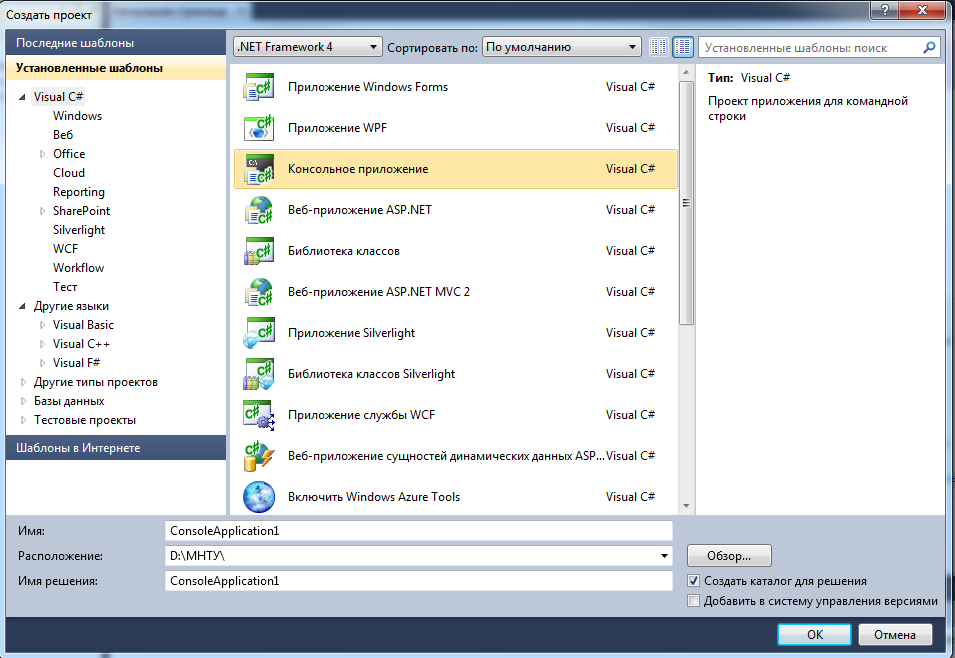
Після запуску Visual Studio (VS) відображається стартова сторінка.

Для створення нового проекту потрібно в меню Файл вибрати команду Создать, потім команду Проект.



Відкриється вікно вибору виду проекту. Visual Studio .Net для мов C#, Visual Basic і C++ пропонує різні види проектів.





**Рис. 1.1. Вікно вибору шаблону проекту VS 2010**

Для С# Visual Studio 2017 пропонуються різні типи проектів, зокрема:

**Приложение Windows Form (Windows Form Application)** – створення Windows-застосунків з інтерфейсом користувача (локальних і розподілених).

**ASP.NET Web Application** - створення динамічних веб-сайтів.

**Приложение WPF (Windows Presentation Foundation***) –* створення презентацій: складних графічних інтерфейсів з можливостями анімації, 2D і 3D графіки. Використовується векторна графіка для 2D, тому інтерфейси будуть максимально незалежні від роздільної здатності екрану і розміру вікна.

**Console Application** – створення консольних програм (без інтерфейсу користувача)

**Библиотека классов (ClassLibrary) -** створення власної DLL

**Windows Form Control Library –** створення бібліотек власних Windowsкомпонентів (елементів керування) та інші типи проектів.

**2. Створення простого консольного проекту. Консольне введення-виведення. Клас Console і його методи**

В файловій системі потрібно створити папку з шифром групи. В цій папці будуть створюватися ваші проекти. Створимо в ній папку Lab1, де будуть зберігатися проекти цієї лабораторної роботи.

Кожну роботу будемо розміщувати в окремій папці.

Створимо перший проект – консольний застосунок, який виводить на консоль повідомлення “Hello, world!”.

1. Виберемо пункт меню File|New|Project, задамо тип проекту ConsoleApplication.

2. У полі назви проекту введемо ConsoleHello.

3. У полі Location вказується шлях до каталогу, де зберігатиметься Рішення, що містить проект. Розмістіть рішення в папці Lab1.

4. Буде створено шаблон проекту.

**3. Дослідження структури консольного проекту**

У великому полі в центрі, яке по суті являє текстовий редактор, знаходиться згенерований за замовчуванням код C #. Згодом ми змінимо його на свій.

Справа знаходиться вікно *Solution Explorer*, в якому можна побачити структуру нашого проекту. В даному випадку у нас згенерувана за замовчуванням структура: вузол Properties або Властивостей (він зберігає файли властивостей додатки і поки нам не потрібен); вузол References - це вузол містить збірки dll, які додані в проект за замовчуванням. Ці збірки якраз містять класи бібліотеки .NET, які буде використовувати C #. Однак не завжди все збірки потрібні. Непотрібні потім можна видалити, в той же час якщо знадобиться додати якусь потрібну бібліотеку, то саме в цей вузол вона буде додаватися.

Далі йде файл конфігурації App.config (поки він нас не цікавить) і безпосередньо сам файл коду програми Program.cs. Якраз цей файл і відкритий в центральному вікні. Спочатку розберемо, що весь цей код представляє:

/\*начало секции подключаемых пространств имен\*/

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

/\*конец секции подключаемых пространств имен\*/

namespace HelloApp /\*объявление нового пространства имен\*/

{

    class Program  /\*объявление нового класса\*/

    {

        static void Main(string[] args) /\*объявление нового метода\*/

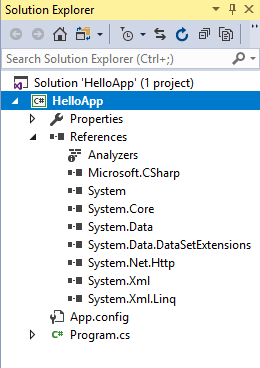
        {

        }/\* конец объявления нового метода\*/

    } /\* конец объявления нового класса\*/

} /\* конец объявления нового пространства имен\*/

На початку файлу йдуть директиви using після яких йдуть назви просторів імен, що підключаються (класи) . Простори назв є організацію класів в загальні блоки. Наприклад, на першому рядку using System; підключається простір імен System, який містить фундаментальні і базові класи платформи .NET. Фізично простори імен (класи) знаходяться бібліотеках dll, що підключаються, які можна побачити у вікні Solution Explorer, відкривши вузол References:



Кваліфікатором доступу **є крапка**, наприклад, Console.WriteLine. Для затримки консолі вставимо виклик методу Console.ReadKey() або Console.ReadLine();

Програма готова для компіляції і запуску!

Для побудови збірки потрібно вибрати в меню **Build** команду **Build** **Solution**.

## Консольне введення-виведення

Для організації консольного введення і виведення використовуються статичні методи класу System.Console.

Console.WriteLine(“Hello, World!”);

Console.Write(“Hello, World!”);

**WriteLine** відрізняється від Write тим, що завершує свою роботу обов'язковим виведенням Escape-послідовності line feed/carriage return.

**Засоби введення**

C# дозволяє здійснювати консольне введення як одного символу, так і цілого рядка.

Методи введення з консолі:

Console.ReadKey(); – очікує натиснення клавіатури

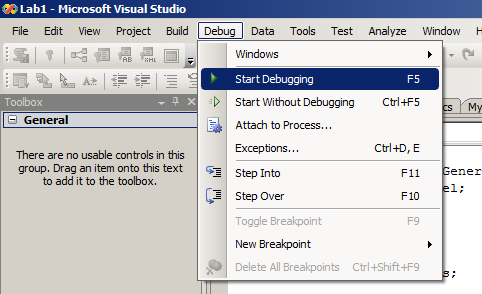
string str = Console.ReadLine() – введення рядка.

string str = Console.Read() – введення одного символа.

**4. Способи запуску проектів на виконання**

В середовищі Visual Studio є **два способи** запуску рішення на виконання:

**Перший спосіб** – режим відлагодження. Для цього слід вибрати меню **Debug|Start Debugging** або кнопку на панелі інструментів (зелений трикутник).



Другий спосіб – без відлагодження. Вибираємо **Start Without Debugging**.

**5. Створення простого Windows проекту**

Виберемо пункт меню File|New|Project, задамо тип проекту **Windows Forms Application,** дамо йому ім'я – WindowsHello. Збережемо його в папці Lab1.

Буде створена форма з іменем Form1 – головне вікно застосунку. Змінилося і середовище Visual Studio. З'явилися вікна **Панель Элементов** (Toolbox) та **Свойства** (Property).

У вікні **Панель Элементов** перелічені доступні елементи керування і компоненти, які можна розміщувати на формі. Вікно **Свойства** призначене для налаштування властивостей і подій об'єктів.

На формі створимо текстове поле **textBox** та кнопку **button**.

При натисненні на кнопку в поле **textBox** буде виводитися "HelloWorld!"

namespace WindowsHello

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

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

{

textBox1.Text = "Hello, world!";

}

}

}

Клас форми розділений на 2 файли. Один з файлів генерує дизайнер, в другому розміщується код обробників подій, який пише програміст. При збірці рішення вони об'єднуються в один файл.

Об’єкт **textBox** має властивість Text, значенням якої є текст, який буде виведений в поле.

**6. Варіанти завдань для Лабораторної роботи № 1**

**Номер варіанта відповідає номеру прізвища студента у списку групи**

1. Створити в C# консольний проект, що реалізує завдання згідно варіанту. Вхідні дані ввести з клавіатури. Результати вивести на консоль.
2. Створити в C# Windows форму, на якій розмістити текстове поле (об‘єкт **texBox**) і кнопку (об‘єкт **button**). При натисканні на кнопку в текстове поле вивести рядок тесту відповідно до номера варіанту.

Виконання цієї роботи може зайняти 10-15 хвилин вашого часу.

|  |  |
| --- | --- |
| **№ варіанту** | **Зміст завдання** |
| 1 | 1. Вивести на консоль повідомлення про себе "Я студент третього курсу – (своє прізвище). Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 2 | 1. Вивести на консоль повідомлення про себе "Я студент третього курсу – (своє прізвище). Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 3 | 1. Вивести на консоль повідомлення про себе "Я студент третього курсу – (своє прізвище). Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 4 | 1. Вивести на консоль повідомлення про себе "Я студент третього курсу – (своє прізвище). Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 5 | 1. Вивести на консоль повідомлення про себе "Я студент третього курсу – (своє прізвище). Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 6 | 1. Вивести на консоль повідомлення про себе "Я студент третього курсу – (своє прізвище). Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 7 | 1. Вивести на консоль повідомлення про себе "Я студент третього курсу – (своє прізвище). Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 8 | 1. Вивести на консоль повідомлення про себе "Я студент третього курсу – (своє прізвище). Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 9 | 1. Вивести на консоль повідомлення про себе "Я студент третього курсу – (своє прізвище). Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 10 | 1. Вивести на консоль повідомлення про себе "Я студент третього курсу – (своє прізвище). Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 11 | 1. Вивести на консоль повідомлення про себе "Я студент третього курсу – (своє прізвище). Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 12 | 1. Вивести на консоль повідомлення про себе "Я студент третього курсу – (своє прізвище). Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 13 | 1. Вивести на консоль повідомлення про себе "Я студент третього курсу – (своє прізвище). Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 14 | 1. Вивести на консоль повідомлення про себе "Я студент третього курсу – (своє прізвище). Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 15 | 1. Вивести на консоль повідомлення про себе "Я студент третього курсу – (своє прізвище). Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 16 | 1. Вивести на консоль повідомлення про себе "Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 17 | 1. Вивести на консоль повідомлення про себе "Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 18 | 1. Вивести на консоль повідомлення про себе "Я студент третього курсу – (своє прізвище). Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 19 | 1. Вивести на консоль повідомлення про себе "Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |
| 20 | 1. Вивести на консоль повідомлення про себе "Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це моя перша робота на С#"  2. Вивести в текстове поле Windows форми рядок **«Моє прізвище <своє прізвище>. Я працюю <місце роботи>. Це мій перший проект»** |

**Література**

1.C# 2005 и платформа .Net 3.0 для профессионалов. Нейгел К., Ивьен Б. и др. –М.: ООО “И.Д. Вильямс", 2008. –1376 с.

2. Д.М. Златопольский. Сборник задач по программированию. 2-е издание. // СПб.: БХВ-Петербург, 2007.-240 с.

**Лабораторна робота № 2**

**Мова С# і платформа MS.Net Framework**

**(max 5 балів)**

**Створення простих програм в середовищі Visual Studio 2017**

**Мета роботи:**

1. Прості обчислення з перетворенням типів даних. Метод Parse

2. Прості арифметичні вирази

3. Клас Convert

4. Перетворення з арифметичного типу в тип string. Метод ToString

**5. Варіанти завдань для Лабораторної роботи № 2.**

**Методичні рекомендації до виконання лабораторної роботи**

1. Прочитайте цю лабораторну роботу та виконайте наведені в ній приклади (вони всі працездатні)

2. Зверніть увагу на те, що при введенні чисел з консолі у змінні арифметичного типу потрібно застосовувати явне приведення типу, так як неявного перетворення з типу **string** до арифметичного немає.

3. Для поглибленого вивчення цього матеріалу прочитайте розділи 1,2 книги [1].

**Порядок виконання роботи**

1. Створити директорію Lab2, в якій буде розміщуватися проект цієї лабораторної роботи.

2. Виконати завдання свого варіанту у вигляді одного проекту

**Приклади виконання завдань**

**1. Прості обчислення з перетворенням типів даних. Метод Parse**

Розглянемо приклад створення простого консольного застосунку, який виконує прості арифметичні обчислення. Запустимо Visual Studio, виберемо тип проекту Console Application, дамо йому назву **Lab2\_1** і натиснемо кнопку **OK**.

Створимо програму, яка вводить два цілих числа, знаходить їх суму і виводить результат на консоль. Потім вводить два дійсні числа, обчислює їх частку і також виводить на консоль.

Ось таким може бути код цієї програми. Його можна просто скопіювати в VS і виконати.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Lab2\_1

{

class Program

{

static void Main(string[] args)

{

//Знаходження суми цілих чисел

Console.WriteLine("Введіть перше число");

int a1 = int.Parse(Console.ReadLine());

Console.WriteLine("Введіть друге число");

int a2 = int.Parse(Console.ReadLine());

Console.WriteLine("Сума чисел a1+a2= " + (a1 + a2));

//Знаходження частки дійсних чисел

Console.WriteLine("Введіть перше число");

double d1 = double.Parse(Console.ReadLine());

Console.WriteLine("Введіть друге число");

double d2 = double.Parse(Console.ReadLine());

Console.WriteLine("Частка чисел d1/d2= " + (d1/d2));

Console.ReadKey();

}

}

}

Дані, які вводяться з консолі мають тип string (це рядки тексту). Тому введені числа потрібно перетворити в арифметичний тип. В нашому прикладі виконується явне перетворення типів з типу string в типи **int** і **double** з використанням методу Parse і неявне - при виведенні результатів обчислень на консоль.

Розглянемо ще приклад явного перетворення типу.

**Приклад**

*string str = Console.ReadLine();*

*int i = int.Parse(str);*

*float f = float.Parse(str);*

*double d = double.Parse(str);*

**2. Прості арифметичні вирази**

Розглянемо дещо складніший приклад обчислення арифметичного виразу. Нехай нам потрібно обчислити значення поліному р=ax3+bx2 + cx+d для будь-якого дійсного x (вводиться з консолі). Значення a,b,c,d є цілими і ініціалізуються в коді. Текст програми може бути таким:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Lab2\_2

{

class Program

{

static void Main(string[] args)

{

//обчислити значення поліному р=ax^3+bx^2+cx+d

int a = 100;

int b = 20;

int c = 25;

Console.WriteLine("Введіть x");

double x = double.Parse(Console.ReadLine());

double p = a \* x \* x \* x + b \* x \* x + c \* x + 30;

Console.WriteLine("p = " + p);

Console.ReadKey();

}

}

}

**3. Клас Convert**

Клас Convert містить статичні методи, які також можна використовувати для перетворень типів. Розглянемо приклад використання цього класу.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Lab1\_2

{

class Program

{

/// <summary>

/// Тестування методів класу Convert

/// </summary>

static void Main(string[] args)

{

string s;

byte b;

int n;

double x;

bool flag;

char sym;

DateTime dt;

sym = '7';

s = Convert.ToString(sym);

x = Convert.ToDouble(s);

n = Convert.ToInt32(x);

b = Convert.ToByte(n);

flag = Convert.ToBoolean(b);

x = Convert.ToDouble(flag);

s = Convert.ToString(flag);

s = "300";

n = Convert.ToInt32(s);

s = "14.09";

s = "14.09.2008";

dt = Convert.ToDateTime(s);

}

}

}

**4. Перетворення з арифметичного типу в тип string. Метод ToString**

Так як метод ToString() визначений в базовому класі object, він має свої реалізації для всіх арифметичних типів даних. Далі наведено простий приклад, який демонструє застосування цього методу.

В першому варіанті використовується **явне** перетворення типу з арифметичного до типу string, а у другому - **неявне**.

Варіант 1.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Lab2\_3

{

class Program

{

/// <summary>

/// Демонстрація перетворення в рядок

/// даних різного типа.

/// </summary>

static void Main(string[] args)

{

string name;

int age;

double salary;

name = "Василь Іванов";

age = 20;

salary = 2700;

string s = "Ім'я: " + name +

". Вік: " + age.ToString() +

". Зарплата: " + salary.ToString();

Console.WriteLine(s);

Console.ReadKey();

}

}

}

**Варіант 2**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Lab2\_3

{

class Program

{

/// <summary>

/// Демонстрація перетворення в рядок

/// даних різного типа.

/// </summary>

static void Main(string[] args)

{

string name;

int age;

double salary;

name = "Василь Іванов";

age = 20;

salary = 2700;

string s = "Ім'я: " + name +

". Вік: " + age +

". Зарплата: " + salary;

Console.WriteLine(s);

Console.ReadKey();

}

}

}

**5. Варіанти завдань для Лабораторної роботи № 2.**

**Номер варіанту відповідає номеру прізвища студента у списку групи**

Написати в C# консольний застосунок, що реалізує завдання згідно варіанту. Вхідні дані ввести з клавіатури. Результати вивести на консоль. Всі завдання варіанта реалізувати **в одному проекті Console\_Lab2**.

Виконання цієї роботи може зайняти 30 хвилин вашого часу.

Кожний студент повинен виконати по 5 завдань.

|  |  |
| --- | --- |
| **№ варіанту** | **Зміст завдання** |
| 1 | 1. Реалізувати функцію обчислення суми та різниці двох цілих чисел  2. Реалізувати функцію піднесення в квадрат добутку двох цілих чисел  3. Знайти значення поліному p=a\*x^4-b\*x^3+c\*x+d. Значення a,b,c,d – дійсні числа, ініціалізувати в коді, x ввести з консолі. Результат вивести на консоль.  4. Обчислити середнє арифметичне і середнє геометричне двох цілих чисел. Результат вивести на консоль. Використати методи класу Convert.  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс. |
| 2 | 1. Реалізувати функцію піднесення до кубу частки двох цілих чисел  2. Реалізувати функцію обчислення добутку двох дійсних чисел  3. Знайти значення поліному p=a\*x^5-1/b\*x^4+c\*x+d. Значення a,b,c,d – цілі числа, ініціалізувати в коді, x ввести з консолі. Результат вивести на консоль. Використати методи класу Convert.  4. Обчислити периметр прямокутного трикутника за заданими сторонами.  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс. |
| 3 | 1. Реалізувати функцію обчислення добутку двох цілих чисел  2. Реалізувати функцію піднесення до кубу суми двох дійсних чисел  3. Знайти значення поліному p=23,5\*x^5 + 30\*x^4+10\*x+87,3. Значення x ввести з консолі. Результат вивести на консоль  4. Обчислити площу прямокутного трикутника за заданими катетами. Результат вивести на консоль. Використати методи класу Convert.  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс. |
| 4 | 1. Реалізувати функцію обчисленнячастки двох дійсних чисел  2. Реалізувати функцію піднесення до кубу частки двох дійсних чисел  3. Знайти значення поліному p=3,5\*x^4 + 3\*x^3+10\*x^2+8,3. Значення x ввести з консолі. Результат вивести на консоль  4. Ввести вартість покупки та суму, яку сплачує покупець. Вивести повідомлення про решту, яку має видати продавець. Результат вивести на консоль. Використати методи класу Convert.  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс. |
| 5 | 1. Реалізувати функцію обчислення суми двох дійсних чисел  2. Реалізувати функцію піднесення цілого числа в квадрат  3. Обчислити площу прямокутника за заданими сторонами  4. Обчислити значення функції z=a\*x^2+b\*y^2 при будь-яких значеннях параметрів і змінних. Результат вивести на консоль. Використати методи класу Convert.  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс. |
| 6 | 1. Реалізувати функцію обчислення різниці двох дійсних чисел  2. Реалізувати функцію піднесення до кубу різниці двох дійсних чисел  3. Обчислити значення функції z=5\*x^2+2\*x\*y+ y^2 при будь-яких значеннях параметрів і змінних. Результат вивести на консоль. Використати методи класу Convert.  4. Обчислити площу квадрату за заданою стороною  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс. |
| 7 | 1. Реалізувати функцію обчислення добутку двох дійсних чисел  2. Реалізувати функцію піднесення цілого числа в квадрат  3. Обчислити значення функції z=6\*x^2 - 12\*x\*y+ 25\*y^2 при будь-яких значеннях параметрів і змінних. Результат вивести на консоль. Використати методи класу Convert  4. Обчислити площу кола за радіусом  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс |
| 8 | 1. Реалізувати функцію обчислення суми трьох цілих чисел  2. Реалізувати функцію обчислення частки двох дійсних чисел  3. Обчислити значення функції y=2\*x2 + 10\*x+40. Змінна х вводиться з консолі  4. Для вказаної площі території і кількості жителів країни знайти щільність населення. Вхідні дані ввести з консолі. Результат вивести на консоль. Використати методи класу Convert  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс |
| 9 | 1. Реалізувати функцію піднесення в квадрат суми двох цілих чисел  2. Реалізувати функцію обчислення суми двох дійсних чисел  3. Обчислити довжину гіпотенузи і площу прямокутного трикутника по заданих катетах  4. Для вказаного об'єму і маси твердого тіла знайти його щільність. Вхідні дані ввести з консолі. Результат вивести на консоль. Використати методи класу Convert  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс |
| 10 | 1. Реалізувати функцію піднесення в квадрат різниці двох цілих чисел  2. Реалізувати функцію обчислення суми трьох чисел  3. Ввести вартість покупки та суму, яку сплачує покупець. Вивести повідомлення про решту, яку має видати продавець  4. Відомі координати на площині двох точок. Обчислити відстань між цими точками. Результат вивести на консоль. Використати методи класу Convert  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс |
| 11 | 1. Реалізувати функцію обчислення виразу x+y+z. Значення невідомих ввести з консолі.  2. Реалізувати функцію піднесення в квадрат добутку двох цілих чисел  3. Обчислити значення в усіх точках функції на заданому інтервалі. Вхідні дані ввести з консолі. Результат вивести на консоль. Функція задається таким чином: F(x) = x2 +4. Результат вивести на консоль. Використати методи класу Convert  4. Трикутник заданий координатами своїх вершин. Знайти периметр і площу трикутника. Результат вивести на консоль. Використати методи класу Convert  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс |
| 12 | 1. Реалізувати функцію піднесення в квадрат суми двох дійсних чисел  2. Реалізувати функцію обчислення суми двох дійсних чисел  3. Обчислити площу поверхні ( ) сфери, якщо відомий радіус r. Значення r ввести з консолі. Результат вивести на консоль. Використати методи класу Convert  4. Обчислити значення функції y=12\*x3 + 13\*x+10. Змінна х вводиться з консолі  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс |
| 13 | 1. Реалізувати функцію обчислення суми двох дійсних чисел  2. Реалізувати функцію піднесення в квадрат добутку двох дійсних чисел  3. Обчислити площу поверхні об’єм () сфери, якщо відомий радіус **r**. Значення r ввести з консолі. Результат вивести на консоль. Використати методи класу Convert  4. Знайти значення полінома p=x^4+6\*x^3+0,1\*x^2-12,3  в заданій точці **x**. Значення **x** ввести з консолі. Результат вивести на консоль.  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс |
| 14 | 1. Реалізувати функцію піднесення до кубу цілого числа  2. Реалізувати функцію піднесення в квадрат різниці двох дійсних чисел  3. Знайти значення полінома *ax3 +bx2 +cx + d.* Значення невідомих ініціалізувати в тексті програми. Результат вивести на консоль.  4. Знайти площу кільця по заданих зовнішньому і внутрішньому радіусах. Результат вивести на консоль. Використати методи класу Convert  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс |
| 15 | 1. Реалізувати функцію обчислення різниці двох цілих чисел  2. Реалізувати функцію піднесення в квадрат частки двох дійсних чисел  3. Обчислити значення функції y=(2x+25)/x. Значення x ввести з консолі. Результат вивести на консоль  4. Для вказаних значень сторін прямокутного паралелепіпеда: a,b,c – знайти його об'єм і площу бічної поверхні. Результат вивести на консоль. Використати методи класу Convert  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс |
| 16 | 1. Реалізувати функцію обчислення суми двох цілих чисел  2. Реалізувати функцію піднесення в квадрат частки двох дійсних чисел  3. Знайти значення полінома *ax3 +bx2 +cx + d.* Значення невідомих ініціалізувати в тексті програми. Результат вивести на консоль.  4. Відомі координати на площині двох точок. Обчислити відстань між цими точками. Результат вивести на консоль. Використати методи класу Convert  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс |
| 17 | 1. Реалізувати функцію піднесення до кубу цілого числа  2. Реалізувати функцію обчислення суми двох цілих чисел  3. Знайти значення полінома *45\*x3 +7\*x2 +15x + d.* Значення невідомих ініціалізувати в тексті програми. Результат вивести на консоль  4. Для вказаного об'єму і маси твердого тіла знайти його щільність. Вхідні дані ввести з консолі. Результат вивести на консоль. Використати методи класу Convert  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс |
| 18 | 1. Реалізувати функцію піднесення в квадрат частки двох дійсних чисел  2. Реалізувати функцію піднесення до кубу суми двох цілих чисел  3. Обчислити вираз y=7\*x2-2\*x+6 для будь-якого x. Значення x ініціалізувати в коді.  4. Ввести вартість покупки та суму, яку сплачує покупець. Вивести повідомлення про решту, яку має видати продавець. Результат вивести на консоль. Використати методи класу Convert.  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс |
| 19 | 1. Реалізувати функцію піднесення до кубу різниці двох цілих чисел  2. Реалізувати функцію обчислення суми двох цілих чисел  3. Обчислити вираз y=12\*а2+7а - 16.  4. Відомі координати на площині двох точок. Обчислити відстань між цими точками. Результат вивести на консоль. Використати методи класу Convert  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс |
| 20 | 1. Реалізувати функцію піднесення в квадрат різниці двох дійсних чисел  2. Реалізувати функцію піднесення до кубу добутку двох цілих чисел  3. Обчислити вираз y=42\*а3+7а2 - 1  4. Обчислити середнє арифметичне і середнє геометричне двох цілих чисел. Результат вивести на консоль. Використати методи класу Convert.  5. Вивести на консоль власні анкетні дані: Прізвище, ім'я, вік, група, курс |

**Література**

1.C# 2005 и платформа .Net 3.0 для профессионалов. Нейгел К., Ивьен Б. и др. –М.: ООО “И.Д. Вильямс", 2008. –1376 с.

2. Д.М. Златопольский. Сборник задач по программированию. 2-е издание. // СПб.: БХВ-Петербург, 2007.-240 с.

**Умовні оператори і цикли**

**Мета роботи:**

1. Створення консольних проектів з використанням операторів if
2. Створення консольних проектів з використанням операторів switch
3. Створення консольних проектів з використанням операторів циклу

**Методичні рекомендації до виконання лабораторної роботи**

1. Прочитайте лекцію 2. Спробуйте відповісти на наведені в кінці лекції запитання

2. Прочитайте цю лабораторну роботу та виконайте наведені в ній приклади (вони всі працездатні)

3. При виконанні завдань зверніть увагу на оператор switch (перемикач). В С# тип виразу в цьому операторі може бути не тільки цілочисельним, а і типу char або string.

4. Також зверніть увагу на новий тип циклу **foreach,** зручний при роботі з масивами.

**Порядок виконання роботи**

1. Створити директорію Lab3, в якій будуть розміщуватися проекти цієї лабораторної роботи.

2. Виконати завдання свого варіанту у вигляді окремих проектів

3. Зафіксувати результати у Контрольній роботі 1.

**Приклади виконання завдань**

**1. Створення консольних проектів з використанням операторів if**

Створимо консольний проект для організації діалогу таким чином, щоб реалізувати наведений нижче алгоритм.

1. Вивести на консоль запрошення для введення імені.

2. Ввести своє ім'я і зберегти в текстовому рядку: string myName.

3. Якщо нічого не введено, то вивести повідомлення про це і завершити роботу.

4. Якщо щось введено, то вивести рядок привітання.

5. Вивести рядок із запитом віку

6. Якщо нічого не введено, то вивести повідомлення про це і завершити роботу.

7. Якщо вік введено, привести число до цілого (int myAge).

8. Якщо число myAge<15, вивести повідомлення "Ви ще не студент. "

9. Якщо число myAge>40, вивести повідомлення "Вчитися ніколи не пізно!"

**Код програми може мати вигляд:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleHello1

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Введіть ім'я");

string myName = Console.ReadLine();

if (myName.Length == 0)

Console.WriteLine("Ви нічого не ввели, прощавайте");

else

Console.WriteLine("Здрастуйте, "+ myName);

Console.WriteLine("Скільки Вам років?");

int myAge = int.Parse(Console.ReadLine());

if (myAge == 0)

Console.WriteLine("Ви нічого не ввели, прощавайте");

else

{

if (myAge<15)

Console.WriteLine("Ви ще не студент");

else if (myAge>40)

Console.WriteLine("Вчитися ніколи не пізно!");

else

Console.WriteLine("Ваш вік" +myAge);

}

Console.ReadKey();

}

}

}

В цьому прикладі використовується декілька операторів розгалуження **if** (останні три вкладені). Зверніть увагу, що у C# в операторі **if** діють ті самі правила синтаксису, що і в С++, якщо в тілі оператора є лише один оператор блоку, то дужки не потрібні, наприклад:

if (myName.Length == 0)

Console.WriteLine("Ви нічого не ввели, прощавайте");

else

Console.WriteLine("Здрастуйте, "+ myName);

Крім цього виконується явне перетворення типів з типу string в тип int з використанням методу Parse і неявне при виведенні віку на консоль.

**2. Створення консольних проектів з використанням операторів switch**

Розробимо простий консольний калькулятор для виконання арифметичних операцій. Вхідні дані будемо вводити з консолі, результати виводити на консоль. Приклад коду з лекції 4.

**Консольний калькулятор на 4 дії**

using System;

namespace ConsoleCalculator

{ class Program

{ static void Main()

{

string buf;

double a, b, res;

Console.WriteLine( "Введіть перший операнд:" );

a = double.Parse( Console.ReadLine() );

Console.WriteLine( "Введіть знак операції" );

char op = (char)Console.Read();

Console.ReadLine();

Console.WriteLine( "Введіть другий операнд:" );

b = double.Parse( Console.ReadLine() );

bool ok = true;

switch (op)

{

case '+' : res = a + b; break;

case '-' : res = a - b; break;

case '\*' : res = a \* b; break;

case '/' : res = a / b; break;

default : res = double.NaN; ok = false; break;

}

if (ok) Console.WriteLine( "Результат: " + res );

else Console.WriteLine( "Неприпустима операція" );

}

}

}

Зверніть увагу, що тип виразу в операторі switch є **char**. Також зверніть увагу, що перетворення з типу string до типу char повинно бути явним, тому що тип char є типом-значенням, а тип string – посилковим.

**char op = (char)Console.Read();**

Для виходу з оператора switch при виконанні умови використовується оператор переходу break.

**3. Створення консольних проектів з використанням операторів циклу**

**3.1. Цикл з передумовою while**

Потрібно написати програму, яка виводить для аргументу х, що змінюється в заданих границях із заданим кроком, таблицю значень наступної функції:

X={t, x<0

tx, 0<=x<10

2t, x>10 }

Назвемо початкове значення аргументу Xn, кінцеве значення аргументу — Xk, крок зміни аргументу — dX і параметр t. Всі величини є дійсні числа (double). Програма повинна виводити таблицю, що складається з двох стовпців: значень аргументу і відповідних ним значень функції.

namespace ConsoleApplication1

{

class Program

{

static void Main(string[] args)

{

double Xn = -2, Xk = 12, dX = 2, t = 2, y;

Console.WriteLine("| x | y |"); // заголовок таблиці

double x = Xn;

while (x <= Xk)

{

y = t;

if (x >= 0 && x < 10) y = t \* x;

if (x >= 10) y = 2 \* t;

Console.WriteLine("| {0,6} | {1,6} |", x, y);

x += dX;

}

Console.ReadKey();

}

}

}

Зверніть увагу, що в тілі циклу використовується два оператори if. На кожному кроці циклу обчислюється значення функції. Цикл завершиться коли умова циклу не буде виконана (тобто значення x стане більше 12).

**3.2. Цикл з пост-умовою do…while**

Цей тип циклу застосовується в тих випадках, коли тіло циклу необхідно обов'язково виконати хоч б один раз.

Приклад 4.7. з теоретичної частини л/р демонструє використання цього циклу. В цьому прикладі на консоль виводиться текст "Будете вчитися?" до тих пір поки не буде введено"y".

Приклад 4.7 лекції 4

namespace ConsoleApplication1

{

class Program

{

static void Main(string[] args)

{

char answer;

do

{

Console.WriteLine("Будете вчитися?");

answer = (char)Console.Read();

Console.ReadLine();

} while (answer != 'y');

}

}

}

**3.3. Цикл з параметром (for)**

Цей тип циклу використовується коли відомі границі циклу (нижня і верхня).

Приклад 4.8 теоретичної частини л/р демонструє використання цього циклу. В цьому прикладі обчислюється сума чисел від 1 до 100. Спочатку створюється змінна цілого типу s, яка ініціалізується нулем. Потім в тілі циклу на кожному кроці до неї додається чергове число: 1,2,3,....100.

int s = 0;

for ( int i = 1; i <= 100; i++ )

s += i;

Зверніть увагу, що в тілі циклу відсутні фігурні дужки блоку. Це тому, що в циклі лише один оператор. Також зверніть увагу на використання операції інкремента:

s += i;

замість призначення: s=s+i;

Обидва оператори дозволені в C# так само як і в C++.

Розглянемо ще один приклад використання оператора циклу з параметром.

**Постановка задачі.**

Написати метод обчислення функції sin(x), використовуючи розкладання в ряд Тейлора за формулою:

\sin x = \sum^{\infin}_{n=0} \frac{(-1)^n}{(2n+1)!} x^{2n+1}

При реалізації задачі використати цикл **for**.

Число x – це значення кута в радіанах, n – кількість членів ряду. Числа **x, n** ввести з консолі. Обчислення факторіала виконати в окремому методі. Порівняти отримане значення із стандартним методом обчислення sin(x).

В цьому прикладі для обчислення ступеня використовується метод Math.Pow(). Статичний клас Math містить методи реалізації математичних функцій.

**Код програми може бути таким:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace My\_sin

{

class Program

{

static double Calc\_sin(double x, int n)

{

//обчислення розкладання sin в ряд

double result =0;

for (int i = 0; i < n; i++)

{

result=result+(Math.Pow((-1),i)\*Math.Pow(x,(2\*i+1)))/F(2\*i+1);

}

return result;

}

static double F(int n)

{

double tmp = 1;

for (int i = 1; i <= n; i++)

{

tmp = tmp \* i;

}

return tmp;

}

static void Main(string[] args)

{

Console.WriteLine("Введіть x - кут в радіанах");

double x = double.Parse(Console.ReadLine());

Console.WriteLine("Введіть показник ступеня n");

int n = int.Parse(Console.ReadLine());

//виклик методу обчислення sin(x) через ряд

double my\_sinus = Calc\_sin(x,n);

//виклик методу з класу Math

double sinus = Math.Sin(x);

double delta = sinus - my\_sinus;

Console.WriteLine("my\_sinus= {0},sin={1},delta={2}", my\_sinus, sinus, delta);

Console.ReadKey();

}

}

}

**6. Варіанти завдань для Лабораторної роботи № 3**

**Номер варіанта відповідає номеру прізвища студента у списку групи**

**Завдання 1 (для всіх)**

Модифікувати консольний калькулятор таким чином, щоб завершення програми виконувалося за запитом програми.

‘Завершити роботу? Введіть **y**, інакше enter’

Використати цикл з пост-умовою (do …while).

**Індивідуальні завдання**

Кожний студент повинен виконати по три завдання свого варіанту

|  |  |
| --- | --- |
| **№ варіан-ту** | **Зміст завдання** |
| 1 | **1. Оператор if**  Написати програму обчислення значень функції у заданій точці x. Значення x вводиться з консолі. Результат вивести на консоль.  Функція F задається таким чином:  F(x) = 4x-1, x<0  F(x)= 25x+10, x>0  F(x)=0, x=0  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа оголосити в коді програми.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера дня тижня (1,2,...7) виводить на екран його назву (понеділок,...)  **4. Оператор циклу**  Обчислити суму чисел від заданого числа до 50. Число ввести з консолі. Якщо введене число >50, вивести повідомлення про помилку, якщо<50 , обчислити суму, якщо =50 – вивести 50. |
| 2 | **1. Оператор if**  Обчислити значення в усіх точках функції на заданому інтервалі. Вхідні дані ввести з консолі. Результат вивести на консоль.  Функція F задається таким чином:  F(x) = x2 +4 , x>0,  F(x)=0, x<=0  Якщо нічого не введено – вивести повідомлення “Потрібно було ввести число” і завершити роботу.  **2. Оператор if.**  Знайти найменше з трьох чисел**.** Числа ввести з консолі.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера місяця (1,2,...12) виводить на екран його назву (січень,...грудень).  **4. Оператор циклу**  Дано натуральне число n. Обчислити добуток перших n членів ряду:  к=2. |
| 3 | **1. Оператор if**  Обчислити значення в усіх точках функції на заданому інтервалі. Вхідні дані ввести з консолі. Результат вивести на консоль.  Функція F задається таким чином:  F(x) = sin^2\*x , x>0  F(x) = 1+2\*sin^2x, x<=0  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа оголосити в коді програми.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера місяця (1,2,...12) виводить на екран пору року (зима, весна....)  **4. Оператор циклу**  Дано натуральне число n. Обчислити: де |
| 4 | **1. Оператор if**  Обчислити значення в усіх точках функції на заданому інтервалі. Вхідні дані ввести з консолі. Результат вивести на консоль.  Функція F задається таким чином:  F(x) = sin^2\*x , x>0  F(x) = 1-2\*sin^2x, x<=0  **2. Оператор if**  Знайти найменше з трьох чисел. Числа ввести з консолі.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера дня тижня (1,2,...7) виводить на екран його назву (понеділок,...)  **4. Оператор циклу**  Відомі оцінки студента по 10 дисциплінах. Обчислити загальну суму балів, найгіршу і найкращу оцінку |
| 5 | **1. Оператор if**  Обчислити значення в усіх точках функції на заданому інтервалі. Вхідні дані ввести з консолі. Результат вивести на консоль.  Функція F задається таким чином:  F(x) = sin^2\*x+1 , x>0  F(x) = 1-2\*sinx, x<=0  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа ввести з консолі.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера кольору у спектрі (1,2,...7) виводить його назву (червоний, помаранчовий, жовтий, зелений, блакитний, синій, фіолетовий) і код RGB  **4. Оператор циклу**  Відомі оцінки студента по 5 дисциплінах. Обчислити суму балів і середню оцінку. |
| 6 | **1. Оператор if**  При заданому радіусу знайти площу кола. При заданій стороні квадрата знайти його площу. Визначити:  а) чи можна вписати коло в квадрат?  б) чи можна вписати квадрат в коло?  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа ввести з консолі.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера кольору у спектрі (1,2,...7) виводить його назву (червоний, помаранчовий, жовтий,.зелений, блакитний, синій, фіолетовий) і код RGB  **4. Оператор циклу**  Дано натуральне число n і дійсне число x >0. Обчислити: |
| 7 | **1. Оператор if**  Відомі рік і місяць народження людини. Визначити вік людини на вказану дату (сьогоднішній день). У разі співпадіння номерів місяців вважати, що пройшов повний рік.  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа оголосити в коді програми.  **3. Оператор switch**  Написати програму, яка в залежності від назви кольору (червоний, помаранчовий, жовтий, зелений, блакитний, синій, фіолетовий) виводить його код RGB  **4. Оператор циклу**  Обчислити Fn - число Фібоначчі з номером n, де F1 = 1; F2 = 1; Fk = Fk-1 + Fk-2 для k>2. |
| 8 | **1. Оператор if**  Ввести два числа. Визначити:  а) яке з них більше  б) яке з них менше.  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа оголосити в коді програми.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера дня тижня (1,2,...7) виводить на екран його назву (понеділок,...)  **4. Оператор циклу**  Дано натуральне число n. Обчислити суму перших 2n членів ряду: |
| 9 | **1. Оператор if**  Визначити максимальне і мінімальне значення з двох різних дійсних чисел  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа оголосити в коді програми.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера кольору у спектрі (1,2,...7) виводить його назву (червоний, помаранчовий, жовтий, зелений, блакитний, синій, фіолетовий) і код RGB  **4. Оператор циклу**  Дано натуральне число n і дійсне число x.  Обчислити: |
| 10 | **1. Оператор if**  Дано радіус кола і сторона квадрату. У якої фігури більша площа?  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа оголосити в коді програми.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера дня тижня (1,2,...7) виводить на екран його назву (понеділок,...)  **4. Оператор циклу**  Дано натуральне число n і дійсне число x >0. Обчислити: |
| 11 | **1. Оператор if**  Дано об'єми і маси двох тіл з різних матеріалів. Матеріал якого з тіл має більшу щільність?  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа оголосити в коді програми.  **3. Оператор switch**  Написати програму, яка для кожного члена сім'ї виводить день народження.  **4. Оператор циклу**  Дано натуральне число n. Обчислити добуток перших n членів ряду: |
| 12 | **1. Оператор if**  Визначити чи є трикутник із сторонами A,B,C рівностороннім  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа оголосити в коді програми.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера кольору у спектрі (1,2,...7) виводить його назву (червоний, помаранчовий, жовтий,.зелений, блакитний, синій, фіолетовий) і код RGB  **4. Оператор циклу**  Дано натуральне число n. Обчислити: де |
| 13 | **1. Оператор if**  Написати програму обчислення значень функції у заданій точці x. Значення x вводиться з консолі. Результат вивести на консоль.  Функція F задається таким чином:  F(x) = 2x +10, x<0  F(x)= 5x^2+10, x>0  F(x)=1, x=0  **2. Оператор if**  Є три цілі числа. Визначити, чи є серед них хоча б одна пара рівних.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера дня тижня (1,2,...7) виводить на екран його назву (понеділок,...)  **4. Оператор циклу**  Дано натуральні числа n і m (50<m<n). Обчислити: де |
| 14 | **1. Оператор if**  Написати програму обчислення значень функції у заданій точці x. Значення x вводиться з консолі. Результат вивести на консоль.  Функція F задається таким чином:  F(x) = 2x +10, x<0  F(x)= 5x^2+10, x>0  F(x)=1, x=0  2.**Оператор if**  Є три дійсні числа. Визначити яке з них найбільше, найменше і середнє.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера кольору у спектрі (1,2,...7) виводить його назву (червоний, помаранчовий, жовтий,.зелений, блакитний, синій, фіолетовий) і код RGB  **4. Оператор циклу**  Дано натуральне число n і дійсне число x.  Обчислити: |
| 15 | **1. Оператор if**  Знайти максимальне і мінімальне з трьох дійсних чисел  **2. Оператор switch**  Написати програму, яка в залежності від порядкового номера кольору у спектрі (1,2,...7) виводить його назву (червоний, помаранчовий, жовтий,.зелений, блакитний, синій, фіолетовий) і код RGB  **3. Оператор циклу**  Дано натуральне число n і дійсне число x >0. Обчислити: |
| 16 | **1. Оператор if**  Написати програму обчислення значень функції у заданій точці x. Значення x вводиться з консолі. Результат вивести на консоль.  Функція F задається таким чином:  F(x) = 2x +10, x<0  F(x)= 5x^2+10, x>0  F(x)=1, x=0  **2. Оператор switch**  Написати програму, яка в залежності від марки мобільного телефону виводить фірму-виробника  **3. Оператор циклу**  Знайти суму двох найбільших з трьох різних чисел |
| 17 | **1. Оператор if**  При заданому радіусу знайти площу кола. При заданій стороні квадрата знайти його площу. Визначити:  а) чи можна вписати коло в квадрат?  б) чи можна вписати квадрат в коло?  **2. Оператор switch**  Написати програму, яка в залежності від порядкового номера дня тижня (1,2,...7) виводить на екран його назву (понеділок,...)  **3. Оператор циклу**  Обчислити Fn - число Фібоначчі з номером n, де F1 = 1; F2 = 1; Fk = Fk-1 + Fk-2 для k>2. |
| 18 | **1. Оператор if**  Знайти добуток двох найменших з трьох різних чисел  **2. Оператор switch**  Написати програму, яка в залежності від знаку операції (+,-,\*,/) виводить на екран її назву (Додавання,....)  **3. Оператор циклу**  Написати метод обчислення ступеню n числа, xn /x (для n>0). Число n вводиться з консолі, результат виводиться на консоль. |
| 19 | **1. Оператор if**  Є три дійсні числа. Визначити яке з них найбільше, найменше і середнє.  **2. Оператор switch**  Написати програму, яка в залежності від знаку логічної операції (<,>,<=,==,!=) виводить на екран її назву (Менше,Більше,...)  **3. Оператор циклу**  Дано натуральне число n і дійсне число x. Обчислити: |
| 20 | **1. Оператор if**  Відомі рік і місяць народження людини. Визначити вік людини на вказану дату (сьогодняшній день). У разі співпадіння номерів місяців вважати, що пройшов повний рік.  **2. Оператор switch**  Написати програму, яка в залежності від порядкового номера кольору у спектрі (1,2,...7) виводить його назву (червоний, помаранчовий, жовтий,.зелений, блакитний, синій, фіолетовий) і код RGB  **3. Оператор циклу**  Дано натуральні числа n і m (50<m<n). Обчислити:  де |

**Література**

1. Лекція 3

2.C# 2005 и платформа .Net 3.0 для профессионалов. Нейгел К., Ивьен Б. и др. –М.: ООО “И.Д. Вильямс", 2008. –1376 с.

3. Д.М. Златопольский. Сборник задач по программированию. 2-е издание. // СПб.: БХВ-Петербург, 2007.-240 с.

**Лабораторна робота №3**

**Умовні оператори і цикли**

**Мета роботи:**

1. Створення консольних проектів з використанням операторів if
2. Створення консольних проектів з використанням операторів switch
3. Створення консольних проектів з використанням операторів циклу

**Методичні рекомендації до виконання лабораторної роботи**

1. Прочитайте лекцію 2. Спробуйте відповісти на наведені в кінці лекції запитання

2. Прочитайте цю лабораторну роботу та виконайте наведені в ній приклади (вони всі працездатні)

3. При виконанні завдань зверніть увагу на оператор switch (перемикач). В С# тип виразу в цьому операторі може бути не тільки цілочисельним, а і типу char або string.

4. Також зверніть увагу на новий тип циклу **foreach,** зручний при роботі з масивами.

**Порядок виконання роботи**

1. Створити директорію Lab3, в якій будуть розміщуватися проекти цієї лабораторної роботи.

2. Виконати завдання свого варіанту у вигляді окремих проектів

3. Зафіксувати результати у Контрольній роботі 1.

**Приклади виконання завдань**

**1. Створення консольних проектів з використанням операторів if**

Створимо консольний проект для організації діалогу таким чином, щоб реалізувати наведений нижче алгоритм.

1. Вивести на консоль запрошення для введення імені.

2. Ввести своє ім'я і зберегти в текстовому рядку: string myName.

3. Якщо нічого не введено, то вивести повідомлення про це і завершити роботу.

4. Якщо щось введено, то вивести рядок привітання.

5. Вивести рядок із запитом віку

6. Якщо нічого не введено, то вивести повідомлення про це і завершити роботу.

7. Якщо вік введено, привести число до цілого (int myAge).

8. Якщо число myAge<15, вивести повідомлення "Ви ще не студент. "

9. Якщо число myAge>40, вивести повідомлення "Вчитися ніколи не пізно!"

**Код програми може мати вигляд:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleHello1

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Введіть ім'я");

string myName = Console.ReadLine();

if (myName.Length == 0)

Console.WriteLine("Ви нічого не ввели, прощавайте");

else

Console.WriteLine("Здрастуйте, "+ myName);

Console.WriteLine("Скільки Вам років?");

int myAge = int.Parse(Console.ReadLine());

if (myAge == 0)

Console.WriteLine("Ви нічого не ввели, прощавайте");

else

{

if (myAge<15)

Console.WriteLine("Ви ще не студент");

else if (myAge>40)

Console.WriteLine("Вчитися ніколи не пізно!");

else

Console.WriteLine("Ваш вік" +myAge);

}

Console.ReadKey();

}

}

}

В цьому прикладі використовується декілька операторів розгалуження **if** (останні три вкладені). Зверніть увагу, що у C# в операторі **if** діють ті самі правила синтаксису, що і в С++, якщо в тілі оператора є лише один оператор блоку, то дужки не потрібні, наприклад:

if (myName.Length == 0)

Console.WriteLine("Ви нічого не ввели, прощавайте");

else

Console.WriteLine("Здрастуйте, "+ myName);

Крім цього виконується явне перетворення типів з типу string в тип int з використанням методу Parse і неявне при виведенні віку на консоль.

**2. Створення консольних проектів з використанням операторів switch**

Розробимо простий консольний калькулятор для виконання арифметичних операцій. Вхідні дані будемо вводити з консолі, результати виводити на консоль. Приклад коду з лекції 4.

**Консольний калькулятор на 4 дії**

using System;

namespace ConsoleCalculator

{ class Program

{ static void Main()

{

string buf;

double a, b, res;

Console.WriteLine( "Введіть перший операнд:" );

a = double.Parse( Console.ReadLine() );

Console.WriteLine( "Введіть знак операції" );

char op = (char)Console.Read();

Console.ReadLine();

Console.WriteLine( "Введіть другий операнд:" );

b = double.Parse( Console.ReadLine() );

bool ok = true;

switch (op)

{

case '+' : res = a + b; break;

case '-' : res = a - b; break;

case '\*' : res = a \* b; break;

case '/' : res = a / b; break;

default : res = double.NaN; ok = false; break;

}

if (ok) Console.WriteLine( "Результат: " + res );

else Console.WriteLine( "Неприпустима операція" );

}

}

}

Зверніть увагу, що тип виразу в операторі switch є **char**. Також зверніть увагу, що перетворення з типу string до типу char повинно бути явним, тому що тип char є типом-значенням, а тип string – посилковим.

**char op = (char)Console.Read();**

Для виходу з оператора switch при виконанні умови використовується оператор переходу break.

**3. Створення консольних проектів з використанням операторів циклу**

**3.1. Цикл з передумовою while**

Потрібно написати програму, яка виводить для аргументу х, що змінюється в заданих границях із заданим кроком, таблицю значень наступної функції:

X={t, x<0

tx, 0<=x<10

2t, x>10 }

Назвемо початкове значення аргументу Xn, кінцеве значення аргументу — Xk, крок зміни аргументу — dX і параметр t. Всі величини є дійсні числа (double). Програма повинна виводити таблицю, що складається з двох стовпців: значень аргументу і відповідних ним значень функції.

namespace ConsoleApplication1

{

class Program

{

static void Main(string[] args)

{

double Xn = -2, Xk = 12, dX = 2, t = 2, y;

Console.WriteLine("| x | y |"); // заголовок таблиці

double x = Xn;

while (x <= Xk)

{

y = t;

if (x >= 0 && x < 10) y = t \* x;

if (x >= 10) y = 2 \* t;

Console.WriteLine("| {0,6} | {1,6} |", x, y);

x += dX;

}

Console.ReadKey();

}

}

}

Зверніть увагу, що в тілі циклу використовується два оператори if. На кожному кроці циклу обчислюється значення функції. Цикл завершиться коли умова циклу не буде виконана (тобто значення x стане більше 12).

**3.2. Цикл з пост-умовою do…while**

Цей тип циклу застосовується в тих випадках, коли тіло циклу необхідно обов'язково виконати хоч б один раз.

Приклад 4.7. з теоретичної частини л/р демонструє використання цього циклу. В цьому прикладі на консоль виводиться текст "Будете вчитися?" до тих пір поки не буде введено"y".

Приклад 4.7 лекції 4

namespace ConsoleApplication1

{

class Program

{

static void Main(string[] args)

{

char answer;

do

{

Console.WriteLine("Будете вчитися?");

answer = (char)Console.Read();

Console.ReadLine();

} while (answer != 'y');

}

}

}

**3.3. Цикл з параметром (for)**

Цей тип циклу використовується коли відомі границі циклу (нижня і верхня).

Приклад 4.8 теоретичної частини л/р демонструє використання цього циклу. В цьому прикладі обчислюється сума чисел від 1 до 100. Спочатку створюється змінна цілого типу s, яка ініціалізується нулем. Потім в тілі циклу на кожному кроці до неї додається чергове число: 1,2,3,....100.

int s = 0;

for ( int i = 1; i <= 100; i++ )

s += i;

Зверніть увагу, що в тілі циклу відсутні фігурні дужки блоку. Це тому, що в циклі лише один оператор. Також зверніть увагу на використання операції інкремента:

s += i;

замість призначення: s=s+i;

Обидва оператори дозволені в C# так само як і в C++.

Розглянемо ще один приклад використання оператора циклу з параметром.

**Постановка задачі.**

Написати метод обчислення функції sin(x), використовуючи розкладання в ряд Тейлора за формулою:

\sin x = \sum^{\infin}_{n=0} \frac{(-1)^n}{(2n+1)!} x^{2n+1}

При реалізації задачі використати цикл **for**.

Число x – це значення кута в радіанах, n – кількість членів ряду. Числа **x, n** ввести з консолі. Обчислення факторіала виконати в окремому методі. Порівняти отримане значення із стандартним методом обчислення sin(x).

В цьому прикладі для обчислення ступеня використовується метод Math.Pow(). Статичний клас Math містить методи реалізації математичних функцій.

**Код програми може бути таким:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace My\_sin

{

class Program

{

static double Calc\_sin(double x, int n)

{

//обчислення розкладання sin в ряд

double result =0;

for (int i = 0; i < n; i++)

{

result=result+(Math.Pow((-1),i)\*Math.Pow(x,(2\*i+1)))/F(2\*i+1);

}

return result;

}

static double F(int n)

{

double tmp = 1;

for (int i = 1; i <= n; i++)

{

tmp = tmp \* i;

}

return tmp;

}

static void Main(string[] args)

{

Console.WriteLine("Введіть x - кут в радіанах");

double x = double.Parse(Console.ReadLine());

Console.WriteLine("Введіть показник ступеня n");

int n = int.Parse(Console.ReadLine());

//виклик методу обчислення sin(x) через ряд

double my\_sinus = Calc\_sin(x,n);

//виклик методу з класу Math

double sinus = Math.Sin(x);

double delta = sinus - my\_sinus;

Console.WriteLine("my\_sinus= {0},sin={1},delta={2}", my\_sinus, sinus, delta);

Console.ReadKey();

}

}

}

**6. Варіанти завдань для Лабораторної роботи № 3**

**Номер варіанта відповідає номеру прізвища студента у списку групи**

**Завдання 1 (для всіх)**

Модифікувати консольний калькулятор таким чином, щоб завершення програми виконувалося за запитом програми.

‘Завершити роботу? Введіть **y**, інакше enter’

Використати цикл з пост-умовою (do …while).

**Індивідуальні завдання**

Кожний студент повинен виконати по три завдання свого варіанту

|  |  |
| --- | --- |
| **№ варіан-ту** | **Зміст завдання** |
| 1 | **1. Оператор if**  Написати програму обчислення значень функції у заданій точці x. Значення x вводиться з консолі. Результат вивести на консоль.  Функція F задається таким чином:  F(x) = 4x-1, x<0  F(x)= 25x+10, x>0  F(x)=0, x=0  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа оголосити в коді програми.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера дня тижня (1,2,...7) виводить на екран його назву (понеділок,...)  **4. Оператор циклу**  Обчислити суму чисел від заданого числа до 50. Число ввести з консолі. Якщо введене число >50, вивести повідомлення про помилку, якщо<50 , обчислити суму, якщо =50 – вивести 50. |
| 2 | **1. Оператор if**  Обчислити значення в усіх точках функції на заданому інтервалі. Вхідні дані ввести з консолі. Результат вивести на консоль.  Функція F задається таким чином:  F(x) = x2 +4 , x>0,  F(x)=0, x<=0  Якщо нічого не введено – вивести повідомлення “Потрібно було ввести число” і завершити роботу.  **2. Оператор if.**  Знайти найменше з трьох чисел**.** Числа ввести з консолі.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера місяця (1,2,...12) виводить на екран його назву (січень,...грудень).  **4. Оператор циклу**  Дано натуральне число n. Обчислити добуток перших n членів ряду:  к=2. |
| 3 | **1. Оператор if**  Обчислити значення в усіх точках функції на заданому інтервалі. Вхідні дані ввести з консолі. Результат вивести на консоль.  Функція F задається таким чином:  F(x) = sin^2\*x , x>0  F(x) = 1+2\*sin^2x, x<=0  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа оголосити в коді програми.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера місяця (1,2,...12) виводить на екран пору року (зима, весна....)  **4. Оператор циклу**  Дано натуральне число n. Обчислити: де |
| 4 | **1. Оператор if**  Обчислити значення в усіх точках функції на заданому інтервалі. Вхідні дані ввести з консолі. Результат вивести на консоль.  Функція F задається таким чином:  F(x) = sin^2\*x , x>0  F(x) = 1-2\*sin^2x, x<=0  **2. Оператор if**  Знайти найменше з трьох чисел. Числа ввести з консолі.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера дня тижня (1,2,...7) виводить на екран його назву (понеділок,...)  **4. Оператор циклу**  Відомі оцінки студента по 10 дисциплінах. Обчислити загальну суму балів, найгіршу і найкращу оцінку |
| 5 | **1. Оператор if**  Обчислити значення в усіх точках функції на заданому інтервалі. Вхідні дані ввести з консолі. Результат вивести на консоль.  Функція F задається таким чином:  F(x) = sin^2\*x+1 , x>0  F(x) = 1-2\*sinx, x<=0  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа ввести з консолі.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера кольору у спектрі (1,2,...7) виводить його назву (червоний, помаранчовий, жовтий, зелений, блакитний, синій, фіолетовий) і код RGB  **4. Оператор циклу**  Відомі оцінки студента по 5 дисциплінах. Обчислити суму балів і середню оцінку. |
| 6 | **1. Оператор if**  При заданому радіусу знайти площу кола. При заданій стороні квадрата знайти його площу. Визначити:  а) чи можна вписати коло в квадрат?  б) чи можна вписати квадрат в коло?  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа ввести з консолі.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера кольору у спектрі (1,2,...7) виводить його назву (червоний, помаранчовий, жовтий,.зелений, блакитний, синій, фіолетовий) і код RGB  **4. Оператор циклу**  Дано натуральне число n і дійсне число x >0. Обчислити: |
| 7 | **1. Оператор if**  Відомі рік і місяць народження людини. Визначити вік людини на вказану дату (сьогоднішній день). У разі співпадіння номерів місяців вважати, що пройшов повний рік.  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа оголосити в коді програми.  **3. Оператор switch**  Написати програму, яка в залежності від назви кольору (червоний, помаранчовий, жовтий, зелений, блакитний, синій, фіолетовий) виводить його код RGB  **4. Оператор циклу**  Обчислити Fn - число Фібоначчі з номером n, де F1 = 1; F2 = 1; Fk = Fk-1 + Fk-2 для k>2. |
| 8 | **1. Оператор if**  Ввести два числа. Визначити:  а) яке з них більше  б) яке з них менше.  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа оголосити в коді програми.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера дня тижня (1,2,...7) виводить на екран його назву (понеділок,...)  **4. Оператор циклу**  Дано натуральне число n. Обчислити суму перших 2n членів ряду: |
| 9 | **1. Оператор if**  Визначити максимальне і мінімальне значення з двох різних дійсних чисел  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа оголосити в коді програми.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера кольору у спектрі (1,2,...7) виводить його назву (червоний, помаранчовий, жовтий, зелений, блакитний, синій, фіолетовий) і код RGB  **4. Оператор циклу**  Дано натуральне число n і дійсне число x.  Обчислити: |
| 10 | **1. Оператор if**  Дано радіус кола і сторона квадрату. У якої фігури більша площа?  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа оголосити в коді програми.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера дня тижня (1,2,...7) виводить на екран його назву (понеділок,...)  **4. Оператор циклу**  Дано натуральне число n і дійсне число x >0. Обчислити: |
| 11 | **1. Оператор if**  Дано об'єми і маси двох тіл з різних матеріалів. Матеріал якого з тіл має більшу щільність?  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа оголосити в коді програми.  **3. Оператор switch**  Написати програму, яка для кожного члена сім'ї виводить день народження.  **4. Оператор циклу**  Дано натуральне число n. Обчислити добуток перших n членів ряду: |
| 12 | **1. Оператор if**  Визначити чи є трикутник із сторонами A,B,C рівностороннім  **2. Оператор if**  Знайти найбільше з трьох чисел. Числа оголосити в коді програми.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера кольору у спектрі (1,2,...7) виводить його назву (червоний, помаранчовий, жовтий,.зелений, блакитний, синій, фіолетовий) і код RGB  **4. Оператор циклу**  Дано натуральне число n. Обчислити: де |
| 13 | **1. Оператор if**  Написати програму обчислення значень функції у заданій точці x. Значення x вводиться з консолі. Результат вивести на консоль.  Функція F задається таким чином:  F(x) = 2x +10, x<0  F(x)= 5x^2+10, x>0  F(x)=1, x=0  **2. Оператор if**  Є три цілі числа. Визначити, чи є серед них хоча б одна пара рівних.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера дня тижня (1,2,...7) виводить на екран його назву (понеділок,...)  **4. Оператор циклу**  Дано натуральні числа n і m (50<m<n). Обчислити: де |
| 14 | **1. Оператор if**  Написати програму обчислення значень функції у заданій точці x. Значення x вводиться з консолі. Результат вивести на консоль.  Функція F задається таким чином:  F(x) = 2x +10, x<0  F(x)= 5x^2+10, x>0  F(x)=1, x=0  2.**Оператор if**  Є три дійсні числа. Визначити яке з них найбільше, найменше і середнє.  **3. Оператор switch**  Написати програму, яка в залежності від порядкового номера кольору у спектрі (1,2,...7) виводить його назву (червоний, помаранчовий, жовтий,.зелений, блакитний, синій, фіолетовий) і код RGB  **4. Оператор циклу**  Дано натуральне число n і дійсне число x.  Обчислити: |
| 15 | **1. Оператор if**  Знайти максимальне і мінімальне з трьох дійсних чисел  **2. Оператор switch**  Написати програму, яка в залежності від порядкового номера кольору у спектрі (1,2,...7) виводить його назву (червоний, помаранчовий, жовтий,.зелений, блакитний, синій, фіолетовий) і код RGB  **3. Оператор циклу**  Дано натуральне число n і дійсне число x >0. Обчислити: |
| 16 | **1. Оператор if**  Написати програму обчислення значень функції у заданій точці x. Значення x вводиться з консолі. Результат вивести на консоль.  Функція F задається таким чином:  F(x) = 2x +10, x<0  F(x)= 5x^2+10, x>0  F(x)=1, x=0  **2. Оператор switch**  Написати програму, яка в залежності від марки мобільного телефону виводить фірму-виробника  **3. Оператор циклу**  Знайти суму двох найбільших з трьох різних чисел |
| 17 | **1. Оператор if**  При заданому радіусу знайти площу кола. При заданій стороні квадрата знайти його площу. Визначити:  а) чи можна вписати коло в квадрат?  б) чи можна вписати квадрат в коло?  **2. Оператор switch**  Написати програму, яка в залежності від порядкового номера дня тижня (1,2,...7) виводить на екран його назву (понеділок,...)  **3. Оператор циклу**  Обчислити Fn - число Фібоначчі з номером n, де F1 = 1; F2 = 1; Fk = Fk-1 + Fk-2 для k>2. |
| 18 | **1. Оператор if**  Знайти добуток двох найменших з трьох різних чисел  **2. Оператор switch**  Написати програму, яка в залежності від знаку операції (+,-,\*,/) виводить на екран її назву (Додавання,....)  **3. Оператор циклу**  Написати метод обчислення ступеню n числа, xn /x (для n>0). Число n вводиться з консолі, результат виводиться на консоль. |
| 19 | **1. Оператор if**  Є три дійсні числа. Визначити яке з них найбільше, найменше і середнє.  **2. Оператор switch**  Написати програму, яка в залежності від знаку логічної операції (<,>,<=,==,!=) виводить на екран її назву (Менше,Більше,...)  **3. Оператор циклу**  Дано натуральне число n і дійсне число x. Обчислити: |
| 20 | **1. Оператор if**  Відомі рік і місяць народження людини. Визначити вік людини на вказану дату (сьогодняшній день). У разі співпадіння номерів місяців вважати, що пройшов повний рік.  **2. Оператор switch**  Написати програму, яка в залежності від порядкового номера кольору у спектрі (1,2,...7) виводить його назву (червоний, помаранчовий, жовтий,.зелений, блакитний, синій, фіолетовий) і код RGB  **3. Оператор циклу**  Дано натуральні числа n і m (50<m<n). Обчислити:  де |

**Література**

1. Лекція 3

2.C# 2005 и платформа .Net 3.0 для профессионалов. Нейгел К., Ивьен Б. и др. –М.: ООО “И.Д. Вильямс", 2008. –1376 с.

3. Д.М. Златопольский. Сборник задач по программированию. 2-е издание. // СПб.: БХВ-Петербург, 2007.-240 с.

**Лабораторна робота № 4**

**Масиви і стандартні класи C#**

**Мета роботи:**

1. Робота з масивами в C#

2. Оператор foreach

3. Генерація випадкових чисел

4. Робота з математичними функціями

5. Лінійний пошук

6. Двійковий пошук

**Методичні рекомендації до виконання лабораторної роботи**

1. Прочитайте теоретичну частину л/р. Спробуйте відповісти на наведені в кінці запитання

2. Прочитайте цю лабораторну роботу та виконайте наведені в ній приклади (вони всі працездатні)

3. При виконанні завдань зверніть увагу на приклад застосування до масивів циклу **foreach**

4. Також зверніть увагу на опис і реалізацію методу бісекцій (ділення навпіл) знаходження коренів нелінійних рівнянь. Те також задача на одновимірні масиви і цикли

5. Для поглибленого вивчення цього матеріалу прочитайте розділ 5 книги [1].

**Порядок виконання роботи**

1. Створити директорію Lab4, в якій будуть розміщуватися проекти цієї лабораторної роботи.

2. Виконати завдання свого варіанту у вигляді одного консольного проекту

3. Зафіксувати результати у Контрольній роботі 1.

**Приклади виконання завдань**

**1. Робота з масивами в C#**

Розглянемо приклад програми з лекції 5, яка визначає суму і кількість від'ємних елементів, а також максимальний елемент масиву, що складається з 6 цілочисельних елементів.

Тут елементи масиву **а** ініціалізуються при створенні масиву. Далі в циклі з параметром елементи масиву виводяться на консоль. Зверніть увагу, що для виводу використовується метод Console.Write а не Console.WriteLine, тому всі елементи виводяться в один рядок. Символ табуляциії "\t" в методі Console.Write розділяє елементи при виводі.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Console\_Lab4

{

class Program

{

static void Main(string[] args)

{

const int n = 6;

int[] a = new int[n] { 3, 12, 5, -9, 8, -4 };

Console.WriteLine( "Початковий масив:" );

for ( int i = 0; i < n; ++i )

Console.Write( "\t" + a[i] );

Console.WriteLine();

long sum = 0; // сума від'ємних елементів

int num = 0; // кількість від'ємних елементів

for ( int i = 0; i < n; ++i )

if ( a[i] < 0 )

{

sum += a[i];

++num;

}

Console.WriteLine( "Сума від'ємних = " + sum );

Console.WriteLine( "Кількість від'ємних = " + num );

int max = a[0]; // максимальний елемент

for ( int i = 1; i < n; ++i )

if ( a[i] > max ) max = a[i];

Console.WriteLine( "Максимальний елемент = " + max );

Console.ReadKey();

}

}

}

Розглянемо ще одну "класичну" задачу – генерація чисел Фібоначчі: послідовності чисел, яка задовольняє умовам

F1 = 1; F2 = 1; Fk = Fk-1 + Fk-2 для k>2.

Цю задачу можна реалізувати з використанням масивів. В наступному прикладі кількість чисел вводиться з консолі, потім створюється одновимірний масив вказаної розмірності, який заповнюється згенерованими числами.

**Генерація чисел Фібоначчі**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Console\_Lab4

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Введіть кількість чисел послідовності");

int n = int.Parse(Console.ReadLine());

int[] fibonachi = new int[n];

fibonachi[0] = 1;

fibonachi[1] = 1;

for (int i = 2; i < n; i++)

{

fibonachi[i] = fibonachi[i - 2] + fibonachi[i - 1];

}

for (int i = 0; i < n; i++)

Console.WriteLine("fibonachi[" + i + "]=" + fibonachi[i]);

Console.ReadKey();

}

}

}

**Генерація простих чисел**

**Алгоритм Ератосфена**

Простим є число, яке ділиться тільки на 1 і на себе.

Алгоритм пошуку послідовності простих чисел запропонував древньогрецький математик Ератосфен, і він отримав назву **Решето Ератосфена.**

**Ідея і загальний опис алгоритму.**

Є розташована в ряд за збільшенням послідовність цілих чисел. Спочатку в ній викреслюються усі числа кратні 2, окрім її самої, і так до N. Далі із списку, що вийшов, береться число, що йде за двійкою, - трійка, викреслюються усі кратні 3 числа, окрім її самої. У такому вигляді алгоритм триває для частини послідовності, що залишилася, і у результаті отримаємо усі прості числа у вказаному діапазоні.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 |
| 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 |
| 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 |
| 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 100 |

У таблиці наведені усі цілі числа від 2 до 100. Червоним помічені ті, які були видалені в процесі виконання алгоритму Решето Ератосфену.

Тепер розглянемо алгоритм детальніше, розбивши його на декілька частин. Отже, для пошуку простих чисел методом Решета Ератосфену треба:

1. Організувати список з чисел від 2 до N, а також логічний масив розмірністю N;
2. У вільну змінну R записати число 2;
3. Виключити усі числа кратні R, починаючи з R\*2;
4. Записати в R наступне за R не закреслене число;
5. Повторювати дії, описані в двох попередніх кроках.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace SimpleNumber

{

class Program

{

//Нахождение простых чисел. Алгоритм "Решето Эратосфена"

//Запишем подряд все числа от 2 до N.

//Дальше вычеркнем из этого списка все числа кратные 2,

// исключая саму двойку, потом вычеркнем все числа кратные 3,

//исключая само число 3, число 4 уже вычеркнуто, вычеркиваем числа кратные 5 и т.д.

//Продолжаем этот процесс, пока квадрат очередного числа не превысит N.

static void Main(string[] args)

{

bool[] table = new bool[100];

int i, j;

// Отмечаем все числа как простые

for (i = 0; i < table.Length; i++)

table[i] = true;

// Вычеркиваем лишнее

for (i = 2; i \* i < table.Length; i++)

if (table[i])

for (j = 2 \* i; j < table.Length; j += i)

table[j] = false;

// Выводим найденное

for (i = 2; i < table.Length; i++)

{

if (table[i])

Console.WriteLine(i);

}

Console.ReadKey();

}

}

}

**2. Оператор foreach**

Оператор **foreach** використовується для перебору елементів в масивах та інших колекціях і є новим типом циклу, реалізованому в C#. В цьому прикладі створюється *ступінчастий масив*, який заповнюється даними. За допомогою вкладених операторів циклу елементи масиву виводяться на консоль у вигляді матриці.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Console\_Lab4

{

class Program

{

static void Main(string[] args)

{

//оператор foreach і ступінчасті масиви

int[][] а = new int[3][];

а[0] = new int[5] { 24, 50, 18, 3, 16 };

а[1] = new int[3] { 7, 9, -1 };

а[2] = new int[4] { 6, 15, 3, 1 };

Console.WriteLine("Початковий масив:");

foreach (int[] mas1 in а)

{

foreach (int x in mas1)

Console.Write("\t" + x);

Console.WriteLine();

}

Console.ReadKey();

}

}

}

**3. Генерація випадкових чисел**

Потреба в генерації випадкових чисел часто виникає в програмуванні. В C# є *нестатичний* клас Random, методи якого дозволяють генерувати різні послідовності випадкових чисел. Нижче наведено приклад з лекції 4 в якому в методі ValsGenerator створюється об'єкт **aRand**. Далі в циклі викликається метод **aRand.Next(100),** який при кожному виклику генерує одне випадкове число від 1 до100 і призначає його черговому елементу масиву. Зверніть увагу, що метод **ValsGenerator** є статичним.

В методі **Main** створюється масив **Data,** викликається метод **ValsGenerator** для його заповнення, потім викликається метод Array.Sort(Data) для його сортування. Після чого елементи масиву виводяться в циклі на консоль.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Console\_Lab4

{

class Program

{

// генератор даних

static void ValsGenerator(int[] Vals)

{

// Random - клас для генерації випадкових чисел

Random aRand = new Random();

// заповнення масиву

for (int i = 0; i < Vals.Length; i++)

Vals[i] = aRand.Next(100);

}

static void Main(string[] args)

{

const int N = 10;

int[] Data = new int[N];

ValsGenerator(Data);

Array.Sort(Data);

Console.WriteLine("Друк відсортованих даних");

for (int i = 0; i < Data.Length; i++)

Console.WriteLine("Data[" + i + "] = " + Data[i]);

Console.ReadLine();

}

}

}

Зверніть увагу, що при виводі масиву

Console.WriteLine("Data[" + i + "] = " + Data[i]);

виконується неявне приведення з типу int до типу string.

**4. Робота з математичними функціями**

Клас Math містить методи для роботи з математичними функціями.

Приклад 4.9. з лекції 5 демонструє використання методів обчислення математичних функцій, які містяться в класі Math.

Розглянемо ще один приклад – використання масивів і математичних функцій в задачах обчислювальної математики.

**Постановка задачі**

Знайти дійсні корені рівняння 6x4-3x3+8x2-5=0 за методом бісекції (ділення навпіл) на відрізках [0, 1], [-1, 0].

**Алгоритм методу:**

Нехай [а,b] відрізок, на якому шукаються корені. Припустимо, що функція f(x) неперервна на [а,b] і на кінцях приймає значення різних знаків .



*Алгоритм* методу полягає в побудові послідовності вкладених відрізків, на кінцях яких функція приймає значення різних знаків. Кожний наступний відрізок отримують діленням навпіл попереднього. Опишемо один крок ітераційного методу. Нехай на *к-*ому кроці знайдено відрізок такий, що . Знайдемо середину відрізку . Якщо, то - корінь і задача вирішена. Якщо ні, то з двох половин відрізку вибираємо той, на кінцях якого функція має протилежні знаки:



, , якщо



, , якщо



*Критерій закінчення ітераційного процесу*: якщо довжина відрізку знаходження кореня менше 2, то ітерації припиняють і за значення кореня із заданою точністю приймають середину відрізку.



Код програми може бути таким:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace bicection

{

class Program

{ //Знайти корені нелінійного рівняння

//6x4-3x3+8x2-5=0

//x1=0.74213

//x2=-0.6365

static double f(double x)

{

//рівняння, для якого шукаємо корені

double y = 6 \* Math.Pow(x, 4) - 3 \* Math.Pow(x, 3) + 8 \* Math.Pow(x, 2) - 5;

return y;

}

// --------------------------------------------------------------------

static double bicect(double left, double right)

{

//метод бісекцій

double eps = 0.00001;

double center = 0;

while (right - left > eps \* 2)

{

center = (right - left) / 2 + left;

if (f(center) \* f(left) > 0)

left = center;

else

right = center;

}

return center;

}

// ----------------------------------------------------

static void Main(string[] args)

{

//метод бісекцій

double x1 = bicect(0, 1);

double x2 = bicect(-1, 0);

Console.WriteLine("Метод бісекцій");

Console.WriteLine("x1={0},x2= {1}", x1, x2);

Console.ReadKey();

}

}

}

В цій програмі є два методи: static double f(double x), в якому визначається рівняння, і метод static double bicect(double left, double right). Для обчислення ступеня в методі **f** використовується метод Math.Pow( ).

Другий метод static double bicect(double left, double right) реалізує ітераційний процес. В методі Main два рази викликається метод бісекцій для знаходження коренів на вказаних відрізках. Результати виводляться на консоль.

**5. Лінійний пошук в масиві**

**Лінійний, послідовний пошук** — алгоритм знаходження заданого значення довільної функції на деякому відрізку.

**Формальний запис алгоритму:**

1. Визначити елемент, який шукаємо ***key;***

2. Встановити границі області пошуку L, R для масиву A [a];

3. Якщо аi = key – елемент знайдено;

4. Інакше – переходимо до наступного елементу.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Search

{

class Program

{

static void Main(string[] args)

{

int i, n, key, nom;

bool x = false;

Console.WriteLine("Розмір масиву");

n = int.Parse(Console.ReadLine());

int[] mas = new int[n]; //визначення масиву

Console.WriteLine("Шуканий елемент");

key = int.Parse(Console.ReadLine());

// Random - клас для генерації випадкових чисел

Random aRand = new Random();

for (i = 0; i < n; i++)

{

//формування масиву, заповнення його випадковими числами

mas[i] = aRand.Next(n);

Console.Write(mas[i] + " ");

}

for (i = 0; i < n; i++)

{ //якщо цей елемент дорівнює шуканому

if (mas[i] == key) //то x призначаємо true

{ x = true; nom = i; break; } //і виходимо з циклу

}

if (x == true)

Console.WriteLine("Елемент знайдено");

else Console.WriteLine("\nЕлемент не знайдено");

Console.ReadKey();

}

}

}

**6. Двійковий (бінарний) пошук**

**Двійковий (бінарний) пошук** (також відомий як метод ділення навпіл і дихотомія) — класичний алгоритм пошуку елемента у відсортованому масиві.

**Формальний запис алгоритму:**

1. Масив ділитися на дві рівні частини, шляхом визначення першого (a), останнього (b) і середнього (c) елементів;
2. Середній елемент порівнюється з шуканим (s):

якщо **s<c.** Останньому елементу призначається значення середнього, тим самим ділянка пошуку зменшується удвічі: b=c;

якщо **s>c**. Першому елементу призначається значення середнього, і ділянка пошуку зменшується удвічі: a=c;

якщо **s=c**. Елемент знайдений, і робота алгоритму завершується**.**

1. Якщо для перевірки не залишився жодного елементу, то алгоритм завершується, інакше виконується перехід до пункту 2.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace BinarySearch

{

class Program

{

static void Main(string[] args)

{

int i, n, key, begin, end, c;

bool x = false;

Console.WriteLine("Розмір масиву");

n = int.Parse(Console.ReadLine());

int[] mas = new int[n]; //визначення масиву

Console.WriteLine("Шуканий елемент");

key = int.Parse(Console.ReadLine());

for (i = 0; i < n; i++)

{

//формування масиву, заповнення його числами кратними n (10,20,....)

mas[i] = n \* i;

Console.Write(mas[i] + " ");

}

begin = 0; end = n; //ліва і права границі масиву

while (begin < end)

{

// власне реалізація алгоритму

c = begin + (end - begin) / 2;

if (key < mas[c]) end = c;

else if (key > mas[c]) begin = c + 1;

else { x = true; break; }

}

if (x == true)

Console.WriteLine("Елемент знайдено");

else Console.WriteLine("\nЕлемент не знайдено");

Console.ReadKey();

}

}

}

**7. Варіанти завдань для Лабораторної роботи № 4**

**Номер варіанта відповідає номеру прізвища студента у списку групи**

Кожний студент повинен виконати по чотири завдання свого варіанту

|  |  |
| --- | --- |
| **№ варі-анту** | **Зміст завдання** |
| 1 | 1. Реалізувати функцію знаходження простих чисел у вказаному діапазоні. Кількість чисел ввести з консолі. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Створити масив для зберігання значень зросту студентів групи (20 чоловік). Заповнити масив за допомогою класу Random цілими числами в діапазоні від 160 до 190 включно. Відсортувати масив і вивести на консоль його елементи.  3. Є прямокутний масив розмірністю (5x5). Визначити суму всіх елементів третього рядка і суму всіх елементів першого стовпчика.  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння  6x4-3x3+8x2-25=0  5. Створити масив з трьох цілих чисел і ініцілізувати при створенні. Визначити мінімальне і максимальне число. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 2 | 1. Реалізувати функцію знаходження простих чисел у вказаному діапазоні. Кількість чисел ввести з консолі. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Згенерувати масив М випадкових чисел в діапазоні від 1 до 50. Обчислити суму елементів масиву, середнє арифметичне (мат.очікування) елементів M, квадратний корінь від М, min і max елементів масиву. Результати вивести на консоль.  3. В двовимірному масиві зберігається інформація про зарплату 18 співробітників за 12 місяців (за січень – першому стовпчику, лютий – другому, ...). Визначити загальний бюджет зарплати за рік, загальну і середню зарплату за квітень.  4. Реалізувати консольній застосунок знаходження методом Ньютона коренів нелінійного рівняння  (x2-6\*x)2 -2(x-3)2=81  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 3 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні 21 до 97. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Згенерувати масив випадкових чисел в діапазоні від 10 до 100. Обчислити добуток елементів масиву, середнє арифметичне (мат.очікування) елементів M, min і max елементів масиву. Результати вивести на консоль.  3. В двовимірному масиві зберігається інформація про зарплату 18 співробітників за 12 місяців (за січень – першому стовпчику, лютий – другому, ...). Визначити загальну зарплату кожного співробітника за рік і середню зарплату за кожний місяць.  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння  (x2+2x)2-(x+1)2=55  5. Створити масив з трьох цілих чисел і ініцілізувати при створенні. Визначити мінімальне і максимальне число. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 4 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 101 до 199. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Згенерувати масив випадкових дійсних чисел в діапазоні від 0 до 1. Відсортувати масив, знайти min і max елементи. Результати вивести на консоль.  3. В двовимірному масиві зберігається інформація про зарплату 30 працівників фірми за 12 місяців (за січень – першому стовпчику, лютий – другому, ...). Визначити середню зарплату фірми за кожний місяць року, а також середню зарплату за кожний місяць кожного співробітника.  4. Реалізувати консольній застосунок знаходження коренів нелінійного рівняння методом бісекції  (x2+x+1)( x2+x+1) -12=0  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 5 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 101 до 199. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Згенерувати дві серії випадкових чисел, що повторюються. Кількість чисел ввести з консолі. Результати вивести на консоль.  3. В двовимірному масиві зберігається інформація про оцінки студентів з двох дисциплін (з дисципліни ОС – в першому стовпчику, з дисципліни Мова С# - в другому). Визначити середню успішність студентів групи і середній бал (рейтинг) кожного студента з цих дисциплін.  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння  (x2-5\*x+7) 2- (x-2)(x-3)=0  5. Створити масив з трьох цілих чисел і ініцілізувати при створенні. Визначити мінімальне і максимальне число. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 6 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 1 до 99. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Згенерувати масив чисел Фібоначчі. Кількість чисел ввести з консолі. Результати вивести на консоль.  3. Фірма має 10 магазинів. Інформація про прибуток кожного магазину за кожний місяць зберігається в двовимірному масиві (першрго – в першому рядку, другого – в другому, і т.д.). Обчислити середній прибуток кожного магазину.  4. Реалізувати консольній застосунок знаходження методом Ньютона коренів нелінійного рівняння  x4-13x2+36=0  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 7 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 11 до 51. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Створити масив для зберігання значень ваги студентів групи (20 чоловік). Заповнити масив за допомогою класу Random цілими числами в діапазоні від 50 до 100 включно. Відсортувати масив і вивести на консоль його елементи.  3. В двовимірному масиві зберігається інформація про оцінки кожного із 20 студентів групи по кожній з 10 дисциплін. Перший рядок – інформація про оцінки першого студента, друга – другого і т.д. Обчислити середню успішність кожного студента.  4. Реалізувати консольній застосунок знаходження коренів нелінійного рівняння методом бісекції  2x8+x4-15=0  5. Створити масив з чотирьох дійсних чисел і ініцілізувати при створенні. Визначити мінімальне і максимальне число. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 8 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 1 до 101. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Створити масив з 10 елементів і заповнити його значеннями в коді. Всі його елементи:  - зменшити на 20;  - помножити на останній елемент  - збільшити на число В, яке ввести з консолі  3. В двовимірному масиві зберігається інформація про оцінки кожного із 20 студентів групи по кожній з 10 дисциплін. Перший рядок – інформація про оцінки першого студента, друга – другого і т.д. Обчислити середню успішність групи по кожній з дисциплін і в цілому успішність групи.  4. Реалізувати консольний застосунок знаходження методом бісекції коренів нелінійного рівняння  x4-13x2+36=0  5. Створити масив з чотирьох дійсних чисел і ініцілізувати при створенні. Визначити мінімальне і максимальне число. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 9 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 11 до 51. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Створити масив з 10 елементів і заповнити його значеннями в коді. Визначити суму всіх елементів масиву, суму квадратів всіх елементів масиву, середнє арифметичне всіх елементів масиву.  3. В двовимірному масиві 3x5 знайти суму елементів в кожному рядку і суму елементів кожного стовпчика.  4. Реалізувати консольній застосунок знаходження коренів нелінійного рівняння методом Ньютона  x4-2x3+x-132=0  5. Створити масив з чотирьох дійсних чисел і ініцілізувати при створенні. Визначити мінімальне і максимальне число. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 10 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 101 до 151. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Створити масив з 10 елементів і заповнити його значеннями в коді. Всі його елементи: збільшити у 2 рази, розділити на перший елемент, зменшити на число А, яке ввести з консолі.  3. Три групи студентів, в кожній з яких 20 чоловік, в сесію здавали по 4 екзамени. Відомості про оцінки кожної групи зберігаються в двовимірних масивах. Визначити кращу групу за середньою оцінкою  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння  8x4+6x3-13x2-x+3=0  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 11 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 1 до 151. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Створити масив з 8 елементів і заповнити його випадковими дійсними числами в діапазоні від 1 до 20. Результати вивести на консоль  3. Створити прямокутний масив і заповнити його випадковими числами від 1 до 40. Знайти суму елементів другого стовпчика    4. Реалізувати консольній застосунок знаходження методом Ньютона коренів нелінійного рівняння  x3+6x+4x2+3=0  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль. |
| 12 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 1 до 131. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена  2. Створити масив з 8 елементів і заповнити його випадковими дійсними числами в діапазоні від 21 до 40. Результати вивести на консоль  3. Створити прямокутний масив і заповнити його випадковими числами від 1 до 40. Знайти суму елементів третього рядка масиву, які менше 20  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння  2x4-x3-9x2+13x-5=0  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль. |
| 13 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 1 до 97. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена  2. Створити масив і заповнити його 20-ма першими цілими числами, які діляться на 13 або на 17 і знаходяться в інтервалі, ліва границя якого рівна 300.  3. Створити прямокутний масив і заповнити його випадковими числами від 1 до 40. Знайти суму елементів першого рядка масиву і кількість його не нульових елементів  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння  (x-1)3+(2x+3)3=27x3+8  5. Створити масив з чотирьох дійсних чисел і ініцілізувати при створенні. Визначити мінімальне і максимальне число. Застосувати методи класу Math. Результати вивести на консоль. |
| 14 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 1 до 97. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена  2. Створити масив і заповнити його 30-ма простими числами (простим є число, яке ділиться на 1 і самого себе)  3. Створити прямокутний масив цілих чисел і  заповнити його випадковими числами від 1 до 100. Визначити суму елементів масиву, які більше 30 і середнє арифметичне всіх елементів  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння  x4-4x3-19x2+106x-120=0  5. Створити масив з чотирьох дійсних чисел і ініцілізувати при створенні. Визначити мінімальне і максимальне число. Застосувати методи класу Math. Результати вивести на консоль. |
| 15 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 11 до 101. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена  2. Створити масив і заповнити його 20-ма випадковими числами в діапазоні 20 до 100. Вивести на консоль елемент за його індексом в масиві.  3. Створити прямокутний масив цілих чисел і заповнити його випадковими числами від 1 до 60. Визначити суму елементів масиву, які більше 10 і середнє арифметичне елементів першого рядка  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння  x4-2x3+x-132=0  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль. |
| 16 | 1. Реалізувати функцію знаходження всіх простих чисел у вказаному діапазоні (ввести з консолі). Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена  2. Створити масив, розмір масиву ввести з консолі, заповнити його випадковими числами від 1 до 50. Відсортувати елементи. Вивести на консоль в прямому і зворотному порядку.  3. Створити прямокутний масив цілих чисел і ініціалізувати його при створенні. Для кожного рядка масиву обчислити суму від'ємних елементів.  4. Реалізувати консольній застосунок знаходження методом Ньютона коренів нелінійного рівняння  6x3-x2-20x+12=0  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль. |
| 17 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 1 до 101. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена  2. Створити масив, розмір масиву ввести з консолі, заповнити його випадковими дійсними числами від 1 до 50. Обчислити корінь квадратний кожного з його елементів і середнє арифметичне двох сусідніх елементів.  3. Створити прямокутний масив цілих чисел і ініціалізувати його при створенні. Для кожного рядка масиву обчислити суму додатних і від'ємних елементів.  4. Реалізувати консольній застосунок знаходження методом Ньютона коренів нелінійного рівняння  x4+1=2(1+x)4  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль. |
| 18 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 1 до 51. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена  2. Створити масив, в якому зберігаються дані про кількість опадів, які випали за кожний день місяця (наприклад, в листопаді). Обчислити загальну і середню кількість опадів за місяць.  3. Створити прямокутний масив цілих чисел і ініціалізувати його при створенні. Для кожного стовпчика масиву обчислити суму додатніх, від'ємних елементів.  4. Реалізувати консольній застосунок знаходження методом Ньютона коренів нелінійного рівняння  x4+1=2(1+x)4  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль. |
| 19 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 1 до 51. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена  2. Створити масив, в якому зберігаються дані про температуру в одному місяці. Обчислити середню температуру в місяці.  3. Створити прямокутний масив цілих чисел і ініціалізувати його при створенні. Для кожного рядка масиву обчислити кількість елементів, які більше 10.  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння.  x4+1=2(1+x)4  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль. |
| 20 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 13 до 151. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена  2. Створити масив, в якому зберігаються дані про кількість опадів, що випали за кожний день місяця (наприклад, в травні). Обчислити в якій половині місяця (першій чи другій) випало більше опадів. Обчислити в якій декаді місяця кількість опадів була найбільшою  3. Створити прямокутний масив цілих чисел і ініціалізувати його при створенні. Знайти максимальний і мінімальний елементи кожного рядка.  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння  x4+1=2(1+x)4  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль. |

**Література**

1. Лекція 4

2.C# 2005 и платформа .Net 3.0 для профессионалов. Нейгел К., Ивьен Б. и др. –М.: ООО “И.Д. Вильямс", 2008. –1376 с.

3. Д.М. Златопольский. Сборник задач по программированию. 2-е издание. // СПб.: БХВ-Петербург, 2007.-240 с.

**Лабораторна робота №5**

**Робота з символами і рядками**

**Мета роботи:**

1. Робота з символами. Клас System.Char

2. Робота з рядками. Клас System.String

3. Робота з рядками. Клас System.StringBuilder

4. Обробка помилок (виключень). Клас Exception

**Методичні рекомендації до виконання лабораторної роботи**

1. Прочитайте теоретичну частину лабораторної роботи. Спробуйте відповісти на наведені в кінці запитання. Виконайте приклади, розглянути в ній.

2. Прочитайте цю лабораторну роботу та виконайте наведені в ній приклади (вони всі працездатні).

3. При виконанні завдань зверніть увагу, що тип даних **char** є вбудованим типом-значенням, а тип string є посилковим (динамічним) типом. Тому при роботі з цими типами необхідне явне приведення типів (з char до string і навпаки).

4. Зверніть також увагу на новий клас для роботи з рядками System.StringBuilder і його відміну від класу System.String.

5. Для поглибленого вивчення цього матеріалу прочитайте розділи 8 і 13 книги [2].

**Порядок виконання роботи**

1. Створити директорію Lab5, в якій будуть розміщуватися проекти цієї лабораторної роботи.

2. Виконати завдання свого варіанту у вигляді окремих консольних проектів, розміщених в одному рішенні Lab5.

**Приклади виконання завдань**

**1. Робота з символами. Клас System.Char**

Символьний тип даних **char** представляє один символ. Це тип відноситься до вбудованих типів даних C# і відповідає стандартному класу Char бібліотеки .NET з простору імен System. У цьому класі визначені статичні методи для роботи з символами. Приклад 5.1. лекції 5 демонструє використання деяких методів цього класу. Приклад 5.3, наведений нижче, демонструє роботу з масивами символів.

**Приклад 5.3**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Lab5

{

class Program

{

static void Main(string[] args)

{

char[] a = { 'm', 'a', 's', 's', 'i', 'v' }; // 1

char[] b = "через тиждень у нас перший кредит".ToCharArray();// 2

PrintArray( "масив а:", a );

int pos = Array.IndexOf( a, 'm' );

a[pos] = 'M';

PrintArray( "Змінений масив а:", a );

PrintArray( "Масив b:", b );

Array.Reverse( b );

PrintArray( "Змінений масив b:", b );

Console.ReadKey();

}

public static void PrintArray( string header, Array a )

{

Console.WriteLine( header );

foreach ( object x in a ) Console.Write( x );

Console.WriteLine( "\n" );

}

}

}

В цьому прикладі створюється два масиви типу char: **a,b**. Зверніть увагу, що в ініціалізаторі масиву b використовуються явне перетворення рядка в масив символів викликом методу ToCharArray().

Далі викликається метод PrintArray( "масив а:", a ). В цьому методі елементи масиву виводяться на консоль. Використовується цикл foreach.

В наступних операторах спочатку знаходиться індекс символу 'm' в масиві a, який замінюється символом M, і знову викликається метод PrintArray( "Змінений масив а:", a );

Після цього викликається PrintArray( "Масив b:", b ); для друку масиву b. Потім викликається метод Array.Reverse( b ), який переставляє всі символу масиву у зворотньому порядку. Знову викликається метод PrintArray( "Змінений масив b:", b ); для виводу обернего масиву b.

**2. Робота з рядками. Клас System.String**

Тип string, призначений для роботи з рядками символів в кодуванні Unicode, є вбудованим посилковим типом C#. Йому відповідає базовий клас System.String бібліотеки .NET. В цьому класі є багато методів для роботи з рядками.

В лекції 5 наведено декілька прикладів для роботи з рядками.

Розглянемо приклад 5.5, в якому з рядка виділяються окремі слова. Ця задача часто зустрічається в програмах обробки тексту.

static void Main(string[] args)

{

string words = "рядок, що містить декілька слів, а також знаки пунктуації: такі як двокрапка і крапка.";

string [] split = words.Split(new Char [] {' ', ',', '.', ':'});

foreach (string s in split)

{

if (s.Trim() != "")

Console.WriteLine(s);

}

Console.ReadKey();

}

В прикладі спочатку створюється рядок **words** і масив **split**, в який записуються виділені слова. Метод **Split** повертає масив рядків з елементами - підрядками основного рядка, між якими знаходяться символи-роздільники. Ці символи ми розмістили в масиві типу Char. Далі в циклі для кожного слова викликається метод Trim(), який видаляє пробільні символи з рядка. В нашому прикладі оператор

if (s.Trim() != "")

перевіряє чи виділене слово не є пробілом. Якщо слово не пробіл, воно виводиться на консоль.

**3. Робота з рядками. Клас System.StringBuilder**

При використанні методів класу String створюється копія рядка. Для роботи безпосередньо зі змістом рядка в пам'яті, в С# є спеціальний клас StringBuilder. Всі зміни, що вносяться до об'єкту цього класу, негайно відображаються в ньому, що ефективніше, ніж робота з копіями рядка.

Методи класу StringBuilder практично дублюють аналогічні методи класу String. Використання цього класу потребує менше ресурсів пам'яті, ніж для класу String.

**4. Обробка помилок (виключень). Клас Exception**

Для обробки програмних помилок (виключень) у мові С# використовується клас Exception.

В цьому прикладі в методі Power(int x, int n)обчислюється xn. Значення x і n вводяться з консолі. Для контролю введених даних використовується блок try... catch.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Lab5

{

class Program

{

static public double Power(int x, int n)

{

double z = 1;

for (int i = 1; n >= i; i++)

{

z = z \* x;

}

return z;

}

static void Main(string[] args)

{

int x;

int n;

try

{

Console.WriteLine("Enter x:");

x = Convert.ToInt32(Console.ReadLine());

if ((x >= 0) & (x <= 999))

{

Console.WriteLine("Enter n:");

n = Convert.ToInt32(Console.ReadLine());

if ((n >= 1) & (n <= 100))

{

Console.WriteLine("The power n of x is {0}",

Power(x, n));

Console.ReadLine();

}

else

{

Console.WriteLine("Error : n must be in [1..100]");

Console.ReadLine();

}

}

else

{

Console.WriteLine("Error : x must be in [0..999]");

Console.ReadLine();

}

}

catch (Exception e)

{

Console.WriteLine("Error : Please enter a numeric argument.");

Console.ReadLine();

}

}

}

}

В кінці лекції 6 наведено комплексний приклад реалізації консольного калькулятора з лексичним аналізом введеного виразу та обробкою можливих помилок.

Виконайте цей приклад (завдання 3), проаналізуйте код застосунку.

**Варіанти завдань для Лабораторної роботи № 5**

**Номер варіанта відповідає номеру прізвища студента у списку групи**

Кожний студент повинен виконати по три завдання свого варіанту

|  |  |
| --- | --- |
| **№ варі-анту** | **Зміст завдання** |
| 1 | 1. Ввести з консолі символ. Перевірити, чи це цифра.  2. Ввести з консолі рядок символів у вигляді речення. Розбити на окремі слова і вивести на консоль. Визначити кількість слів.  3. Ввести з консолі назву футбольного клубу. Визначити кількість символів у ньому. Вивести на консоль 2 рядка:  1)“Цей клуб” <назва клубу> мій улюблений  2) “У слові” <назва клубу> є <кількість символів>  4. Ввести з консолі текст, в якому є цифри. Вибрати з нього цифри і вивести їх на консоль.  5.Реалізувати вдосконалений консольний арифметичний калькулятор. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while . |
| 2 | 1. Ввести з консолі три символи. Перевірити, чи вони утворюють число.  2. Ввести з консолі рядок символів на нижньому регістрі і перетворити всі букви у прописні (на верхньому регістрі). Результат вивести на консоль.  3. Є назви двох країн (можна оголосити в коді чи ввести з консолі). Визначити яка з назв довша. Вивести на консоль ці назви та їх довжини.  4. Створити в коді змінну, в яку записати текст та цифри. Знайти суму цифр та максимальну цифру. Вивести на консоль.    5.Реалізувати вдосконалений консольний арифметичний калькулятор. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while . |
| 3 | 1. Ввести слово. Підрахувати кількість символів в ньому.  2. Напишіть процедуру, що підраховує частоту використання групи символів в заданому тексті. Проведіть дослідження творів двох поетів, підрахувавши частоти використання голосних і приголосних. Для представлення текстів використовуйте клас Char[].  3. Ввести назву міста. Визначити чи довжина назви є парним числом. Привести всі літери назви до верхнього регістру. Вивести результати на консоль.  4. Є текст, на початку якого є пробіли і в якому є цифри. Знайти порядковий номер максимальної цифри, починаючи з першого символа, що не є пробілом.  5.Реалізувати вдосконалений консольний арифметичний калькулятор. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while |
| 4 | 1. Ввести текст, в якому є три числа (20,40,10), розділені пробілами. Знайти суму.  2. Напишіть процедуру, що розділяє вихідний текст на слова. Для представлення текстів використовуйте клас StringBuilder.  3. Є 2 прізвища. Визначити яке з них довше. Вивести на консоль ці призвища, їх довжини та повідомлення про те, яке з них довше.  4. Є текст з цифрами. Знайти найбільше з цифр.  5. Реалізувати вдосконалений консольний арифметичний калькулятор. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while . |
| 5 | 1. Ввести текст з цифр і літер. Знайти найбільшу кількість цифр, які йдуть підряд.  2. Напишіть процедуру, що розділяє текст на речення. Для представлення текстів використовуйте клас string.  3. Є назви трьох міст. Вивести на екран найдовшу і найкоротшу назви та кількість символів у них.  4. Створити в коді два речення. Для кожного слова першого речення перевірити чи входить воно в друге речення.  5.Реалізувати вдосконалений консольний арифметичний калькулятор. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while . |
| 6 | 1. Є слово з помилкою **роцесорп**. Переставити літери для отримання слова **процесор.**  2. Вихідний текст є реченням. Напишіть процедуру, що розділяє вихідний текст на слова. Для представлення текстів використовуйте клас Char[].  3. Ввести слово. Визначити чи однакові у ньому другий і четвертий символи. Результат вивести на консоль.  4. Є два речення. Вивести на консоль слова, які є в обох реченнях.  5.Реалізувати вдосконалений консольний арифметичний калькулятор. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while . |
| 7 | 1. Є слово з помилкою **Ітернетн.** Виправити помилку шляхом перестановки символів  .  2. Вихідний текст є реченням. Напишіть процедуру, що розділяє текст на слова. Для представлення текстів використовуйте клас string.  3. Ввести назву футбольного клубу. Вивести на консоль цю назву у стовпчик.  4. Є рядок “Мова С#”. Вставити в середину слово “програмуваня”.  5.Реалізувати вдосконалений консольний арифметичний калькулятор. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while . |
| 8 | 1. Є два слова. Вивести на консоль тільки ті літери слів, які є в обох словах.  2. Напишіть метод IsIder, яка перевіряє чи є вихідний текст правильно побудованим ідентифікатором. Для представлення тексту використовуйте клас Char[].  3. Ввести назву футбольного клубу. Вивести на консоль цю назву у стовпчик.  4. Є рядок “Мова програмуваня С#”. Видолити слово програмуваня.  5.Реалізувати вдосконалений консольний арифметичний калькулятор. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while . |
| 9 | 1. Перевірити чи є число 2002 паліндромом  2. Напишіть процедуру IsIder, яка перевіряє чи є вихідний текст правильно побудованим ідентифікатором. Для представлення тексту використовуйте клас string.  3. Є рядок “Прийшла золота осінь”. Виділіть слово “золота” і приведіть його до верхнього регістру.  4. Є рядок “Мова програмування С#”. Визначити чи входить в нього слово C#.  5. Реалізувати вдосконалений консольний арифметичний калькулятор. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while . |
| 10 | 1. Ввести речення з трьох слів. Обчислити кількість літер в кожному слові.  2. Напишіть процедуру IsInt, яка перевіряє чи є вихідний текст правильно побудованим цілим числом. Для представлення тексту використовуйте клас Char[].  3. Ввести з консолі рядок символів у вигляді речення. Розбити на окремі слова і вивести на консоль. Визначити кількість слів.  4. Ввести з консолі назву футбольного клубу. Визначити кількість символів у ньому. Вивести на консоль 2 рядка:  1)“Цей клуб” <назва клубу> мій улюблений  2) “У слові” <назва клубу> є <кількість символів>  5.Реалізувати вдосконалений консольний арифметичний калькулятор. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while. |
| 11 | 1. Ввести речення. Вивести всі його різні слова.  2. Напишіть процедуру IsInt, яка перевіряє чи є вихідний текст правильно побудованим цілим числом. Для представлення тексту використовуйте клас string.  3.Реалізувати вдосконалений консольний арифметичний калькулятор. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while . |
| 12 | 1. Ввести речення. Вивести на консоль найдовше слово в ньому.  2. Напишіть процедуру IsFloat, яка перевіряє чи є вихідний текст правильно побудованим числом з плаваючою крапкою. Для представлення тексту використовуйте клас Char[].  3.Реалізувати вдосконалений консольний арифметичний калькулятор. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while . |
| 13 | 1. Ввести три слова окремими рядками. Об'єднати слова в речення. Використати клас StringBuilder.  2. Напишіть процедуру IsFloat, яка перевіряє чи є вихідний текст правильно побудованим числом з плаваючою крапкою. Для представлення тексту використовуйте клас string.  3.Реалізувати вдосконалений консольний арифметичний калькулятор. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while. |
| 14 | 1. Ввести назву футбольного клубу. Підрахувати кількість символів в ньому. Вивести на консоль назву клубу у стовпчик (по 1 символу в рядку).  2. Напишіть процедуру IsFloat, яка перевіряє чи є вихідний текст правильно побудованим числом з плаваючою крапкою. Для представлення тексту використовуйте клас StringBuffer.  3.Реалізувати вдосконалений консольний арифметичний калькулятор. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while. |
| 15 | 1. Ввести з консолі речення. Підрахувати кількість пробілів в ньому.  2. Напишіть процедуру IsNumber, перевіряючу чи є вихідний текст правильно побудованим числом. Для представлення тексту використовуйте клас Char[].  3.Реалізувати вдосконалений консольний арифметичний калькулятор. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while. |
| 16 | 1. Ввести два прізвища. Визначити яке з них довше. Вивести на консоль довше прізвище.  2.Реалізувати вдосконалений консольний арифметичний калькулятор.  3. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while. |
| 17 | 1. Є назви трьох міст. Вивести на консоль найдовше і найкоротше з них.    2. Напишіть процедуру «Строгий Паліндром», що визначає чи є заданий текст паліндромом. Нагадаю, палиндромом називається симетричний текст, що однаково читається як зліва направо, так і справа наліво.  3. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while. |
| 18 | 1. З слова **інформатика** шляхом перетворень виділити слова **форма** і **мат.**  2. Напишіть процедуру «Паліндром», що визначає чи є заданий текст паліндромом. При аналізі текста: - пропуски не враховуються; - регістр не враховується; - букви «е» і «е», «и» і «й» вважаються однаковими.  Фраза, яку Мальвіна диктувала Буратіно, - «А роза упала на лапу Азора» вважається палиндромом.  3.Реалізувати вдосконалений консольний арифметичний калькулятор. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while. |
| 19 | 1. Шляхом перестановки символів виправити помилку у слові **заменек**, щоб отримати **екзамен.**  2. Напишіть процедуру Reverse, що міняє порядок елементів масиву.  3.Реалізувати вдосконалений консольний арифметичний калькулятор. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while. |
| 20 | 1. Шляхом перестановки символів виправити помилку у слові **рограмуванняп**, щоб отримати **програмування.**  2. Напишіть процедуру виділення слова **екзамен** з тексту "На наступному тижні буде екзамен" .  3.Реалізувати вдосконалений консольний арифметичний калькулятор. З консолі ввести повністю весь вираз, наприклад, **23+5**, результат вивести на консоль. При виборі операції використати оператор switch. Виконати розбір виразу, перевірку допустимості вхідних даних. Обробку помилок виконати з використанням блоку try...catch. Завершення роботи програми зробити за запрошенням "Для завершення роботи введіть exit" через цикл do... while. |

**Література**

1. Лекція 6

2.C# 2005 и платформа .Net 3.0 для профессионалов. Нейгел К., Ивьен Б. и др. –М.: ООО “И.Д. Вильямс", 2008. –1376 с.

3. Д.М. Златопольский. Сборник задач по программированию. 2-е издание. // СПб.: БХВ-Петербург, 2007.-240 с.

**Лабораторна робота №6**

**Робота з текстовими файлами**

**Мета роботи:**

1. Вивчення методів введення-виведення в текстовий файл.

**2. Варіанти завдань для Лабораторної роботи № 6.**

**Методичні рекомендації до виконання лабораторної роботи**

1. Прочитайте теоретичну частину лабораторної роботи №6. Спробуйте відповісти на наведені в кінці лекції запитання. Виконайте приклади, розглянути в лекції.

2. Прочитайте цю лабораторну роботу та виконайте наведені в ній приклади (вони всі працездатні).

3. Для поглибленого вивчення цього матеріалу прочитайте розділ 24 книги [2].

**Порядок виконання роботи**

1. Створити директорію Lab6, в якій будуть розміщуватися проекти цієї лабораторної роботи.

2. Виконати завдання свого варіанту у вигляді окремих консольних проектів

3. Зафіксувати результати у Контрольній роботі 1.

**Приклади виконання завдань**

В С# для роботи з файлами і потоками використовуються класи **FileStream, StreamWriter**, **StreamReader, FileInfo** і інші класи.

Для введення і виведення даних не потоками, а рядками призначені класи StringReader, StringWriter.

Розглянемо приклад 7.3лекції 7**.** В цьому прикладі спочатку створюється об'єкт **f** класу StreamWriter, який містить файл для виводу **output.txt**. Якщо не вказати повний шлях до файлу як нашому прикладі, то файл буде створений в директорії, де розміщується exe-файл програми: ...\\bin\\degug\\output.txt

Далі визначаються і ініціалізуються змінні **int i** та **string s**, які записуються у вихідний файл. Метод Close() закриває файл.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.IO;

namespace Lab5

{

class Program

{

static void Main(string[] args)

{

StreamWriter f = new StreamWriter("output.txt"); // 2

int i = 3;

string s = "Мова програмування С# - це C++++ ";

f.WriteLine("i = " + i); // 3

f.WriteLine("s = " + s); // 4

f.Close(); // 5

}

}

}

Наступний приклад 7.4 лекції 6 демонструє введення даних з файлу input.txt, розташованому в каталозі D:\C#.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.IO;

namespace Lab5

{

class Program

{

static void Main(string[] args)

{

StreamReader f = new StreamReader("d:\\C#\\input.txt");

string s = f.ReadLine();

Console.WriteLine("s = " + s);

char c = (char)f.Read();

f.ReadLine();

Console.WriteLine("c = " + c);

string buf;

buf = f.ReadLine();

int i = Convert.ToInt32(buf);

Console.WriteLine(i);

buf = f.ReadLine();

double x = Convert.ToDouble(buf);

Console.WriteLine(x);

buf = f.ReadLine();

double y = double.Parse(buf);

Console.WriteLine(y);

buf = f.ReadLine();

decimal z = decimal.Parse(buf);

Console.WriteLine(z);

f.Close();

}

}

}

В цьому прикладі по черзі читаються рядки з файлу d:\\C#\\input.txt" і виводяться на консоль.

**2. Варіанти завдань для Лабораторної роботи № 6.**

**Номер варіанту відповідає номеру прізвища студента у списку групи**

Кожний студент повинен виконати по три завдання свого варіанту

|  |  |
| --- | --- |
| **№ варі-анту** | **Зміст завдання** |
| 1 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Створити програму, що дозволяє писати і читати дані в текстовий файл. Даними є речення української мови. |
| 2 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Створити програму, що дозволяє писати і читати дані в текстовий файл. Даними є арифметичні вирази. |
| 3 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Створити програму, що дозволяє писати і читати дані в текстовий файл. Даними є вирази мови C#. |
| 4 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Побудуйте Windows-форму, що дозволяє писати і читати дані в текстовий файл. Даними є оператори мови C#. |
| 5 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Побудуйте Windows-форму, що дозволяє писати і читати дані в текстовий файл. Даними є числа типа double. |
| 6 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Розробити консольний застосунок, що дозволяє писати і читати дані в текстовий файл. Дані є записами результатів іспиту – прізвище студента і його оцінка. Ім'я текстового файлу включає назву предмету. |
| 7 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Побудуйте Windows-форму, що дозволяє писати і читати дані в текстовий файл. Дані представляють записи бібліотечного каталогу – прізвище автора, назва книги, видавництво, ідентифікаційний номер ISBN, рік видання, число сторінок, ціна. |
| 8 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Дано текстовий файл, що містить речення. Побудувати Windows-форму, що читає речення і що створює список слів, що входять в речення. |
| 9 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Є файл класу FileStream, що містить речення мови. Розробити консольний застосунок, що створює список речень, які містять вказане слово. |
| 10 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Є файл класу FileStream, що містить записи з результатами іспиту. Побудувати Windows-форму, яка читає файл, вибирає в ньому студентів з оцінками більше 60 балів. |
| 11 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Є файл класу FileStream, що містить записи з результатами іспиту. Побудувати Windows-форму, що створює список студенів з оцінками менше 30 балів. |
| 12 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Побудуйте Windows-форму, що дозволяє писати і читати дані в текстовий файл. Даними є математичні формули. |
| 13 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Побудуйте Windows-форму, що дозволяє писати текстовий файл згенеровані прості числа. |
| 14 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Побудуйте Windows-форму, що дозволяє писати і читати дані в текстовий файл. Даними є згенеровані випадкові числа. |
| 15 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Побудуйте Windows-форму, що дозволяє писати і читати дані в текстовий файл. Даними є згенеровані числа Фібоначчі. |
| 16 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Побудуйте Windows-форму, що дозволяє писати і читати дані в текстовий файл. Даними є числа типа double. |
| 17 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Побудуйте Windows-форму, що дозволяє писати і читати дані в текстовий файл. Дані є записами результатів іспиту – прізвище студента і його оцінка. Ім'я текстового файлу включає назву предмету. |
| 18 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Побудуйте Windows-форму, що дозволяє писати і читати дані в текстовий файл. Дані представляють записи бібліотечного каталогу – прізвище автора, назва книги, видавництво, ідентифікаційний номер ISBN, рік видання, число сторінок, ціна. |
| 19 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Є файл класу FileStream, що містить записи з результатами іспиту. Побудувати Windows-форму, яка читає файл, вибирає в ньому студентів з оцінками більше 46 балів |
| 20 | 1. В консольному арифметичному калькуляторі замість консольного введення-виведення реалізувати роботу з файлом. Вхідний файл містить обчислюваний вираз. Вихідний – обчислюваний вираз, текст "Результат=" і сам результат обчислення.  2. Розробити Windows-форму, в текстові поля якої ввести свої прізвище, ім'я, рік народження. Сформувати буфер з введеними даними. Буфер вивести в файл lab6.txt.  3. Створити файл, в який записати результати іспиту у вигляді:  <Прізвище> <оцінка>. Побудувати Windows-форму, що читає записи файлу і виводить прізвища студентів з оцінками менше 46 балів. |

**Література**

1. Лекція 6

2.C# 2005 и платформа .Net 3.0 для профессионалов. Нейгел К., Ивьен Б. и др. –М.: ООО “И.Д. Вильямс", 2008. –1376 с.

3. Д.М. Златопольский. Сборник задач по программированию. 2-е издание. // СПб.: БХВ-Петербург, 2007.-240 с.

**Лабораторна робота № 7**

**Створення класів в C#**

**Мета роботи**

Навчитися створювати власні класи та використовувати їх в програмах.

**Варіанти завдань для Лабораторної роботи № 7.**

**Методичні рекомендації до виконання лабораторної роботи**

1. Прочитайте теоретичну частину лабораторної роботи. Спробуйте відповісти на наведені в кінці лекції запитання.

2. Прочитайте цю лабораторну роботу та виконайте наведені в ній приклади (вони всі працездатні).

3. При вивченні теми лекції і виконанні завдань зверніть увагу на особливості створення конструкторів класу в C#:

- Конструктор не повертає значення (навіть типу void).

- Конструктор може бути з параметрами і без параметрів.

- В класі можуть бути визначені декілька конструкторів з різними списками параметрів.

- Якщо конструктор відсутній, то він створюється автоматично (конструктор за замовчуванням). Такий конструктор не має параметрів.

- Якщо в класі визначений хоча б один конструктор, конструктор за замовчуванням не створюється.

4. Також зверніть увагу, що для доступу до закритих полів в C# призначені спеціальні методи-властивості get i set. Якщо відсутня частина set, властивість доступна лише для читання (read-only), якщо відсутня частина get, властивість доступна лише для запису (write-only).

5. Для поглибленого вивчення цього матеріалу прочитайте розділи 3,4 книги [2].

**Порядок виконання роботи**

1. Створити директорію Lab7, в якій будуть розміщуватися проекти цієї лабораторної роботи.

2. Виконати завдання свого варіанту у вигляді окремих консольних проектів в одному рішенні.

3. Зафіксувати результати у Контрольній роботі 7.

**Приклади виконання завдань**

**1. Розробка і тестування класу Student**

**Приклад 1.** Розглянемо клас Student, який містить відкриті поля і конструктор з параметри для їх ініціалізації. Клас містить один відкритий метод public void StudentRating(int R), який виводить відповідний текст, в залежності від рейтингу.

В методі Main виконується тестування класу.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Lab7

{

class Student

{

public string Name; //имя

public int Age; // возраст

public string Role; // роль

public string Facultet;

public string Group;

public int Course;

public int Rating;

public Student(string N, int A, string R, string F, string G, int C)

{

//конструктор з параметрами

Name = N;

Age = A;

Role = R;

Facultet = F;

Group = G;

Course = C;

}

public void StudentRating(int R)

{

Rating = R;

if (Rating >= 80)

Console.WriteLine("Привіт відмінникам");

else

if (Rating <= 30)

Console.WriteLine("Треба вчитися краще!");

else

Console.WriteLine("Можна вчитися ще краще!");

}

}

class Program

{

static void Main(string[] args)

{

//дані рейтингу

//ініціалізація полів класу виконується в конструкторі з параметрами

Student newStudent = new Student("Бака", 20, "студент", "КННІ", "K-01", 3);

Console.WriteLine("Хто ви?");

Console.WriteLine("Прізвище = " + newStudent.Name);

Console.WriteLine("Вік= " + newStudent.Age);

Console.WriteLine("Роль= " + newStudent.Role);

Console.WriteLine("Факультет = " + newStudent.Facultet);

Console.WriteLine("група= " + newStudent.Group);

Console.WriteLine("курс= " + newStudent.Course);

Console.WriteLine("Ваш рейтинг?");

string r = Console.ReadLine();

newStudent.Rating = int.Parse(r);

newStudent.StudentRating(newStudent.Rating);

Console.ReadLine();

}

}

}

Зверніть увагу, що опис класу розміщується в просторі імен, а не класі Program. Опис класу можна також розмістити в окремому файлі – модулі класу (рекомендується).

Змінимо клас, зробимо поля Name i Age закритими, а доступ до них реалізуємо через властивості get() і set().

**Приклад 2**

**Використання властивостей get і set замість призначення у конструкторі**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Lab7

{

class Student

{

string Name;

int Age;

public string Role; // роль

public string Facultet;

public string Group;

public int Course;

public int Rating;

public Student(string F, string G, int C)

{

//конструктор з параметрами

Role = R;

Facultet = F;

Group = G;

Course = C;

}

//методи get і set для полів класу

public string name

{

get

{

return Name;

}

set

{

Name = value;

}

}

public int age

{

get

{

return Age;

}

set

{

Age = value;

}

}

public void StudentRating(int R)

{

Rating = R;

if (Rating >= 80)

Console.WriteLine("Привіт відмінникам");

else

if (Rating <= 30)

Console.WriteLine("Треба вчитися краще!");

else

Console.WriteLine("Можна вчитися ще краще!");

}

}

class Program

{

static void Main(string[] args)

{

//дані рейтингу

//ініціалізація полів класу виконується в конструкторі з параметрами

Student newStudent = new Student("КННІ", "K-01", 3);

Console.WriteLine("Хто ви?");

//використовуємо властивість

newStudent.name=Console.ReadLine();

Console.WriteLine("Скільки вам років?");

//використовуємо властивість

newStudent.age = int.Parse(Console.ReadLine());

Console.WriteLine("Прізвище = " + newStudent.name);

Console.WriteLine("Вік= " + newStudent.age);

Console.WriteLine("Роль= " + newStudent.Role);

Console.WriteLine("Факультет = " + newStudent.Facultet);

Console.WriteLine("група= " + newStudent.Group);

Console.WriteLine("курс= " + newStudent.Course);

Console.WriteLine("Ваш рейтинг?");

string r = Console.ReadLine();

//просто призначаємо значення відкритому полю класу

newStudent.Rating = int.Parse(r);

newStudent.StudentRating(newStudent.Rating);

Console.ReadLine();

}

}

}

**2. Розробка і тестування методів класу Account**

Програма є автоматизованою системою обліку банківських відомостей.

На кожного клієнта банку зберігаються наступні відомості:

Прізвище, ім’я, по-батькові;

Дата народження;

Паспортні дані;

Ідентиф.код;

Місце роботи (навчання);

Номери рахунків.

Для кожного клієнта визначимо операції: додати нового клієнта, видалити клієнта, змінити реквізити клієнта.

На кожному рахунку зберігається інформація про поточний баланс.

З кожним рахунком можна виконувати наступні дії:

відкриття, закриття,

внесення грошей, зняття грошей,

перегляд балансу.

Створимо два класи:

Класс Client – описує інформацію про клієнта

Клас Account - банківський рахунок

**Класс Client**

**Варіант 1. Конструктор без параметрів**

public class Client

{

//поля класу

private string Name;

private DateTime BirthDate;

private string Passport;

private string ID;

private string Job;

private string nomAccount;

public Client()

{

//конструктор без параметрів

}

}

Область видимості полів класу відповідно до правил має бути визначена або як **закрита**, або як **захищена**. Доступ же до полів - членів класу має бути організований або за допомогою методів, або за допомогою властивостей класу. Створимо властивості класу Client, які забезпечують читання і запис значень полів класу.

public class Client

{

//поля класу

private string Name;

private DateTime BirthDate;

private string Passport;

private string ID;

private string nomAccount;

public Client()

{

//конструктор без параметрів

}

//методи get і set для полів класу

public string passport

{

get

{

return Passport;

}

set

{

Passport = value;

}

}

public string name

{

get

{

return Name;

}

set

{

Name = value;

}

}

public int age

{

get

{

int a;

a = DateTime.Now.Year - BirthDate.Year;

return a;

}

}

public DateTime birthdate

{

get

{

return BirthDate;

}

set

{

if (DateTime.Now > value)

BirthDate = value;

else

throw new Exception("Введена невірна дата народження");

}

}

public string ID\_kod

{

get {return ID; }

set

{

ID = value;

}

}

public string Nom\_Account

{

get { return nomAccount; }

set

{

nomAccount = value;

}

}

//

}

Як видно з прикладу, властивість складається з методів **set і get**. При цьому властивість повинна містити хоч би один з методів. Метод set дозволяє змінювати значення поля класу, get - отримувати значення. У метод set передається значення параметра за допомогою змінної value. Обидва методи можуть містити довільну кількість операторів, що описують алгоритм виконання дій в процесі читання або запису значення в полі класу. У даному прикладі властивості passport і name дозволяють просто дістати доступ до полів класу, читаючи або встановлюючи значення відповідних змінних. Властивість birthdate також призначена для читання і запису значення змінної - члена класу BirthDate. При цьому при читанні значення (операція get) відбувається просто передача значення змінної BirthDate, при спробі ж запису нового значення в цю змінну відбувається перевірка допустимості встановлюваного значення змінної. В даному випадку перевірка зводиться до порівняння нового значення дати народження з поточною датою. Якщо встановлюване значення дати народження більше або рівно поточній даті, генерується виключення, яке не дозволяє записати нове значення в змінну, - член класу.

Властивість age застосовується для отримання поточного віку клієнта. Вона призначена лише для читання значення із змінної, тому містить лише метод get. При використанні властивості age відбувається обчислення поточного значення віку клієнта в роках шляхом віднімання року народження від поточного значення року.

Використання властивостей аналогічно використанню змінних. У наступному прикладі створюється об'єкт с1 класу Client. Потім поля цього об'єкту заповнюються значеннями з використанням властивостей. Після цього на екран виводяться значення полів, для цього також застосовуються властивості класу:

class Program

{

static void Main(string[] args)

{

//

Client c1 = new Client();

c1.name = "Вася";

c1.passport = "9002";

c1.birthdate = new DateTime(1967, 08, 03);

c1.ID\_kod = "123456789";

c1.Nom\_Account = "8097";

Console.WriteLine("Имя=" + c1.name);

Console.WriteLine("Паспорт=" + c1.passport );

Console.WriteLine("Возраст=" + c1.age);

Console.WriteLine("Иден.код=" + c1.ID\_kod);

Console.WriteLine("Код счета=" + c1.Nom\_Account);

Console.ReadKey();

}

}

**Варіант 2. Конструктор з параметрами**

public Client(string ClientName, string ClientPassport,DateTime ClientBirthDate)

{

//конструктор без параметрів

name = ClientName;

passport = ClientPassport;

birthdate = ClientBirthDate;

}

Видно, що конструктор має три параметри. У тілі конструктора відбувається запис переданих як параметри значень у відповідні поля класу за допомогою властивостей даного класу. У випадку з датою народження це дозволяє не дублювати процедуру перевірки введеної дати, а скористатися алгоритмом, реалізованим у властивості birthdate.

Створимо також метод, що дозволяє змінити значення полів об'єкту класу Client:

public void EditClient(string ClientName, string ClientPassport, DateTime ClientBirthDate)

{

Name = ClientName;

Passport = ClientPassport;

birthdate = ClientBirthDate;

}

Як видно з прикладу, код цього методу практично повністю ідентичний конструктору з параметрами з різницею в імені, а також в типі значення, яке повертається. Звичайно, в даному випадку можна було б обійтися і використанням властивостей для зміни значень полів класу, проте, інколи буває корисно, аби такого роду зміни були реалізовані в рамках одного методу, тим більше, якщо алгоритм змін є нестандартним.

Тепер, з використанням конструктора з параметрами, можна створити і відразу ж ініціалізувати об'єкт класу Client:

Client c1=new Client("Вася","9002",new DateTime(1967,08,03));

Змінений код з викликом редагування полів класу:

static void Main(string[] args)

{

Client c1 = new Client("Вася", "9002", new DateTime(1967, 08, 03));

c1.ID\_kod = "123456789";

c1.Nom\_Account = "8097";

Console.WriteLine("Имя=" + c1.name);

Console.WriteLine("Паспорт=" + c1.passport );

Console.WriteLine("Возраст=" + c1.age);

Console.WriteLine("Иден.код=" + c1.ID\_kod);

Console.WriteLine("Код счета=" + c1.Nom\_Account);

Console.ReadKey();

// редагування полів

c1.EditClient("Иван", "4567", new DateTime(1987, 09, 01));

//Відредаговані поля

Console.WriteLine("Имя=" + c1.name);

Console.WriteLine("Паспорт=" + c1.passport);

Console.WriteLine("Возраст=" + c1.age);

Console.ReadKey();

}

**Клас Account**

Реалізуємо два варіанти класу Account.

**Варіант 1.**

У першому варіанті - класі Account - активно використовуватимемо поля класу. Окрім двох основних полів credit і debit, які зберігають прихід і витрату рахунку, введемо поле balance, яке задає поточний стан рахунку, і два поля, пов'язані з останньою виконуваною операцією. Поле sum зберігатиме суму грошей поточної операції, а поле result - результат виконання операції. Полів в класу багато, і як наслідок, в методах класу аргументів буде небагато.

public class Account

{

//закрытые поля класса

int debit=0, credit=0, balance =0;

int sum =0, result=0;

/// <summary>

/// Зачисление на счет с проверкой

/// </summary>

/// <param name="sum">зачисляемая сумма</param>

public void putMoney(int sum)

{

this.sum = sum;

if (sum >0)

{

credit += sum; balance = credit - debit; result =1;

}

else result = -1;

Mes();

}//putMoney

/// <summary>

/// Снятие со счета с проверкой

/// </summary>

/// <param name="sum"> снимаемая сумма</param>

public void getMoney(int sum)

{

this.sum = sum;

if(sum <= balance)

{

debit += sum; balance = credit - debit; result =2;

}

else result = -2;

Mes();

}//getMoney

/// <summary>

/// Уведомление о выполнении операции

/// </summary>

void Mes()

{

switch (result)

{

case 1:

Console.WriteLine("Операция зачисления денег прошла успешно!");

Console.WriteLine("Cумма={0},Ваш текущий баланс={1}",sum, balance);

break;

case 2:

Console.WriteLine("Операция снятия денег прошла успешно!");

Console.WriteLine("Cумма={0},Ваш текущий баланс={1}", sum,balance);

break;

case -1:

Console.WriteLine("Операция зачисления денег не выполнена!");

Console.WriteLine("Сумма должна быть больше нуля!");

Console.WriteLine("Cумма={0},Ваш текущий баланс={1}", sum,balance);

break;

case -2:

Console.WriteLine("Операция снятия денег не выполнена!");

Console.WriteLine("Сумма должна быть не больше баланса!");

Console.WriteLine("Cумма={0},Ваш текущий баланс={1}", sum,balance);

break;

default:

Console.WriteLine("Неизвестная операция!");

break;

}

Console.ReadLine();

}

Як можна бачити, лише у методів **getMoney** і **putMoney** є один вхідний аргумент. Це той аргумент, який потрібний по суті справи, оскільки лише клієнт може вирішити, яку суму він хоче зняти або покласти на рахунок. Інших аргументів в методів класу немає - вся інформація передається через поля класу. Зменшення числа аргументів приводить до підвищення ефективності роботи з методами, оскільки зникають витрати на передачу фактичних аргументів. Але при цьому ускладнюються операції роботи з вкладом, оскільки потрібно у момент виконання операції оновлювати значення полів класу. Закритий метод Mes викликається після виконання кожної операції, повідомляючи про те, як прошла операція, і інформуючи клієнта про поточний стан його балансу.

**Варіант 2**

Спроектуємо аналогічний клас Account1, який відрізняється лише тим, що у нього буде менше полів. Замість поля balance в класі з'явиться відповідна функція з цим же іменем, замість полів sum і result з'являться аргументи в методів, що забезпечують необхідну передачу інформації. От як виглядає цей клас:

public class Account1

{

//закрытые поля класса

int debit=0, credit=0;

/// <summary>

/// Зачисление на счет с проверкой

/// </summary>

/// <param name="sum">зачисляемая сумма</param>

public void putMoney(int sum)

{

int res =1;

if (sum >0)credit += sum;

else res = -1;

Mes(res,sum);

} //putMoney

/// <summary>

/// Снятие со счета с проверкой

/// </summary>

/// <param name="sum"> снимаемая сумма</param>

public void getMoney(int sum)

{

int res=2;

if(sum <= balance())debit += sum;

else res = -2;

balance();

Mes(res, sum);

}//getMoney

/// <summary>

/// вычисление баланса

/// </summary>

/// <returns>текущий баланс</returns>

int balance()

{

return(credit - debit);

}

/// <summary>

/// Уведомление о выполнении операции

/// </summary>

void Mes(int result, int sum)

{

switch (result)

{

case 1:

Console.WriteLine("Операция зачисления денег прошла успешно!");

Console.WriteLine("Cумма={0},Ваш текущий баланс={1}", sum,balance());

break;

case 2:

Console.WriteLine("Операция снятия денег прошла успешно!");

Console.WriteLine("Cумма={0},Ваш текущий баланс={1}", sum,balance());

break;

case -1:

Console.WriteLine("Операция зачисления денег не выполнена!");

Console.WriteLine("Сумма должна быть больше нуля!");

Console.WriteLine("Cумма={0},Ваш текущий баланс={1}", sum,balance());

break;

case -2:

Console.WriteLine("Операция снятия денег не выполнена!");

Console.WriteLine("Сумма должна быть не больше баланса!");

Console.WriteLine("Cумма={0},Ваш текущий баланс={1}", sum,balance());

break;

default:

Console.WriteLine("Неизвестная операция!");

break;

}

}

} //Account1

Порівнюючи цей клас з класом Account, можна бачити, що число полів скоротилося з п'яти до двох, спростилися основні методи getMoney і putMoney. Але, як плата, в класі з'явився додатковий метод balance(), що багато разів викликається, і в методі Mes тепер з'явилися два аргументи. Який клас кращий? Однозначно сказати не можна, все залежить від контексту, від пріоритетів, заданих при створенні конкретної системи.

Процедура класу Testing, що тестує роботу з класами Account і Account1:

class Program

{

static void TestAccounts()

{

Account myAccount = new Account();

myAccount.putMoney(6000);

myAccount.getMoney(2500);

myAccount.putMoney(1000);

myAccount.getMoney(4000);

myAccount.getMoney(1000);

//Аналогічна робота з класом Account1

Console.WriteLine("Новый класс и новый счет!");

Account1 myAccount1 = new Account1();

myAccount1.putMoney(6000);

myAccount1.getMoney(2500);

myAccount1.putMoney(1000);

myAccount1.getMoney(4000);

myAccount1.getMoney(1000);

Console.WriteLine("Конец работы");

Console.ReadLine();

}//

//

Виклик TestAccounts з методу Main

static void Main(string[] args)

{

TestAccounts();

}

}

}

**Варіанти завдань для Лабораторної роботи № 7**

**Номер варіанта відповідає номеру прізвища студента у списку групи**

|  |  |
| --- | --- |
| **№ варіан-ту** | **Зміст завдання** |
| 1 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Створити метод NewStudent додавання нових студентів і запис інформації про них в файл. Реалізувати метод StudentRating(int R). Використати цикл do while для опитування користувача.  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. .  Перенести методи StudentRating(int R) та NewStudent із завдання 1.  3. Розробити клас **Тварина** з 4 полями і двома методами. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 2 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Створити метод NewStudent додавання нових студентів і запис інформації про них в файл. Реалізувати метод StudentRating(int R). Використати цикл do while для опитування користувача.  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. .  Перенести методи StudentRating(int R) та NewStudent із завдання 1.  3. Розробити клас **Персона** з 4 полями і двома методами. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 3 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Створити метод NewStudent додавання нових студентів і запис інформації про них в файл. Реалізувати метод StudentRating(int R). Використати цикл do while для опитування користувача.  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. .  Перенести методи StudentRating(int R) та NewStudent із завдання 1.  3. Розробити клас **Співробітник** з 4 полями і двома методами. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 4 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Створити метод NewStudent додавання нових студентів і запис інформації про них в файл. Реалізувати метод StudentRating(int R). Використати цикл do while для опитування користувача.  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. .  Перенести методи StudentRating(int R) та NewStudent із завдання 1.  3. Розробити клас **Виріб** з 4 полями і двома методами. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 5 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Створити метод NewStudent додавання нових студентів і запис інформації про них в файл. Реалізувати метод StudentRating(int R). Використати цикл do while для опитування користувача.  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. .  Перенести методи StudentRating(int R) та NewStudent із завдання 1.  3. Розробити клас **Організація** з 4 полями і двома методами. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 6 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Створити метод NewStudent додавання нових студентів і запис інформації про них в файл. Реалізувати метод StudentRating(int R). Використати цикл do while для опитування користувача.  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. .  Перенести методи StudentRating(int R) та NewStudent із завдання 1.  3. Розробити клас **Друкарське Видання** з 4 полями і двома методами. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 7 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Створити метод NewStudent додавання нових студентів і запис інформації про них в файл. Реалізувати метод StudentRating(int R). Використати цикл do while для опитування користувача.  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. .  Перенести методи StudentRating(int R) та NewStudent із завдання 1.  3. Розробити клас **Рослина** з 3 полями і двома методами. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 8 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Створити метод NewStudent додавання нових студентів і запис інформації про них в файл. Реалізувати метод StudentRating(int R). Використати цикл do while для опитування користувача.  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Перенести методи StudentRating(int R) та NewStudent із завдання 1.  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. Ініціалізувати поля класу даними **з полів форми**.  Перенести метод StudentRating(int R) із завдання 1.  3. Розробити клас **Місцевість** з 3 полями і двома методами. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 9 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Створити метод NewStudent додавання нових студентів і запис інформації про них в файл. Реалізувати метод StudentRating(int R). Використати цикл do while для опитування користувача.  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. .  Перенести методи StudentRating(int R) та NewStudent із завдання 1.  3. Розробити клас **Товар** з 3 полями і двома методами. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 10 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Створити метод NewStudent додавання нових студентів і запис інформації про них в файл. Реалізувати метод StudentRating(int R). Використати цикл do while для опитування користувача.  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. .  Перенести методи StudentRating(int R) та NewStudent із завдання 1.  3. Розробити клас **Документ** з 3 полями і двома методами. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 11 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Створити метод NewStudent додавання нових студентів і запис інформації про них в файл. Реалізувати метод StudentRating(int R). Використати цикл do while для опитування користувача.  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. .  Перенести методи StudentRating(int R) та NewStudent із завдання 1.  3. Розробити клас **Транспорт** з 3 полями і двома методами. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 12 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Реалізувати метод StudentRating(int R).  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. Ініціалізувати поля класу даними **з полів форми**.  Перенести метод StudentRating(int R) із завдання 1.  3. Розробити клас **Викладач** з 3 полями і одним методом. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 13 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Реалізувати метод StudentRating(int R).  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. Ініціалізувати поля класу даними **з полів форми**.  Перенести метод StudentRating(int R) із завдання 1.  3. Розробити клас **Рослинний\_продукт**  з 3 полями і одним методом. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 14 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Реалізувати метод StudentRating(int R).  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. Ініціалізувати поля класу даними **з полів форми**.  Перенести метод StudentRating(int R) із завдання 1.  3. Розробити клас **Книга**  з 3 полями і одним методом. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 15 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Реалізувати метод StudentRating(int R).  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. Ініціалізувати поля класу даними **з полів форми**.  Перенести метод StudentRating(int R) із завдання 1.  3. Розробити клас **Меблі**  з 3 полями і одним методом. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 16 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Реалізувати метод StudentRating(int R).  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. Ініціалізувати поля класу даними **з полів форми**.  Перенести метод StudentRating(int R) із завдання 1.  3. Розробити клас **Будівля** з 3 полями і одним методом. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 17 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Реалізувати метод StudentRating(int R).  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. Ініціалізувати поля класу даними **з полів форми**.  Перенести метод StudentRating(int R) із завдання 1.  3. Розробити клас **Повітряний транспорт** з 3 полями і одним методом. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 18 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Реалізувати метод StudentRating(int R).  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. Ініціалізувати поля класу даними **з полів форми**.  Перенести метод StudentRating(int R) із завдання 1.  3. Розробити клас **Програмний продукт (Software)**  з 3 полями і одним методом. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 19 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Реалізувати метод StudentRating(int R).  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. Ініціалізувати поля класу даними **з полів форми**.  Перенести метод StudentRating(int R) із завдання 1.  3. Розробити клас **Комп'ютер (Hardware)**  з 3 полями і одним методом. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |
| 20 | 1. Розробити консольний застосунок для роботи з класом Student  з полями: name, LastName, Adress, Passport, Age, Telephon, Rating.  Всі поля класу зробити закритими, а доступ до них реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. Реалізувати метод StudentRating(int R).  2. Реалізувати Windows форму "Анкета студента". Створити клас Student з полями: name, LastName, Adress, Passport, Age, Telephon, Rating. Ініціалізувати поля класу даними **з полів форми**.  Перенести метод StudentRating(int R) із завдання 1.  3. Розробити клас **Видання**  з 3 полями і одним методом. Доступ до полів реалізувати через get i set. Реалізувати конструктор без параметрів, а ініціалізацію полів виконати через властивості. |

**Література**

1. Лекція 8

2.C# 2005 и платформа .Net 3.0 для профессионалов. Нейгел К., Ивьен Б. и др. –М.: ООО “И.Д. Вильямс", 2008. –1376 с.

**Лабораторна робота №8**

**Класи. Поліморфізм**

**Мета роботи**

1. Використання перевантажених методів і операцій (поліморфізм методів і операцій).

2. Індексатори

**3. Варіанти завдань для Лабораторної роботи № 8.**

**Методичні рекомендації до виконання лабораторної роботи**

1. Уважно прочитайте теоретичну частину лабораторної роботи №8 та цю лабораторну роботу.

2. Реалізуйте та проаналізуйте всі приклади з теоретичної частини.

3. Зверніть увагу, що в C# є **дві** різні реалізації перевантаження методів. В першому випадку реалізації, який і розглядається в цій роботі, в одному класі може бути кілька методів з **однаковими іменами**. При цьому вони повинні мати різні типи і/або кількість параметрів.

В другому випадку перевантаження методів використовується при спадкуванні класів – віртуальні методи (лаб.робота 9).

Крім перевантаження методів в C# можна створювати методи зі змінною кількістю параметрів. При цьому останнім параметром повинен бути масив.

4. Для поглибленого вивчення цього матеріалу прочитайте розділи 3,4 книги [2].

**Порядок виконання роботи**

1. Створити директорію Lab8, в якій будуть розміщуватися проекти цієї лабораторної роботи.

2. Виконати завдання свого варіанту у вигляді окремих консольних проектів.

**Приклади виконання завдань**

**1. Перевантаження методів**

Використання декількох методів з одним і тим самим **іменем**, але різними типами **параметрів** називається *перевантаженням методів.*

Компілятор визначає, який саме метод потрібно викликати за типом фактичних параметрів. Наприклад, нижче наведено декілька реалізацій метода max з лекції 8, який повертає найбільше значення для різних типів і кількості параметрів.

// Повертає найбільше з двох цілих:

int max( int а, int b )

// Повертає найбільше з трьох цілих:

int max( int а, int b, int з )

// Повертає найбільше першого параметра і довжини другого:

int max ( int а, string b )

// Повертає найбільше другого параметра і довжини першого:

int max ( string b, int а )

...

Console.WriteLine( max( 1, 2 ));

Console.WriteLine( max( 1, 2, 3 ));

Console.WriteLine( max( 1, "2" ));

Console.WriteLine( max( "1", 2 ));

Багато методів, які реалізовані в мові C# перевантажені.

**2. Варіанти завдань для Лабораторної роботи № 8**

**Номер варіанта відповідає номеру прізвища студента у списку групи**

|  |  |
| --- | --- |
| **№ варіан-ту** | **Зміст завдання** |
| 1 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Тварина** створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини.  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини. |
| 2 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Персона** створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини.  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини. |
| 3 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Співробітник** створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини.  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини. |
| 4 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Виріб** створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини.  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини. |
| 5 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Організація**  створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини.  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини. |
| 6 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Друкарське Видання** створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини.  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини. |
| 7 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Рослина** створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини .  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини. |
| 8 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Місцевість** створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини .  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини. |
| 9 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Товар** створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини .  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини |
| 10 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Документ** створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини .  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини. |
| 11 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Транспорт** створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини .  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини. |
| 12 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Викладач**  створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини .  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини. |
| 13 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Рослинний\_продукт**  створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини .  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини. |
| 14 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Книга**  створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини .  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини. |
| 15 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Меблі**  створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини .  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини. |
| 16 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Будівля**  створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини .  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини. |
| 17 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Повітряний транспорт** створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини .  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини. |
| 18 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Програмний продукт** створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини .  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини. |
| 19 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Комп'ютер** створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини .  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини. |
| 20 | 1. Реалізувати програму з теоретичної частини (Приклад 9.5). Модифікувати  програму таким чином, щоб прочитати кожний рядок і виконати його розбиття на окремі слова: Прізвище і ініціали, рік народження, оклад*.*  2. В розробленому в лаб. 7 класі **Видання**  створити 2 перевантажені методи.  3.Індексатори. Виконати та продемонструвати приклади теоретичної частини .  4. Індексатори. В клас Student додати масив з 10 елементів – Оцінки за кредит з різних дисциплін. Ініціалізувати масив за допомогою індексатора. Розробити метод розрахунку середнього рейтингу студента.  5. Перевантаження операцій. Виконати і розібратися з прикладами теоретичної частини |

**Література**

1. Лекція 9

2.C# 2005 и платформа .Net 3.0 для профессионалов. Нейгел К., Ивьен Б. и др. –М.: ООО “И.Д. Вильямс", 2008. –1376 с.

3. Шилдт Г. С# 4.0: полное руководство. : Пер. с англ. — М. : ООО "И.Д. Вильямс", 2011. — 1056 с.

**Лабораторна робота №9**

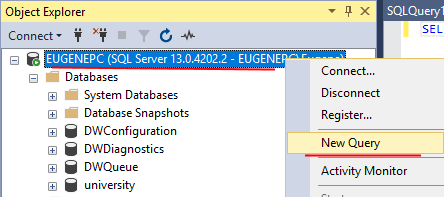
**Основи T-SQL**

План

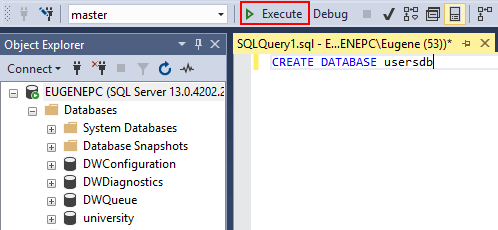
1. Створення бази даних
2. Прикріплення бази даних
3. Видалення бази даних
4. Створення таблиць
5. Створення таблиці в SQL Management Studio
6. Видалення таблиць
7. Перейменування таблиці
8. Типи даних T-SQL
9. Атрибути і обмеження стовпців і таблиць
10. Зовнішні ключі

**Створення бази даних**

Для створення бази даних використовується команда CREATE DATABASE. Щоб створити нову базу даних відкриємо SQL Server Management Studio. Натиснемо на призначення сервера у вікні Object Explorer і в меню виберемо пункт New Query



У центральне поле для введення виразів sql введемо наступний код:



Для виконання команди натиснемо на панелі інструментів на кнопку *Execute* або на клавішу F5. І на сервері з'явиться нова база даних.

Після створення бази даних, ми можемо встановити її в якості поточної за допомогою команди USE:

USE usersdb;

**Прикріплення бази даних**

Можлива ситуація, що у нас вже є файл бази даних, яка, наприклад, створений на іншому комп'ютері. Файл бази даних являє собою файл з розширенням mdf, і цей файл, в принципі, ми можемо переносити. Однак, навіть, якщо ми скопіюємо його на комп'ютер з встановленим MS SQL Server, просто так скопійована база даних на сервері не з'явиться. Для цього необхідно виконати прикріплення бази даних до сервера. У цьому випадку застосовується вираз:

CREATE DATABASE назва\_бази\_даних

ON PRIMARY (FILENAME = 'шлях\_до\_файлу\_mdf\_на\_локальному\_компьютері')

FOR ATTACH;

В якості каталогу для бази даних краще використовувати каталог, де зберігаються інші бази даних сервера. На *Windows 10* за замовчуванням це каталог C: \ Program Files \ Microsoft SQL Server \ MSSQL13.MSSQLSERVER \ MSSQL \ DATA. Наприклад, нехай у нашому випадку, файл з даними називається userstoredb.mdf. І перед нами стоїть задача, цей файл додати на сервер як базу даних. Спочатку його треба скопіювати в вище зазначений каталог. Потім для прикріплення бази до сервера треба використовувати наступну команду:

CREATE DATABASE contactsdb

ON PRIMARY (FILENAME = 'C:\ProgramFiles\MicrosoftSQLServer\MSSQL13.MSSQLSERVER\MSSQL\DATA\userstored.mdf')

FOR ATTACH;

Після виконання команди на сервері з'явиться база даних contactsdb.

**Видалення бази даних**

Для видалення бази даних застосовується команда DROP DATABASE, яка має наступний синтаксис:

DROP DATABASE database\_name1 [, database\_name2] ...

Після команди через кому ми можемо перерахувати всі бази даних які видаляються. Наприклад, видалення бази даних contactsdb:

DROP DATABASE contactsdb

Варто враховувати, що навіть якщо база даних, що видаляється, була прикріплена, то все одно будуть видалені всі файли бази даних.

**Створення та видалення таблиць**

Для створення таблиць застосовується команда CREATE TABLE. З цією командою можна використовувати ряд операторів, які визначають стовпці таблиці і їх атрибути. І крім того, можна використовувати ряд операторів, які визначають властивості таблиці в цілому. Одна база даних може містити до 2 мільярдів таблиць.

Загальний синтаксис створення таблиці виглядає наступним чином:

CREATE TABLE назва\_таблиці

(назва\_стовпця тип\_даних атрибути\_ стовпця1,

назва\_ стовпця 2 тип\_даних атрибути\_ стовпця2,

................................................

назва\_ стовпцяN тип\_даних атрибути\_ стовпцяN,

атрибути\_таблиці)

Після команди CREATE TABLE йде назва таблиця. Ім'я таблиці виконує роль її ідентифікатора в базі даних, тому воно повинно бути унікальним. Ім'я повинно мати довжину не більше 128 символів. Ім'я може складатися з алфавітно-цифрових символів, а також символів $ і знаку підкреслення. Причому першим символом повинна бути буква або знак підкреслення.

Ім'я об'єкта не може включати пробіли і не може бути одним із ключових слів мови Transact-SQL. Якщо ідентифікатор все ж містить пробільні символи, то його слід брати в лапки. Якщо необхідно в якості імені використовувати ключові слова, то ці слова поміщаються в квадратні дужки.

Після імені таблиці в дужках вказуються параметри всіх стовпців і в самому кінці атрибути, які відносяться до всієї таблиці. Атрибути стовпців і атрибути таблиці є необов'язковими компонентами, і їх можна не вказувати.

У самому простому вигляді команда CREATE TABLE повинна містити як мінімум ім'я таблиці, імена і типи стовпців.

Таблиця може містити від 1 до 1024 стовпців. Кожен стовпець повинен мати унікальне в рамках поточної таблиці ім'я, і йому має бути призначено тип даних.

Наприклад, визначення найпростішої таблиці Customers:

CREATE TABLE Customers

(

Id INT,

Age INT,

FirstName NVARCHAR (20),

LastName NVARCHAR (20),

Email VARCHAR (30),

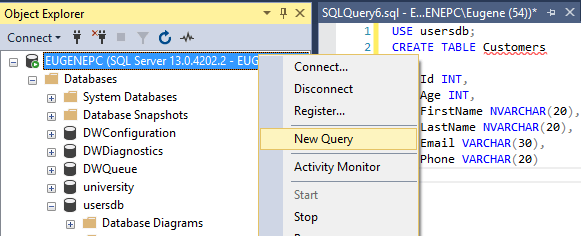
Phone VARCHAR (20)

)

В даному випадку в таблиці Customers визначаються шість стовпців: Id, FirstName, LastName, Age, Email, Phone. Перші два стовпці представляють ідентифікатор клієнта і його вік і мають тип INT, тобто будуть зберігати числові значення. Наступні два стовпці представляють ім'я і прізвище клієнта і мають тип NVARCHAR (20), тобто представляють рядок UNICODE довжиною не більше 20 символів. Останні два стовпці Email і Phone представляють адресу електронної пошти та телефон клієнта і мають тип VARCHAR (30/20) - вони також зберігають рядок, але не в кодуванні UNICODE.

**Створення таблиці в SQL Management Studio**

Створимо просту таблицю на сервері. Для цього відкриємо SQL Server Management Studio та натиснемо правою кнопкою миші на назву сервера. У контекстному меню виберемо пункт New Query.



Таблиця створюється в рамках поточної бази даних. Якщо ми запускаємо вікно редактора SQL як це зроблено вище - з під назви сервера, то база даних за замовчуванням не встановлена. І для її установки необхідно застосувати команду USE, після якої вказується ім'я бази даних. Тому введемо в поле редактора SQL-команд такі вирази:

USE usersdb;

CREATE TABLE Customers

(

Id INT,

Age INT,

FirstName NVARCHAR (20),

LastName NVARCHAR (20),

Email VARCHAR (30),

Phone VARCHAR (20)

);

Тобто в базу даних додається таблиця Customers, яка була розглянута раніше.

Також можна відкрити редактор з під бази даних, також натиснувши на неї правою кнопкою миші і вибравши New Query:

У цьому випадку в якості поточної буде розглядатися та база даних, з під якої був відкритий редактор, і додатково її встановлювати за допомогою команди USE не потрібно.

**Видалення таблиць**

Для видалення таблиць використовується команда DROP TABLE, яка має наступний синтаксис:

DROP TABLE table1 [, table2, ...]

Наприклад, видалення таблиці Customers:

DROP TABLE Customers

**Перейменування таблиці**

Для перейменування таблиць застосовується системна процедура "sp\_rename". Наприклад, перейменування таблиці Users в UserAccounts в базі даних usersdb:

USE usersdb;

EXEC sp\_rename 'Users', 'UserAccounts';

**Типи данних T-SQL**

При створенні таблиці для всіх її стовпців необхідно вказати певний тип даних. Тип даних визначає, які значення можуть зберігатися в стовпці, скільки вони будуть займати місця в пам'яті.

Мова T-SQL надає множину різних типів. Залежно від характеру значень, їх можна розділити на групи.

**Числові типи даних**

BIT: зберігає значення 0 або 1. Фактично є аналогом булевого типу в мовах програмування. Займає 1 байт.

TINYINT: зберігає числа від 0 до 255. Займає 1 байт. Добре підходить для зберігання невеликих чисел.

SMALLINT: зберігає числа від -32 768 до 32 767. Займає 2 байта

INT: зберігає числа від -2 147 483 648 до 2 147 483 647. Займає 4 байта. Найбільш використовуваний тип для зберігання чисел.

BIGINT: зберігає дуже великі числа від -9 223 372 036 854 775 808 до 9 223 372 036 854 775 807, які займають в пам'яті 8 байт.

DECIMAL: зберігає числа з фіксованою точністю. Займає від 5 до 17 байт в залежності від кількості чисел після коми. Даний тип може приймати два параметри precision і scale: DECIMAL (precision, scale). Параметр precision представляє максимальну кількість цифр, які може зберігати число. Це значення повинно знаходитися в діапазоні від 1 до 38. За замовчуванням воно дорівнює 18. Параметр scale представляє максимальну кількість цифр, які може містити число після коми. Це значення повинно знаходитися в діапазоні від 0 до значення параметра precision. За замовчуванням воно дорівнює 0.

NUMERIC: даний тип аналогічний типу DECIMAL.

SMALLMONEY: зберігає дробові значення від -214 748.3648 до 214 748.3647. Призначено для зберігання грошових величин. Займає 4 байти. Еквівалентний типу DECIMAL (10,4).

MONEY: зберігає дробові значення від -922 337 203 685 477.5808 до 922 337 203 685 477.5807. Являє грошові величини і займає 8 байт. Еквівалентний типу DECIMAL (19,4).

FLOAT: зберігає числа від -1.79E + 308 до 1.79E + 308. Займає від 4 до 8 байт в залежності від дробової частини. Може мати форму визначення у вигляді FLOAT (n), де n представляє число біт, які використовуються для зберігання десяткової частини числа (мантиси). За замовчуванням n = 53.

REAL: зберігає числа від -340E + 38 to 3.40E + 38. Займає 4 байти. Еквівалентний типу FLOAT (24).

Приклади числових стовпців:

Salary MONEY,

TotalWeight DECIMAL (9,2),

Age INT,

Surplus FLOAT

**Типи даних, що представляють дату і час**

DATE: зберігає дати від 0001-01-01 (1 січня 0001 року) до 9999-12-31 (31 грудня 9999 року). Займає 3 байта.

TIME: зберігає час в діапазоні від 00: 00: 00.0000000 до 23: 59: 59.9999999. Займає від 3 до 5 байт. Може мати форму TIME (n), де n представляє кількість цифр від 0 до 7 в дробової частини секунд.

DATETIME: зберігає дати і час від 01/01/1753 до 31/12/9999. Займає 8 байт.

DATETIME2: зберігає дати і час в діапазоні від 01/01/0001 00: 00: 00.0000000 до 31/12/9999 23: 59: 59.9999999. Займає від 6 до 8 байт в залежності від точності часу. Може мати форму DATETIME2 (n), де n представляє кількість цифр від 0 до 7 в дробової частини секунд.

SMALLDATETIME: зберігає дати і час в діапазоні від 01/01/1900 до 06/06/2079, тобто найближчі дати. Займає від 4 байта.

DATETIMEOFFSET: зберігає дати і час в діапазоні від 0001-01-01 до 9999-12-31. Зберігає детальну інформацію про час з точністю до 100 наносекунд. Займає 10 байт.

Поширені формати дат:

yyyy-mm-dd - 2017-07-12

dd / mm / yyyy - 12/07/2017

mm-dd-yy - 07-12-17

В такому форматі двозначні числа від 00 до 49 сприймаються як дати в діапазоні 2000-2049. А числа від 50 до 90 як діапазон чисел 1950 - тисячі дев'ятсот дев'яносто дев'ять.

Month dd, yyyy - July 12 2017

Поширені формати часу:

hh: mi - 13:21

hh: mi am / pm - 1:21 pm

hh: mi: ss - 1:21:34

hh: mi: ss: mmm - 1: 21: 34: 12

hh: mi: ss: nnnnnnn - 1: 21: 34: 1234567

**Строкові типи даних**

CHAR: зберігає рядок довжиною від 1 до 8 000 символів. На кожен символ виділяє по 1 байту. Не підходить для багатьох мов, так як зберігає символи не в кодуванні Unicode.

Кількість символів, які може зберігати стовпець, передається в дужках. Наприклад, для стовпця з типом CHAR (10) буде виділено 10 байт. І якщо ми збережемо в стовпці рядок менше 10 символів, то він буде доповненим пробілами.

VARCHAR: зберігає рядок. На кожен символ виділяється 1 байт. Можна вказати конкретну довжину для стовпця - від 1 до 8 000 символів, наприклад, VARCHAR (10). Якщо рядок повинен мати більше 8000 символів, то задається розмір MAX, а на зберігання рядка може виділятися до 2 Гб: VARCHAR (MAX).

Не підходить для багатьох мов, так як зберігає символи не в кодуванні Unicode.

На відміну від типу CHAR якщо в стовпець з типом VARCHAR (10) буде збереженим рядок в 5 символів, то в ньому буде збережено саме п'ять символів.

NCHAR: зберігає рядок в кодуванні Unicode завдовжки від 1 до 4 000 символів. На кожен символ виділяється 2 байти. Наприклад, NCHAR (15)

NVARCHAR: зберігає рядок в кодуванні Unicode. На кожен символ виділяється 2 байта. Можливо задати конкретний розмір від 1 до 4 000 символів:. Якщо рядок повинен мати більше 4000 символів, то задається розмір MAX, а на зберігання рядка може виділятися до 2 Гб.

Ще два типи TEXT і NTEXT є застарілими і тому їх не рекомендується використовувати. Замість них застосовуються VARCHAR і NVARCHAR відповідно.

Приклади визначення строкових стовпців:

Email VARCHAR (30),

Comment NVARCHAR (MAX)

**Бінарні типи даних**

BINARY: зберігає бінарні дані у вигляді послідовності від 1 до 8 000 байт.

VARBINARY: зберігає бінарні дані у вигляді послідовності від 1 до 8 000 байт, або до 231-1 байт при використанні значення MAX (VARBINARY (MAX)).

Ще один бінарний тип - тип IMAGE є застарілим, і замість нього рекомендується застосовувати тип VARBINARY.

**Решта типів даних**

UNIQUEIDENTIFIER: унікальний ідентифікатор GUID (по суті рядок з унікальним значенням), який займає 16 байт.

TIMESTAMP: певна кількість, яке зберігає номер версії рядка в таблиці. Займає 8 байт.

CURSOR: представляє набір рядків.

HIERARCHYID: представляє позицію в ієрархії.

SQL\_VARIANT: може зберігати дані будь-якого іншого типу даних T-SQL.

XML: зберігає документи XML або фрагменти документів XML. Займає в пам'яті до 2 Гб.

TABLE: представляє визначення таблиці.

GEOGRAPHY: зберігає географічні дані, такі як широта і довгота.

GEOMETRY: зберігає координати місцезнаходження на площині.

**Атрибути і обмеження стовпців і таблиць**

При створенні стовпців в T-SQL ми можемо використовувати ряд атрибутів, які є обмеженнями. Розглянемо ці атрибути.

PRIMARY KEY

За допомогою виразу PRIMARY KEY стовпець можна зробити первинним ключем.

CREATE TABLE Customers

(

Id INT PRIMARY KEY,

Age INT,

FirstName NVARCHAR (20),

LastName NVARCHAR (20),

Email VARCHAR (30),

Phone VARCHAR (20)

)

Первинний ключ унікально ідентифікує рядок в таблиці. В якості первинного ключа необов'язково повинні виступати стовпці з типом int, вони можуть представляти будь-який інший тип.

Установка первинного ключа на рівні таблиці:

CREATE TABLE Customers

(

Id INT,

Age INT,

FirstName NVARCHAR (20),

LastName NVARCHAR (20),

Email VARCHAR (30),

Phone VARCHAR (20),

PRIMARY KEY (Id)

)

Первинний ключ може бути складовим (compound key). Такий ключ може знадобитися, якщо у нас відразу два стовпці повинні унікально ідентифікувати рядок в таблиці. наприклад:

CREATE TABLE OrderLines

(

OrderId INT,

ProductId INT,

Quantity INT,

Price MONEY,

PRIMARY KEY (OrderId, ProductId)

)

Тут поля OrderId і ProductId разом виступають як складові первинного ключа. Тобто в таблиці OrderLines не може бути двох рядків, де для обох з цих полів одночасно були б одні і ті ж значення.

IDENTITY

Атрибут IDENTITY дозволяє зробити стовпець ідентифікатором. Цей атрибут може призначатися для стовпців числових типів INT, SMALLINT, BIGINT, TYNIINT, DECIMAL і NUMERIC. При додаванні нових даних в таблицю SQL Server буде інкрементувати на одиницю значення цього стовпця у останнього запису. Як правило, в ролі ідентифікатора виступає той же стовпець, який є первинним ключем, хоча в принципі це необов'язково.

CREATE TABLE Customers

(

Id INT PRIMARY KEY IDENTITY,

Age INT,

FirstName NVARCHAR (20),

LastName NVARCHAR (20),

Email VARCHAR (30),

Phone VARCHAR (20)

)

Також можна використовувати повну форму атрибута:

IDENTITY (seed, increment)

Тут параметр seed вказує на початкове значення, з якого буде починатися відлік. А параметр increment визначає, наскільки буде збільшуватися в такому значенні. За замовчуванням атрибут використовує наступні значення:

IDENTITY (1, 1)

Тобто відлік починається з 1. А наступні значення збільшуються на одиницю. Але ми можемо це поведінка перевизначити. наприклад:

Id INT IDENTITY (2, 3)

В даному випадку відлік почнеться з 2, а значення кожної наступного запису буде збільшуватися на 3. Тобто перший рядок буде мати значення 2, другий - 5, третій - 8 і т.д.

Також слід враховувати, що в таблиці тільки один стовпець повинен мати такий атрибут.

UNIQUE

Якщо ми хочемо, щоб стовпець мав тільки унікальні значення, то для нього можна визначити атрибут UNIQUE.

CREATE TABLE Customers

(

Id INT PRIMARY KEY IDENTITY,

Age INT,

FirstName NVARCHAR (20),

LastName NVARCHAR (20),

Email VARCHAR (30) UNIQUE,

Phone VARCHAR (20) UNIQUE

)

В даному випадку стовпці, які представляють електронну адресу і телефон, матимуть унікальні значення. І ми не зможемо додати в таблицю два рядки, у яких значення для цих стовпців буде збігатися.

Також ми можемо визначити цей атрибут на рівні таблиці:

CREATE TABLE Customers

(

Id INT PRIMARY KEY IDENTITY,

 Age INT,

    FirstName NVARCHAR(20),

    LastName NVARCHAR(20),

    Email VARCHAR(30),

    Phone VARCHAR(20),

    UNIQUE(Email, Phone)

)

NULL і NOT NULL

Щоб вказати, чи може стовпець приймати значення NULL, при визначенні стовпця йому можна задати атрибут NULL або NOT NULL. Якщо цей атрибут явно не буде використаний, то за замовчуванням стовпець буде допускати значення NULL. Винятком є той випадок, коли стовпчик виступає в ролі первинного ключа - в цьому випадку за замовчуванням стовпець має значення NOT NULL.

CREATE TABLE Customers

(

    Id INT PRIMARY KEY IDENTITY,

    Age INT,

    FirstName NVARCHAR(20) NOT NULL,

    LastName NVARCHAR(20) NOT NULL,

    Email VARCHAR(30) UNIQUE,

    Phone VARCHAR(20) UNIQUE

)

DEFAULT

Атрибут DEFAULT визначає значення за замовчуванням для стовпця. Якщо при додаванні даних для стовпця не передбачено значення, то для нього буде використовуватися значення за замовчуванням.

CREATE TABLE Customers

(

Id INT PRIMARY KEY IDENTITY,

Age INT DEFAULT 18,

FirstName NVARCHAR (20) NOT NULL,

LastName NVARCHAR (20) NOT NULL,

Email VARCHAR (30) UNIQUE,

Phone VARCHAR (20) UNIQUE

);

Тут для стовпця Age передбачено значення за замовчуванням 18.

CHECK

Ключове слово CHECK задає обмеження для діапазону значень, які можуть зберігатися в стовпці. Для цього після слова CHECK вказується в дужках умова, якому повинен відповідати стовпець або декілька стовпців. Наприклад, вік клієнтів не може бути менше 0 або більше 100:

CREATE TABLE Customers

(

Id INT PRIMARY KEY IDENTITY,

Age INT DEFAULT 18 CHECK (Age> 0 AND Age <100),

FirstName NVARCHAR (20) NOT NULL,

LastName NVARCHAR (20) NOT NULL,

Email VARCHAR (30) UNIQUE CHECK (Email! = ''),

Phone VARCHAR (20) UNIQUE CHECK (Phone! = '')

);

Тут також вказується, що стовпці Email і Phone не можуть мати порожній рядок як значення (порожній рядок не є еквівалентною значенням NULL).

Для з'єднання умов використовується ключове слово AND. Умови можна задати у вигляді операцій порівняння більше (>), менше (<), не дорівнює (! =).

Також за допомогою CHECK можна створити обмеження в цілому для таблиці:

CREATE TABLE Customers

(

Id INT PRIMARY KEY IDENTITY,

Age INT DEFAULT 18,

FirstName NVARCHAR (20) NOT NULL,

LastName NVARCHAR (20) NOT NULL,

Email VARCHAR (30) UNIQUE,

Phone VARCHAR (20) UNIQUE,

CHECK ((Age> 0 AND Age <100) AND (Email! = '') AND (Phone! = ''))

)

## Зовнішні ключі

Зовнішні ключі застосовуються для установки зв'язку між таблицями. Зовнішній ключ встановлюється для стовпців з залежною, підлеглої таблиці, і вказує на один із стовпців з головної таблиці. Хоча, як правило, зовнішній ключ вказує на первинний ключ з пов'язаної головної таблиці, але це не обов'язково має бути неодмінною умовою. Зовнішній ключ також може вказувати на якийсь інший стовпець, який має унікальне значення.

Загальний синтаксис установки зовнішнього ключа на рівні стовпця:

[FOREIGN KEY] REFERENCES головна\_таблиця (стовпчик\_головної\_таблиці)

[ON DELETE {CASCADE | NO ACTION}]

[ON UPDATE {CASCADE | NO ACTION}]

Для створення обмеження зовнішнього ключа на рівні стовпця після ключового слова REFERENCES вказується ім'я пов'язаної таблиці і в круглих дужках ім'я пов'язаного стовпця, на який буде вказувати зовнішній ключ. Також зазвичай додаються ключові слова FOREIGN KEY, але в принципі їх необов'язково вказувати. Після висловлення REFERENCES йде вираз ON DELETE і ON UPDATE.

Загальний синтаксис установки зовнішнього ключа на рівні таблиці:

FOREIGN KEY (стобец1, столбец2, ... столбецN)

REFERENCES главная\_табліца (столбец\_главной\_табліци1, столбец\_главной\_табліци2, ... столбец\_главной\_табліциN)

[ON DELETE {CASCADE | NO ACTION}]

[ON UPDATE {CASCADE | NO ACTION}]

Наприклад, визначимо дві таблиці і зв'яжемо їх за допомогою зовнішнього ключа:

CREATE TABLE Customers

(

Id INT PRIMARY KEY IDENTITY,

Age INT DEFAULT 18,

FirstName NVARCHAR (20) NOT NULL,

LastName NVARCHAR (20) NOT NULL,

Email VARCHAR (30) UNIQUE,

Phone VARCHAR (20) UNIQUE

);

CREATE TABLE Orders

(

Id INT PRIMARY KEY IDENTITY,

CustomerId INT REFERENCES Customers (Id),

CreatedAt Date

);

Тут визначені таблиці Customers і Orders. Customers є головною і представляє клієнта. Orders є залежною і являє замовлення, зроблене клієнтом. Ця таблиця через стовпець CustomerId пов'язана з таблицею Customers і її стовпцем Id. Тобто стовпець CustomerId є зовнішнім ключем, який вказує на стовпець Id з таблиці Customers.

Визначення зовнішнього ключа на рівні таблиці виглядало б так:

CREATE TABLE Orders

(

Id INT PRIMARY KEY IDENTITY,

CustomerId INT,

CreatedAt Date,

FOREIGN KEY (CustomerId) REFERENCES Customers (Id)

);

**Завдання. (2 бали)**

1. Прочитати лабораторну роботу №2;
2. До бази проекту додати таблиці, задати їх первинні ключі і пов’язати їх.

БД повинна включати усі таблиці на сьогодні.

**Лабораторна робота № 10**

**Мета роботи:**

1. Вивчення спадкування реалізації класів

2. Віртуальні методи

3. Абстрактні класи

**4. Варіанти завдань для Лабораторної роботи № 10.**

**Методичні рекомендації до виконання лабораторної роботи**

1. Уважно прочитайте лекцію теоретичну частину до л/р та цю лабораторну роботу.

2. Виконайте і проаналізуйте всі наведені в приклади.

3. Зверніть увагу, що в С# заборонено множинне спадкування класів, тобто кожний похідний клас може мати тільки одного предка (базовий клас).

4. Якщо в базовому класі є конструктор з параметрами, то для його використання в похідному класі необхідно використовувати конструкцію

...(...):base(...){...}, яка розташовується в оголошенні конструктора похідного класу між заголовком конструктора і тілом.

5. Для поглибленого вивчення цього матеріалу прочитайте розділи 3,4 книги [2] та розділ 11 книги [3].

**Варіанти завдань для Лабораторної роботи № 3**

**Номер варіанта відповідає номеру прізвища студента у списку групи**

|  |  |
| --- | --- |
| **№**  **варіан-ту** | **Зміст завдання** |
| 1 | 1. Розробити ієрархію класів: **Тварина**, Кішка, Собака. Розробити конструктори і методи роботи з кожним класом. В базовому класі зробити конструктор з параметрами. В похідних також з параметрами. В похідних класах створити по 3 окремі методи.  Розробити два віртуальні методи в базовому класі та їх реалізації в кожному з похідних класів.  2. Зробити клас Тварина абстрактним, а реалізацію віртуальних методів перенести в похідні класи. |
| 2 | 1. Розробити ієрархію класів: **Персона**, Робочий, Інженер.  Розробити конструктори і методи роботи з кожним класом. В базовому класі зробити конструктор з параметрами. В похідних також з параметрами. В похідних класах створити по 3 окремі методи.  Розробити два віртуальні методи в базовому класі і їх реалізації в кожному з похідних класів.  2. Зробити клас Персона абстрактним, а реалізацію методів перенести в похідні класи. |
| 3 | 1. Розробити ієрархію класів: **Співробітник**, Службовець, Робочий. Розробити конструктори і методи роботи з кожним класом. В базовому класі зробити конструктор з параметрами. В похідних також з параметрами. В похідних класах створити по 3 окремі методи.  Розробити два віртуальні методи в базовому класі і їх реалізації в кожному з похідних класів.  2. Зробити клас **Співробітник** абстрактним, а реалізацію методів перенести в похідні класи. |
| 4 | 1. Розробити ієрархію класів: Деталь, Механізм, **Виріб**, Вузол.  Розробити конструктори і методи роботи з кожним класом. В базовому класі зробити конструктор з параметрами. В похідних також з параметрами. В похідних класах створити по 3 окремі методи.  Розробити два віртуальні методи в базовому класі та їх реалізації в кожному з похідних класів.  2. Зробити клас **Виріб** абстрактним, а реалізацію методів перенести в похідні класи. |
| 5 | 1. Розробити ієрархію класів: **Організація**, Страхова Компанія, Нафтогазова Компанія, Завод.  Розробити конструктори і методи роботи з кожним класом. В базовому класі зробити конструктор з параметрами. В похідних також з параметрами. В похідних класах створити по 3 окремі методи.  Розробити два віртуальні методи в базовому класі та їх реалізації в кожному з похідних класів.  2. Зробити клас **Організація** абстрактним, а реалізацію методів перенести в похідні класи. |
| 6 | 1. Розробити ієрархію класів: Журнал, Книга, **ДрукарськеВидання.** .Розробити конструктори і методи роботи з кожним класом. В базовому класі зробити конструктор з параметрами. В похідних також з параметрами. В похідних класах створити по 3 окремі методи.  Розробити два віртуальні методи в базовому класі та їх реалізації в кожному з похідних класів.  2. Зробити клас **ДрукарськеВидання** абстрактним, а реалізацію методів перенести в похідні класи. |
| 7 | 1. Розробити ієрархію класів: **Рослина,** дерево, квіти.  Розробити конструктори і методи роботи з кожним класом. В базовому класі зробити конструктор з параметрами. В похідних також з параметрами. В похідних класах створити по 3 окремі методи.  Розробити два віртуальні методи в базовому класі та їх реалізації в кожному з похідних класів.  2. Зробити клас **Рослина** абстрактним, а реалізацію методів перенести в похідні класи. |
| 8 | 1. Розробити ієрархію класів: **Місцевість**, Село, Місто. Розробити конструктори і методи роботи з кожним класом. В базовому класі зробити конструктор з параметрами. В похідних також з параметрами. В похідних класах створити по 3 окремі методи.  Розробити два віртуальні методи в базовому класі та їх реалізації в кожному з похідних класів.  2. Зробити клас **Місцевість** абстрактним, а реалізацію методів перенести в похідні класи. |
| 9 | 1. Розробити ієрархію класів: Іграшка, Продукти, **Товар.** Розробити конструктори і методи роботи з кожним класом. В базовому класі зробити конструктор з параметрами. В похідних також з параметрами. В похідних класах створити по 3 окремі методи.  Розробити два віртуальні методи в базовому класі та їх реалізації в кожному з похідних класів.  2. Зробити клас **Товар** абстрактним, а реалізацію методів перенести в похідні класи. |
| 10 | 1. Розробити ієрархію класів: Квитанція, Накладна, **Документ**, Рахунок. Розробити конструктори і методи роботи з кожним класом. В базовому класі зробити конструктор з параметрами. В похідних також з параметрами. В похідних класах створити по 3 окремі методи.  Розробити два віртуальні методи в базовому класі та їх реалізації в кожному з похідних класів.  2. Зробити клас **Документ** абстрактним, а реалізацію методів перенести в похідні класи. |
| 11 | 1. Розробити ієрархію класів: Автомобіль, Потяг, **Транспорт.**  Розробити конструктори і методи роботи з кожним класом. В базовому класі зробити конструктор з параметрами. В похідних також з параметрами. В похідних класах створити по 3 окремі методи.  Розробити два віртуальні методи в базовому класі та їх реалізації в кожному з похідних класів.  2. Зробити клас **Транспорт** абстрактним, а реалізацію методів перенести в похідні класи. |
| 12 | 1. Розробити ієрархію класів: **Викладач**, професор, доцент.  Розробити конструктори і методи роботи з кожним класом. В базовому класі зробити конструктор з параметрами. В похідних також з параметрами. В похідних класах створити по 3 окремі методи.  Розробити два віртуальні методи в базовому класі та їх реалізації в кожному з похідних класів.  2. Зробити клас **Викладач** абстрактним, а реалізацію методів перенести в похідні класи. |
| 13 | 1. Розробити ієрархію класів: **Рослинний\_продукт** , Овочі, Фрукти, Зернові. Розробити конструктори і методи роботи з кожним класом. В базовому класі зробити конструктор з параметрами. В похідних також з параметрами. В похідних класах створити по 3 окремі методи.  Розробити два віртуальні методи в базовому класі та їх реалізації в кожному з похідних класів.  2. Зробити клас **Рослинний\_продукт** абстрактним, а реалізацію методів перенести в похідні класи. |
| 14 | 1. Розробити ієрархію класів: **Книга**, Підручник, Конспект, Навчальний Матеріал. Розробити конструктори і методи роботи з кожним класом. В базовому класі зробити конструктор з параметрами. В похідних також з параметрами. В похідних класах створити по 3 окремі методи.  Розробити два віртуальні методи в базовому класі та їх реалізації в кожному з похідних класів.  2. Зробити клас **Книга** абстрактним, а реалізацію методів перенести в похідні класи. |
| 15 | 1. Розробити ієрархію класів: **Меблі**, Корпусні, М'які меблі, Кухні. Розробити конструктори і методи роботи з кожним класом. В базовому класі зробити конструктор з параметрами. В похідних також з параметрами. В похідних класах створити по 3 окремі методи.  Розробити два віртуальні методи в базовому класі та їх реалізації в кожному з похідних класів.  2. Зробити клас **Меблі**,абстрактним, а реалізацію методів перенести в похідні класи. |

**Література**

1. Лекція 9

2.C# 2005 и платформа .Net 3.0 для профессионалов. Нейгел К., Ивьен Б. и др. –М.: ООО “И.Д. Вильямс", 2008. –1376 с.

3. Шилдт Г. С# 4.0: полное руководство. : Пер. с англ. — М. : ООО "И.Д. Вильямс", 2011. — 1056 с.

**Лабораторна робота №11**

**Класи. Інтерфейси**

**Мета роботи:**

1. Вивчення спадкування інтерфейсів

**Методичні рекомендації до виконання лабораторної роботи**

1. Уважно прочитайте теоретичну частину та цю лабораторну роботу.

2. Виконайте і проаналізуйте всі наведені в лекції приклади.

3. Зверніть увагу, що клас, який успадковує інтерфейс, зобов'язаний повністю реалізувати **всі** методи інтерфейсу. У цьому відмінність від класу, що успадковує абстрактний клас, де похідний клас може реалізувати лише деякі методи базового абстрактного класу.

Клас, що успадковує інтерфейс і реалізує його методи, повинен оголосити відповідні методи класу **відкритими** (**public)**.

4. Для поглибленого вивчення цього матеріалу прочитайте розділ 4 книги [2] та розділ 12 книги [3].

**2. Варіанти завдань для Лабораторної роботи № 10**

**Номер варіанта відповідає номеру прізвища студента у списку групи**

|  |  |
| --- | --- |
| **№**  **варіан-ту** | **Зміст завдання** |
| 1 | 1. Створити інтерфейс з двома методами: Golos () i Poroda() та шаблонами властивостей Name i Age. Реалізувати методи і властивості в похідних класах Кішка, Собака базового класу Тварина. Оголошення класів розмістити в окремих файлах.  2. Модифікувати завдання 1, створивши ієрархію інтерфейсів (по одному методу в кожному). Реалізувати методи і властивості в похідних класах Кішка, Собака.  3. Створити ієрархію класів **Банківський рахунок,** Поточний, Депозитний. Розробити інтерфейс з двома методами: Новий\_рахунок(), Видалити\_рахунок(). Реалізувати інтерфейсні методи в базовому класі. В похідних класах викликати ці методи та реалізувати методи: Поповнити рахунок(), Зняти\_з рахунку(). В програмі організувати роботу з рахунками. |
| 2 | 1. Створити інтерфейс з двома методами: Zarplata() I Role() та шаблонами властивостей Name, Stag\_Rob. Реалізувати методи і властивості інтерфейсу у похідних класахРобочий, Інженер базоаого класу **Персона.** Оголошення класів розмістити в окремих файлах.  2. Модифікувати завдання 1, створивши ієрархію інтерфейсів (по одному методу в кожному). Реалізувати методи і властивості інтерфейсу у похідних класахРобочий, Інженер базового класу **Персона.**  3. Створити ієрархію класів **Банківський рахунок,** Поточний, Депозитний. Розробити інтерфейс з двома методами: Новий\_рахунок(), Видалити\_рахунок(). Реалізувати інтерфейсні методи в базовому класі. В похідних класах викликати ці методи та реалізувати методи: Поповнити рахунок(), Зняти\_з рахунку(). В програмі організувати роботу з рахунками. |
| 3 | 1. Створити інтерфейс з двома методами: Role() i Zarplata () та шаблонами властивостей Name i Stag\_Rob. Реалізувати методи і властивості інтерфейсу у похідних класахСлужбовець, Робочий базового класу **Співробітник.** Оголошення класів розмістити в окремих файлах.  2. Модифікувати завдання 1, створивши ієрархію інтерфейсів (по одному методу в кожному). Реалізувати методи і властивості інтерфейсу у похідних класахСлужбовець, Робочий .  3. Створити ієрархію класів **Банківський рахунок,** Поточний, Депозитний. Розробити інтерфейс з двома методами: Новий\_рахунок(), Видалити\_рахунок(). Реалізувати інтерфейсні методи в базовому класі. В похідних класах викликати ці методи та реалізувати методи: Поповнити рахунок(), Зняти\_з рахунку(). В програмі організувати роботу з рахунками. |
| 4 | 1. Створити інтерфейс з двома методами: Zamina() i Type () та шаблонами властивостей Name i Year. Реалізувати методи і властивості інтерфейсу у похідних класахДеталь, Механізм базового класу **Виріб.** Оголошення класів розмістити в окремих файлах.  2. Модифікувати завдання 1, створивши ієрархію інтерфейсів (по одному методу в кожному). Реалізувати методи і властивості інтерфейсу у похідних класахДеталь, Механізм.  3. Створити ієрархію класів **Банківський рахунок,** Поточний, Депозитний. Розробити інтерфейс з двома методами: Новий\_рахунок(), Видалити\_рахунок(). Реалізувати інтерфейсні методи в базовому класі. В похідних класах викликати ці методи та реалізувати методи: Поповнити рахунок(), Зняти\_з рахунку(). В програмі організувати роботу з рахунками. |
| 5 | 1. Створити інтерфейс з двома методами: About\_Company() i Get\_Address () та шаблонами властивостей Name i President. Реалізувати методи і властивості інтерфейсу у похідних класахСтрахова Компанія, Нафтогазова Компанія базового класу **Організація.** Оголошення класів розмістити в окремих файлах.  2. Модифікувати завдання 1, створивши ієрархію інтерфейсів (по одному методу в кожному). Реалізувати методи і властивості інтерфейсу у похідних класахСлужбовець, Робочий .  3. Створити ієрархію класів **Банківський рахунок,** Поточний, Депозитний. Розробити інтерфейс з двома методами: Новий\_рахунок(), Видалити\_рахунок(). Реалізувати інтерфейсні методи в базовому класі. В похідних класах викликати ці методи та реалізувати методи: Поповнити рахунок(), Зняти\_з рахунку(). В програмі організувати роботу з рахунками. |
| 6 | 1. Створити інтерфейс з двома методами: About\_Company() i Get\_Price () та шаблонами властивостей Name i Type. Реалізувати методи і властивості інтерфейсу у похідних класахЖурнал, Книга базового класу **Друкарське\_Видання.**  Оголошення класів розмістити в окремих файлах.  2. Модифікувати завдання 1, створивши ієрархію інтерфейсів (по одному методу в кожному). Реалізувати методи і властивості інтерфейсу у похідних класахЖурнал, Книга.  3. Створити ієрархію класів **Банківський рахунок,** Поточний, Депозитний. Розробити інтерфейс з двома методами: Новий\_рахунок(), Видалити\_рахунок(). Реалізувати інтерфейсні методи в базовому класі. В похідних класах викликати ці методи та реалізувати методи: Поповнити рахунок(), Зняти\_з рахунку(). В програмі організувати роботу з рахунками. |
| 7 | 1. Створити інтерфейс з двома методами: Get\_Price (), Сount() та шаблонами властивостей Name i About. Реалізувати методи і властивості інтерфейсу у похідних класах **Д**ерево, Квіти базового класу **Рослина.** Оголошення класів розмістити в окремих файлах.  2. Модифікувати завдання 1, створивши ієрархію інтерфейсів (по одному методу в кожному). Реалізувати методи і властивості інтерфейсу у похідних класахДерево, Квіти.  3. Створити ієрархію класів **Банківський рахунок,** Поточний, Депозитний. Розробити інтерфейс з двома методами: Новий\_рахунок(), Видалити\_рахунок(). Реалізувати інтерфейсні методи в базовому класі. В похідних класах викликати ці методи та реалізувати методи: Поповнити рахунок(), Зняти\_з рахунку(). В програмі організувати роботу з рахунками. |
| 8 | 1. Створити інтерфейс з двома методами: Get\_Population (), Get\_Area() та шаблонами властивостей Name i Address. Реалізувати методи і властивості інтерфейсу у похідних класах Село, Місто базового класу **Місцевість**.  2. Модифікувати завдання 1, створивши ієрархію інтерфейсів (по одному методу в кожному). Реалізувати методи і властивості інтерфейсу у похідних класахСело, Місто.  3. Створити ієрархію класів **Банківський рахунок,** Поточний, Депозитний. Розробити інтерфейс з двома методами: Новий\_рахунок(), Видалити\_рахунок(). Реалізувати інтерфейсні методи в *базовому класі*. В похідних класах викликати ці методи та реалізувати методи: Поповнити рахунок(), Зняти\_з рахунку(). В програмі організувати роботу з рахунками. |
| 9 | 1. Створити інтерфейс з двома методами: Get\_Type (), Count() та шаблонами властивостей Name i Price. Реалізувати методи і властивості інтерфейсу у похідних класах Іграшка, Продукти базового класу **Товар**. Оголошення класів розмістити в окремих файлах.  2. Модифікувати завдання 1, створивши ієрархію інтерфейсів (по одному методу в кожному). Реалізувати методи і властивості інтерфейсу у похідних класахІграшка, Продукти.  3. Створити ієрархію класів **Банківський рахунок,** Поточний, Депозитний. Розробити інтерфейс з двома методами: Новий\_рахунок(), Видалити\_рахунок(). Реалізувати інтерфейсні методи в *базовому класі*. В похідних класах викликати ці методи та реалізувати методи: Поповнити рахунок(), Зняти\_з рахунку(). В програмі організувати роботу з рахунками. Оголошення класів розмістити в окремих файлах. |
| 10 | 1. Створити інтерфейс з двома методами: Get\_Requisites (), Get\_Cost() та шаблонами властивостей Author i Date. Реалізувати методи і властивості інтерфейсу у похідних класах Квитанція, Накладна базового класу **Документ**. Оголошення класів розмістити в окремих файлах.  2. Модифікувати завдання 1, створивши ієрархію інтерфейсів (по одному методу в кожному). Реалізувати методи і властивості інтерфейсу у похідних класах Квитанція, Накладна .  3. Створити ієрархію класів **Банківський рахунок,** Поточний, Депозитний. Розробити інтерфейс з двома методами: Новий\_рахунок(), Видалити\_рахунок(). Реалізувати інтерфейсні методи в *базовому класі*. В похідних класах викликати ці методи та реалізувати методи: Поповнити рахунок(), Зняти\_з рахунку(). В програмі організувати роботу з рахунками. |
| 11 | 1. Створити інтерфейс з двома методами: Get\_Requisites (), Get\_Cost() та шаблонами властивостей Author i Date. Реалізувати методи і властивості інтерфейсу у похідних класах Автомобіль, Потяг базового класу **Транспорт**.  2. Модифікувати завдання 1, створивши ієрархію інтерфейсів (по одному методу в кожному). Реалізувати методи і властивості інтерфейсу у похідних класах Квитанція, Накладна.  3. Створити ієрархію класів **Банківський рахунок,** Поточний, Депозитний. Розробити інтерфейс з двома методами: Новий\_рахунок(), Видалити\_рахунок(). Реалізувати інтерфейсні методи в *базовому класі*. В похідних класах викликати ці методи та реалізувати методи: Поповнити рахунок(), Зняти\_з рахунку(). В програмі організувати роботу з рахунками. |

**Література**

1. Лекція 11

2.C# 2005 и платформа .Net 3.0 для профессионалов. Нейгел К., Ивьен Б. и др. –М.: ООО “И.Д. Вильямс", 2008. –1376 с.

3. Шилдт Г. С# 4.0: полное руководство. : Пер. с англ. — М. : ООО "И.Д. Вильямс", 2011. — 1056 с.

**Лабораторна робота №12**

**ТЕМА: ОСНОВИ T-SQL**

**План**

1. . Додавання елементів в БД за допомогою запитів

2**.** Видалення елементів із БД за допомогою запитів

3.Сортування БД за допомогою запитів

4. Фільтрація елементів БД

## 5.Оновлення даних. Команда UPDATE

6. Видалення даних. команда DELETE

## 7.Оператори GROUP BY и HAVING

1. **Додавання елементів в БД за допомогою запитів**

Для додавання даних застосовується команда INSERT, яка має наступний формальний синтаксис:

**INSERT [INTO] ім'я\_таблиці [(спісок\_столбців)] VALUES (значення1, значення2, ... значеніеN)**

Спочатку йде вираз INSERT INTO, потім в дужках можна вказати список стовпців через кому, в які треба додавати дані, і в кінці після слова VALUES в дужках перераховуються значення, що додаються для стовпців.

Наприклад, нехай раніше була створена наступна база даних:

**CREATE DATABASE productsdb;**

**GO**

**USE productsdb;**

**CREATE TABLE Products**

**(**

**Id INT IDENTITY PRIMARY KEY,**

**ProductName NVARCHAR (30) NOT NULL,**

**Manufacturer NVARCHAR (20) NOT NULL,**

**ProductCount INT DEFAULT 0,**

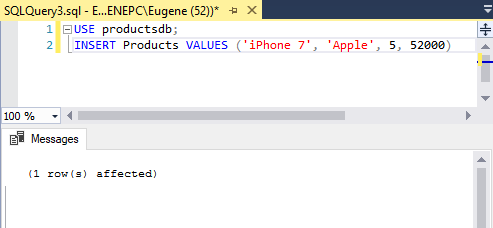
**Price MONEY NOT NULL**

**)**

Додамо в неї один рядок за допомогою команди INSERT:

**INSERT Products VALUES ( 'iPhone 7', 'Apple', 5, 52000)**

Після вдалого виконання в SQL Server Management Studio в поле повідомлень повинно з'явитися повідомлення "1 row (s) affected":



Варто враховувати, що значення для стовпців в дужках після ключового слова VALUES передаються по порядку їх оголошення. Наприклад, в виразі CREATE TABLE вище можна побачити, що першим стовпцем йде Id. Але так як для нього заданий атрибут IDENTITY, то значення цього стовпця автоматично генерується, і його можна не вказувати. Другий стовпець представляє ProductName, тому перше значення - рядок "iPhone 7" буде передано саме цього стовпцю. Друге значення - рядок "Apple" буде передано в третій стовпець Manufacturer і так далі. Тобто значення передаються стовпцями наступним чином:

ProductName: 'iPhone 7'

Manufacturer: 'Apple'

ProductCount: 5

Price: 52000

Також при введенні значень можна вказати безпосередні стовпці, в які будуть додаватися значення:

**INSERT INTO Products (ProductName, Price, Manufacturer)**

**VALUES ( 'iPhone 6S', 41000, 'Apple')**

Тут значення вказуються тільки для трьох стовпців. Причому тепер значення передаються в порядку проходження стовпців:

ProductName: 'iPhone 6S'

Manufacturer: 'Apple'

Price: 41000

Для невказаних стовпців (в даному випадку ProductCount) буде додаватися значення за замовчуванням, якщо заданий атрибут DEFAULT, або значення NULL. При цьому не вказані стовпці повинні допускати значення NULL або мати атрибут DEFAULT.

Також ми можемо додати відразу кілька рядків:

**INSERT INTO Products**

**VALUES**

**( 'IPhone 6', 'Apple', 3, 36000),**

**( 'Galaxy S8', 'Samsung', 2, 46000),**

**( 'Galaxy S8 Plus', 'Samsung', 1, 56000)**

В даному випадку в таблицю будуть додані три рядки.

Також при додаванні ми можемо вказати, щоб для стовпця використовувалося значення за замовчуванням за допомогою ключового слова DEFAULT або значення NULL:

**INSERT INTO Products (ProductName, Manufacturer, ProductCount, Price)**

**VALUES ( 'Mi6', 'Xiaomi', DEFAULT, 28000)**

В даному випадку для стовпця ProductCount буде використано значення за замовчуванням (якщо воно встановлено, якщо його немає - то NULL).

Якщо всі стовпці мають атрибут DEFAULT, що визначає значення за замовчуванням, або допускають значення NULL, то можна для всіх стовпців вставити значення за замовчуванням:

**INSERT INTO Products**

**DEFAULT VALUES**

Але якщо брати таблицю Products, то подібна команда завершиться з помилкою, так як кілька полів не мають атрибута DEFAULT і при цьому не допускають значення NULL.

**2. Отримання елементів із БД за допомогою запитів**

Для отримання даних застосовується команда SELECT. У спрощеному вигляді вона має наступний синтаксис:

**SELECT спісок\_столбцов FROM ім'я\_таблиці**

Наприклад, нехай раніше була створена таблиця Products, і в неї додані деякі початкові дані:

**CREATE TABLE Products**

**(**

**Id INT IDENTITY PRIMARY KEY,**

**ProductName NVARCHAR (30) NOT NULL,**

**Manufacturer NVARCHAR (20) NOT NULL,**

**ProductCount INT DEFAULT 0,**

**Price MONEY NOT NULL**

**);**

**INSERT INTO Products**

**VALUES**

**( 'IPhone 6', 'Apple', 3, 36000),**

**( 'IPhone 6S', 'Apple', 2, 41000),**

**( 'IPhone 7', 'Apple', 5, 52000),**

**( 'Galaxy S8', 'Samsung', 2, 46000),**

**( 'Galaxy S8 Plus', 'Samsung', 1, 56000),**

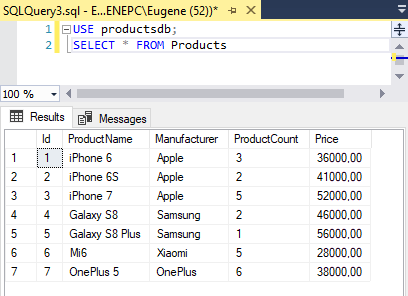
**( 'Mi6', 'Xiaomi', 5, 28000),**

**( 'OnePlus 5', 'OnePlus', 6, 38000)**

Отримаємо всі об'єкти з цієї таблиці:

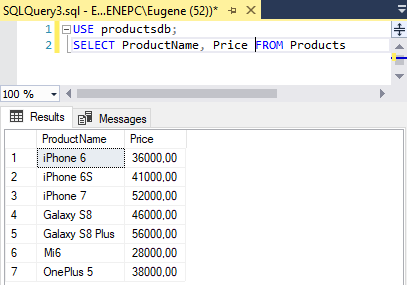
**SELECT \* FROM Products**

Символ зірочка \* вказує, що нам треба отримати всі стовпці.



Отримання всіх стовпців за допомогою символу зірочки \* вважається не дуже гарною практикою, так як, як правило, не всі стовпці бувають потрібні. І більш оптимальний підхід полягає у вказівці всіх необхідних стовпців після слова SELECT. Виняток становить той випадок, коли потрібно отримати дані по абсолютно всіх стовпцях таблиці. Також використання символу \* може бути переважно в таких ситуаціях, коли точно не відомі назви стовпців.

Якщо нам потрібно отримати дані не по всіх, а по якимось конкретним стовпцями, то тоді всі ці специфікації стовпців перераховуються через кому після SELECT:

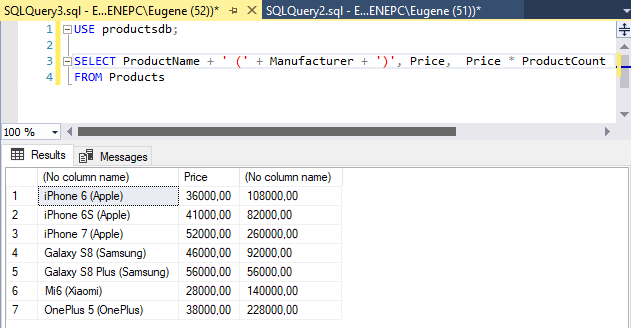


Специфікація стовпця необов'язково повинна представляти його назву. Це може бути будь-який вираз, наприклад, результат арифметичної операції. Так, виконаємо наступний запит:

**SELECT ProductName + '(' + Manufacturer + ')', Price, Price \* ProductCount**

**FROM Products**

Тут при вибірці будуть створюватися три стовпці. Перший стовпець представляє результат об'єднання двох стовпців ProductName і Manufacturer. Другий стовпець - стандартний стовпець Price. А третій стовпець представляє значення стовпця Price, помножене на значення стовпця ProductCount.



За допомогою оператора AS можна змінити назву вихідного стовпця або визначити його псевдонім:

**SELECT**

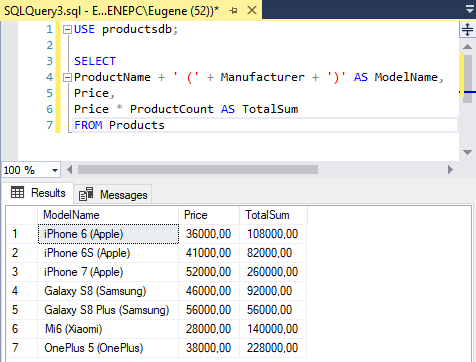
**ProductName + '(' + Manufacturer + ')' AS ModelName,**

**Price,**

**Price \* ProductCount AS TotalSum**

**FROM Products**

В даному випадку результатом вибірки є дані по 3-м стовпцях. Перший стовпець ModelName об'єднує стовпці ProductName і Manufacturere, другий пропонує стандартний стовпець Price. Третій стовпець TotalSum зберігає добуток стовпців ProductCount і Price.

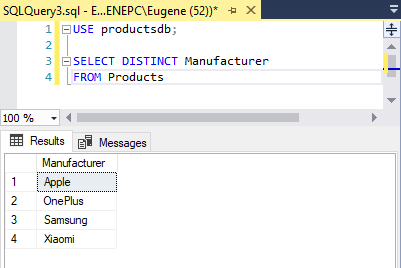


**DISTINCT**

Оператор DISTINCT дозволяє вибрати унікальні рядки. Наприклад, в нашому випадку в таблиці може бути по кілька товарів від одних і тих же виробників. Виберемо всіх виробників:

SELECT DISTINCT Manufacturer

FROM Products



В даному випадку критерієм розмежування рядків є стовпець Manufacturer. Тому в результуючій вибірці будуть тільки унікальні значення Manufacturer. І якщо, наприклад, в базі даних є два товари з виробником Apple, то ця назва буде зустрічатися в результуючій вибірці тільки один раз.

**Вибірка з додаванням**

**SELECT INTO**

Вираз SELECT INTO дозволяє вибрати з однієї таблиці деякі дані в іншу таблицю, при цьому друга таблиця створюється автоматично. наприклад:

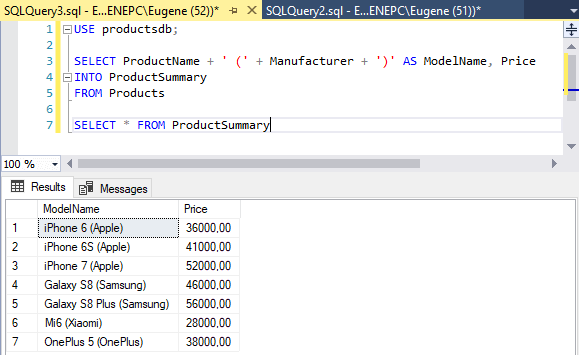
**SELECT ProductName + '(' + Manufacturer + ')' AS ModelName, Price**

**INTO ProductSummary**

**FROM Products**

**SELECT \* FROM ProductSummary**

Після виконання цієї команди в базі даних буде створена ще одна таблиця ProductSummary, яка буде мати два стовпці ModelName і Price, а дані для цих стовпців будуть взяті з таблиці Products:



**3.Сортування БД за допомогою запитів**

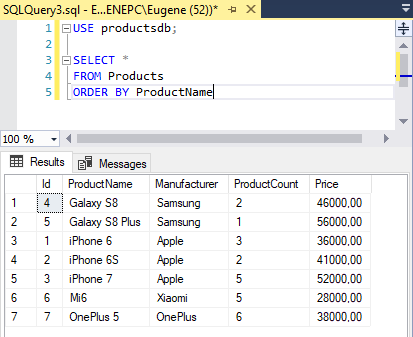
Оператор ORDER BY дозволяє впорядкувати значення за певним стовпцем:

**SELECT \***

**FROM Products**

**ORDER BY ProductName**

В даному випадку рядки сортуються за зростанням значення стовпця ProductName:



Сортування також можна проводити за псевдонімом стовпчика, який визначається за допомогою оператора AS:

**SELECT ProductName, ProductCount \* Price AS TotalSum**

**FROM Products**

**ORDER BY TotalSum**

За замовчуванням застосовується сортування за зростанням. За допомогою додаткового оператора DESC можна задати сортування по спадаючій.

**SELECT ProductName**

**FROM Products**

**ORDER BY ProductName DESC**

За замовчуванням замість DESC використовується оператор ASC:

**SELECT ProductName**

**FROM Products**

**ORDER BY ProductName ASC**

Якщо необхідно впорядкувати відразу за кількома стовпцями, то всі вони перераховуються після оператора ORDER BY:

**SELECT ProductName, Price, Manufacturer**

**FROM Products**

**ORDER BY Manufacturer, ProductName**

У цьому випадку спочатку рядки упорядковано відповідно до колонки Manufacturer по зростанню. Потім якщо є два рядки, в яких стовпець Manufacturer має однакове значення, то вони сортуються по стовпцю ProductName також по зростанню. Але знову ж таки за допомогою ASC і DESC можна окремо для різних стовпців визначити сортування по зростанню і зменшенням:

**SELECT ProductName, Price, Manufacturer**

**FROM Products**

**ORDER BY Manufacturer ASC, ProductName DESC**

**4.Фільтрація елементів БД**

Для фільтрації в команді SELECT застосовується оператор WHERE. Після цього оператора ставиться умова, якій повинен відповідати рядок:

WHERE умова

Якщо умова істина, то рядок потрапляє в результуючу вибірку.

Як можна використовувати операції порівняння? Ці операції порівнюють два вирази. В T-SQL можна застосовувати такі операції порівняння:

=: Порівняння на рівність (на відміну від сі-подібних мов в T-SQL для порівняння на рівність використовується один знак дорівнює)

<>: Порівняння на нерівність

<: Менше ніж

>: Більше ніж

! <: Не менш ніж

!>: Але лише

<=: Менше ніж або дорівнює

> =: Більше ніж або дорівнює

Наприклад, знайдемо всіх товари, виробником яких є компанія Samsung:

**SELECT \* FROM Products**

**WHERE Manufacturer = 'Samsung'**

Варто відзначити, що в даному випадку регістр не має значення, і ми могли б використовувати для пошуку і рядок "Samsung", і "SAMSUNG", і "samsung". Всі ці варіанти давали б еквівалентний результат вибірки.

Інший приклад - знайдемо всі товари, у яких ціна більше 45000:

**SELECT \* FROM Products**

**WHERE Price> 45000**

**Логічні оператори**

Для об'єднання декількох умов в одну можуть використовуватися логічні оператори. В T-SQL є такі логічні оператори:

AND: операція логічного І. Вона об'єднує два вирази:

OR: операція логічного АБО. Вона також об'єднує два вирази:

NOT: операція логічного заперечення. Якщо вираз в цій операції помилково, то загальне умова істина.

Якщо ці оператори зустрічаються в одному вираженні, то спочатку виконується NOT, потім AND і в кінці OR.

Наприклад, виберемо усі товари, у яких виробник Samsung і одночасно ціна більше 50000:

**SELECT \* FROM Products**

**WHERE Manufacturer = 'Samsung' AND Price> 50000**

**IS NULL**

Ряд стовпців може допускати значення NULL. Це значення не еквівалентне порожній рядку ''. NULL являє повна відсутність будь-якого значення. І для перевірки на наявність подібного значення застосовується оператор IS NULL.

Наприклад, виберемо усі товари, у яких не встановлено поле ProductCount:

**SELECT \* FROM Products**

**WHERE ProductCount IS NULL**

Якщо, навпаки, необхідно отримати рядки, у яких поле ProductCount не дорівнює NULL, то можна використовувати оператор NOT:

**SELECT \* FROM Products**

**WHERE ProductCount IS NOT NULL**

## 5.Оновлення даних. Команда UPDATE

Для зміни вже наявних рядків в таблиці застосовується команда UPDATE. Вона має наступний формальний синтаксис:

**UPDATE ім'я\_таблиці**

**SET стовпець1 = значення1, стовпець2 = значення2, ... стовпецьN = значеніеN**

**[FROM вибірка AS псевдонім\_виборки]**

**[WHERE умован\_овленія]**

Наприклад, збільшимо у всіх товарів ціну на 5000:

**UPDATE Products**

**SET Price = Price +5000**

Використуємо критерій, і змінимо назву виробника з "Samsung" на "Samsung Inc.":

UPDATE Products

SET Manufacturer = 'Samsung Inc.'

WHERE Manufacturer = 'Samsung'

Більш складний запит - замінимо у поля Manufacturer значення "Apple" на "Apple Inc." в перших 2 рядках:

**UPDATE Products**

**SET Manufacturer = 'Apple Inc.'**

**FROM**

**(SELECT TOP 2 \* FROM Products WHERE Manufacturer = 'Apple') AS Selected**

**WHERE Products.Id = Selected.Id**

За допомогою підзапиту після ключового слова FROM проводиться вибірка перших двох рядків, в яких Manufacturer = 'Apple'. Для цієї вибірки буде визначено псевдонім Selected.

Далі йде умова оновлення Products.Id = Selected.Id. Тобто фактично ми маємо справу з двома таблицями - Products і Selected (яка є похідною від Products). У Selected знаходиться два перших рядки, в яких Manufacturer = 'Apple'. У Products - взагалі всі рядки. І оновлення проводиться тільки для тих рядків, які є в вибірці Selected. Тобто якщо в таблиці Products десятки товарів з виробником Apple, то оновлення торкнеться тільки двох перших з них.

**6. Видалення даних. команда DELETE**

Для видалення застосовується команда DELETE:

**DELETE [FROM] ім'я\_таблиці**

**WHERE умова\_видалення**

Наприклад, видалимо рядки, у яких id дорівнює 9:

**DELETE Products**

**WHERE Id = 9**

Або видалимо всі товари, виробником яких є Xiaomi і які мають ціну менше 15000:

**DELETE Products**

**WHERE Manufacturer = 'Xiaomi' AND Price <15000**

Більш складний приклад - видалимо перші два товари, у яких виробник - Apple:

**DELETE Products FROM**

**(SELECT TOP 2 \* FROM Products**

**WHERE Manufacturer = 'Apple]') AS Selected**

**WHERE Products.Id = Selected.Id**

Після першого оператора FROM йде вибірка двох рядків з таблиці Products. Цій вибірці призначається псевдонім Selected за допомогою оператора AS. Далі встановлюємо умову, якщо Id в таблиці Products має те ж значення, що і Id у вибірці Selected, то рядок видаляється.

## 7.Оператори GROUP BY і HAVING

Для групування даних в T-SQL застосовуються оператори GROUP BY і HAVING, для використання яких застосовується наступний формальний синтаксис:

**SELECT стовпці**

**FROM таблиця**

**[WHERE умова\_фільтраціі\_рядків]**

**[GROUP BY стовпці\_для\_групування]**

**[HAVING умова\_фільтраціі\_груп]**

**[ORDER BY стовпці\_для\_сортування]**

**ЗАВДАННЯ**

1. Створити в середовищі Management Studio MS SQL по 5 запитів для БД проектів. (Для кожного студента окремо, навіть, якщо вони пишуть один проект)

**Лабораторна робота №14**

**Тема:ADO.NET**

**План**

1. Введення в ADO.NET

**2.** C# и MS SQL Server

**1.Введення в ADO.NET**

ADO.NET надає технологію роботи з даними, яка заснована на платформі .NET Framework. Ця технологія являє нам набір класів, через які ми можемо відправляти запити до баз даних, встановлювати підключення, отримувати відповідь від бази даних і виробляти ряд інших операцій.

Причому важливо зазначити, що систем управління базами даних може бути безліч. У своїй сутності вони можуть відрізнятися. MS SQL Server, наприклад, для створення запитів використовує мову T-SQL, а MySQL і Oracle застосовують мову PL-SQL. Різні системи баз даних можуть мати різні типи даних. Однак функціонал ADO.NET побудований таким чином, щоб надати розробникам уніфікований інтерфейс для роботи з самими різними СУБД.

Основу інтерфейсу взаємодії з базами даних в ADO.NET представляє обмежене коло об'єктів: Connection, Command, DataReader, DataSet і DataAdapter. За допомогою об'єкта Connection відбувається установка підключення до джерела даних. Об'єкт Command дозволяє виконувати операції з даними з БД. Об'єкт DataReader зчитує отримані в результаті запиту дані. Об'єкт DataSet призначений для зберігання даних з БД і дозволяє працювати з ними незалежно від БД. І об'єкт DataAdapter є посередником між DataSet і джерелом даних. Головним чином, через ці об'єкти і буде йти робота з базою даних.

Однак щоб використовувати один і той же набір об'єктів для різних джерел даних, необхідний відповідний провайдер даних. Власне через провайдер даних в ADO.NET і здійснюється взаємодія з базою даних. Причому для кожного джерела даних в ADO.NET може бути свій провайдер, який власне і визначає конкретну реалізацію вищевказаних класів.

За замовчуванням в ADO.NET є наступні вбудовані провайдери:

Провайдер для MS SQL Server

Провайдер для OLE DB (Надає доступ до деяких старих версій MS SQL Server, а також до БД Access, DB2, MySQL і Oracle)

Провайдер для ODBC (провайдер для тих джерел даних, для яких немає своїх провайдерів)

Провайдер для Oracle

Провайдер EntityClient. Провайдер даних для технології ORM Entity Framework

Провайдер для сервера SQL Server Compact 4.0

Крім цих провайдерів, які є вбудованими, існує також безліч інших, призначених для різних баз даних, наприклад, для MySQL.

Основні простори імен, які використовуються в ADO.NET:

System.Data: визначає класи, інтерфейси, делегати, які реалізують архітектуру ADO.NET

System.Data.Common: містить класи, спільні для всіх провайдерів ADO.NET

System.Data.Design: визначає класи, які використовуються для створення своїх власних наборів даних

System.Data.Odbc: визначає функціональність провайдера даних для ODBC

System.Data.OleDb: визначає функціональність провайдера даних для OLE DB

System.Data.Sql: зберігає класи, які підтримують специфічну для SQL Server функціональність

System.Data.OracleClient: визначає функціональність провайдера для баз даних Oracle

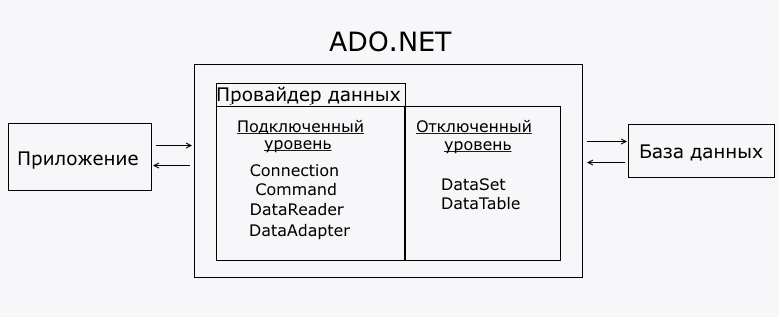
System.Data.SqlClient: визначає функціональність провайдера для баз даних MS SQL Server

System.Data.SqlServerCe: визначає функціональність провайдера для SQL Server Compact 4.0

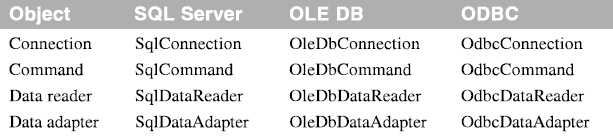
System.Data.SqlTypes: містить класи для типів даних MS SQL Servera

Microsoft.SqlServer.Server: зберігає компоненти для взаємодії SQL Server і середовища CLR

Схематично архітектуру ADO.NET можна представити таким чином:



Функціонально класи ADO.NET можна розбити на два рівня: підключень і відключень. Кожен провайдер даних .NET реалізує свої версії об'єктів Connection, Command, DataReader, DataAdapter і ряду інших, який складають рівень підключень. Тобто за допомогою них встановлюється підключення до БД і виконується з нею взаємодію. Як правило, реалізації цих об'єктів для кожного конкретного провайдера в своїй назві мають префікс, який вказує на назву провайдера:



Інші класи, такі як DataSet, DataTable, DataRow, DataColumn і ряд інших складають рівень відключень, так як після отримання даних в DataSet ми можемо працювати з цими даними незалежно від того, чи встановлено підключення чи ні. Тобто після отримання даних з БД додаток може бути відключено від джерела даних.

# 2.C# и MS SQL Server

## Рядок підключення

Після визначення джерела даних ми можемо до нього підключатися. Для цього створимо проект простого консольного застосування.

Насамперед нам треба визначити рядок підключення, що надає інформацію про базу даних та сервері, до яких належить встановити підключення:

**class Program**

**{**

**static void Main (string [] args)**

**{**

**string connectionString = @ "Data Source =. \ SQLEXPRESS; Initial Catalog = usersdb; Integrated Security = True";**

**}**

**}**

При використанні різних систем управління базами даних, різних провайдерів даних .NET рядок підключення може відрізнятися. Навіть для підключення однієї і тієї ж бази даних рядок підключення може змінюватися в залежності від обставин.

Рядок підключення представляє набір параметрів у вигляді пар ключ = значення. В даному випадку для підключення до раніше створеної бази даних usersdb ми визначаємо рядок підключення з трьох параметрів:

Data Source: вказує на назву сервера. За замовчуванням це ". \ SQLEXPRESS". Оскільки в рядку використовується слеш, то на початку рядка ставиться символ @. Якщо ім'я сервера бази даних відрізняється, то відповідно його і треба використовувати.

Initial Catalog: вказує на назву бази даних на сервері

Integrated Security: встановлює перевірку справжності

Жорстке кодування рядка підключення (тобто її визначення в коді програми), як правило, рідко використовується. Набагато більш гнучкий шлях представляє визначення її в спеціальних конфігураційних файлах програми. У проектах десктопних додатків це файл App.config, а в веб-додатках це в основному файл Web.config. Хоча додаток також може використовувати інші способи визначення конфігурації.

У нашому випадку, так як ми створили проект консольного застосування, то у нас в проекті повинен бути файл App.config, який на даний момент має наступне визначення:

**<? Xml version = "1.0" encoding = "utf-8"?>**

**<Configuration>**

**<Startup>**

**<SupportedRuntime version = "v4.0" sku = ". NETFramework, Version = v4.6" />**

**</ Startup>**

**</ Configuration>**

Змінимо його, додавши визначення рядка підключення:

**<? Xml version = "1.0" encoding = "utf-8"?>**

**<Configuration>**

**<Startup>**

**<SupportedRuntime version = "v4.0" sku = ". NETFramework, Version = v4.6" />**

**</ Startup>**

**<ConnectionStrings>**

**<Add name = "DefaultConnection" connectionString = "Data Source =. \ SQLEXPRESS; Initial Catalog = usersdb; Integrated Security = True"**

**providerName = "System.Data.SqlClient" />**

**</ ConnectionStrings>**

**</ Configuration>**

Для визначення всіх підключень в програмі в межах вузла <configuration> додається новий вузол <connectionStrings>. У цьому вузлі визначаються рядки підключення за допомогою елемента <add>. Ми можемо використовувати в додатку безліч рядків підключення, і відповідно також у файлі можна визначити безліч елементів <add>.

Кожен рядок підключення має назву, яке визначається за допомогою атрибута name. В даному випадку рядок підключення називається "DefaultConnection". Назва може бути довільною.

Атрибут connectionString власне зберігає рядок підключення, тобто весь той текст, який ми вище визначали в методі Main. І третій атрибут providerName задає простір імен провайдера даних. Так як ми будемо підключатися до бази даних MS SQL Server, то відповідно ми будемо використовувати провайдер для SQL Server, функціональність якого полягає в просторі імен System.Data.SqlClient.

Тепер отримаємо цей рядок підключення в додатку:

**using System;**

**using System.Configuration;**

**namespace AdoNetConsoleApp**

**{**

**class Program**

**{**

**static void Main (string [] args)**

**{**

**// string connectionString = @ "Data Source =. \ SQLEXPRESS; Initial Catalog = usersdb; Integrated Security = True";**

**// отримуємо рядок підключення**

**string connectionString = ConfigurationManager.ConnectionStrings [ "DefaultConnection"]. ConnectionString;**

**Console.WriteLine (connectionString);**

**Console.Read ();**

**}**

**}**

**}**

Перш за все щоб працювати з конфігурацією додатка, нам треба додати в проект бібліотеку System.Configuration.dll.

За допомогою об'єкта ConfigurationManager.ConnectionStrings [ "назва\_рядка\_підключення"] ми можемо отримати рядок підключення і використовувати його в додатку.

Параметри рядка підключення

Application Name: назва програми. Може приймати в якості значення будь-який рядок. Значення за замовчуванням: ".Net SqlClient Data Provide"

AttachDBFileName: зберігає повний шлях де прикріплюється базі даних

Connect Timeout: часовий період в секундах, через який очікується установка підключення. Приймає одне зі значень з інтервалу 0-32767. За замовчуванням дорівнює 15.

В якості альтернативної назви параметра може використовуватися Connection Timeout

Data Source: назва примірника SQL Servera, з яким буде йти взаємодія. Це може бути назва локального сервера, наприклад,

## Створення підключення

Щоб підключатися до бази даних, нам необхідно створити і використовувати об'єкт SqlConnection:

**using System;**

**using System.Data.SqlClient;**

**namespace AdoNetConsoleApp**

**{**

**class Program**

**{**

**static void Main (string [] args)**

**{**

**string connectionString = @ "Data Source =. \ SQLEXPRESS; Initial Catalog = usersdb; Integrated Security = True";**

**// Створення підключення**

**SqlConnection connection = new SqlConnection (connectionString);**

**try**

**{**

**// Відкриваємо підключення**

**connection.Open ();**

**Console.WriteLine ( "Підключення відкрито");**

**}**

**catch (SqlException ex)**

**{**

**Console.WriteLine (ex.Message);**

**}**

**finally**

**{**

**// закриваємо підключення**

**connection.Close ();**

**Console.WriteLine ( "Підключення закрито ...");**

**}**

**Console.Read ();**

**}**

**}**

**}**

У конструктор об'єкту SqlConnection передається рядок підключення, який ініціалізує об'єкт. Щоб використовувати цей об'єкт і підключатися до бази даних, ми повинні виконати його метод Open (), а після завершення роботи з базою даних нам треба викликати метод Close () для закриття підключення. У разі виникнення помилок закриття підключення відбувається в блоці finally.

В якості альтернативного методу можна використовувати конструкцію using, яка автоматично закриває підключення:

**static void Main (string [] args)**

**{**

**string connectionString = @ "Data Source =. \ SQLEXPRESS; Initial Catalog = usersdb; Integrated Security = True";**

**using (SqlConnection connection = new SqlConnection (connectionString))**

**{**

**connection.Open ();**

**Console.WriteLine ( "Підключення відкрито");**

**}**

**Console.WriteLine ( "Підключення закрито ...");**

**Console.Read ();**

**}**

Асинхронне підключення

Для створення асинхронного підключення необхідно використовувати асинхронний метод OpenAsync ():

**class Program**

**{**

**static void Main (string [] args)**

**{**

**ConnectWithDB (). GetAwaiter ();**

**}**

**private static async Task ConnectWithDB ()**

**{**

**string connectionString = @ "Data Source =. \ SQLEXPRESS; Initial Catalog = usersdb; Integrated Security = True";**

**using (SqlConnection connection = new SqlConnection (connectionString))**

**{**

**await connection.OpenAsync ();**

**Console.WriteLine ( "Підключення відкрито");**

**}**

**Console.WriteLine ( "Підключення закрито ...");**

**}**

**}**

Отримання інформації про підключення

Об'єкт SqlConnection має низку властивостей, які дозволяють отримати інформацію про підключення:

**string connectionString = @ "Data Source =. \ SQLEXPRESS; Initial Catalog = usersdb; Integrated Security = True";**

**using (SqlConnection connection = new SqlConnection (connectionString))**

**{**

**connection.Open ();**

**Console.WriteLine ( "Підключення відкрито");**

**// Виведення інформації про підключення**

**Console.WriteLine ( "Властивості підключення:");**

**Console.WriteLine ( "\ tСтрока підключення: {0}", connection.ConnectionString);**

**Console.WriteLine ( "\ tБаза даних: {0}", connection.Database);**

**Console.WriteLine ( "\ tСервер: {0}", connection.DataSource);**

**Console.WriteLine ( "\ tВерсія сервера: {0}", connection.ServerVersion);**

**Console.WriteLine ( "\ tСостояніе: {0}", connection.State);**

**Console.WriteLine ( "\ tWorkstationld: {0}", connection.WorkstationId);**

**}**

**Console.WriteLine ( "Підключення закрито ...");**

Можливі помилки та їх вирішення

При підключенні до бази даних може зіткнутися з низкою помилок, однією з найбільш поширених є наступна:

**Unhandled Exception: System.ArgumentException: Keyword not supported ..**.

Дана помилка викликана через неправильні вказівки параметрів рядка підключення, причому неважливо, як ми визначаємо рядок підключення - в коді або в файлі конфігурації. У повідомленні про помилку йдеться який параметр вказаний неправильно, і в цьому випадку його треба виправити.

Ще одна поширена помилка говорить про те, що даної бази немає або принаймні немає подібної бази даних:

**Can not open database "назва бази даних" requested by the login. The login failed.**

**Login failed for user 'користувач'**

В цьому випадку треба переконатися, що на сервері є база даних з такою назвою, а якщо є, то перевірити, чи є доступ для даного користувача до цієї бд.

І ще одна часто зустрічається помилка:

**A network-related or instance-specific error occurred while establishing a connection to SQL Server.**

**The server was not found or was not accessible ...**

У цьому випадку, як правило, SQL Server не запущений. І його треба запустити або перезапустити, через панель служб.

Також може бути, що в рядку підключення вказано неправильне ім'я сервера, або MS SQL Server взагалі не встановлено.

## Виконання команд і SqlCommand

Після установки підключення ми можемо виконати з базою даних будь-які команди, наприклад, додати в базу даних об'єкт, видалити, змінити його або просто витягти. Команди представлені об'єктом інтерфейсу System.Data.IDbCommand. Провайдер для MS SQL надає його реалізацію у вигляді класу SqlCommand. Цей клас інкапсулює sql-вираз, який потрібно виконати.

Для виконання команди нам буде потрібно sql-вираз і об'єкт підключення:

**string connectionString = @ "Data Source =. \ SQLEXPRESS; Initial Catalog = usersdb; Integrated Security = True";**

**using (SqlConnection connection = new SqlConnection (connectionString))**

**{**

**connection.Open ();**

**SqlCommand command = new SqlCommand ();**

**command.CommandText = "SELECT \* FROM Users";**

**command.Connection = connection;**

**}**

За допомогою властивості CommandText встановлюється SQL-вираз, який буде виконуватися. В даному випадку це запит на отримання всіх об'єктів з таблиці Users. А за допомогою властивості Connection можна встановити об'єкт підключення SqlConnection.

В якості альтернативи можна було б використовувати одну з версій конструктора класу:

**string connectionString = @ "Data Source =. \ SQLEXPRESS; Initial Catalog = usersdb; Integrated Security = True";**

**string sqlExpression = "SELECT \* FROM Users";**

**using (SqlConnection connection = new SqlConnection (connectionString))**

**{**

**connection.Open ();**

**SqlCommand command = new SqlCommand (sqlExpression, connection);**

**}**

Щоб виконати команду, необхідно застосувати один з методів SqlCommand:

ExecuteNonQuery: просто виконує sql-вираз і повертає кількість змінених записів. Підходить для sql-виразів INSERT, UPDATE, DELETE.

ExecuteReader: виконує sql-вираз і повертає рядки з таблиці. Підходить для sql-вирази SELECT.

ExecuteScalar: виконує sql-вираз і повертає одне скалярне значення, наприклад, число. Підходить для sql-вирази SELECT в парі з одною із вбудованих функцій SQL, як наприклад, Min, Max, Sum, Count.

Додавання об'єктів

Виконаємо команду по додаванню одного об'єкта в таблицю Users бази даних usersdb, яка раніше була створена:

**class Program**

**{**

**static void Main (string [] args)**

**{**

**string connectionString = @ "Data Source =. \ SQLEXPRESS; Initial Catalog = usersdb; Integrated Security = True";**

**string sqlExpression = "INSERT INTO Users (Name, Age) VALUES ( 'Tom', 18)";**

**using (SqlConnection connection = new SqlConnection (connectionString))**

**{**

**connection.Open ();**

**SqlCommand command = new SqlCommand (sqlExpression, connection);**

**int number = command.ExecuteNonQuery ();**

**Console.WriteLine ( "Додано об'єктів: {0}", number);**

**}**

**Console.Read ();**

**}**

**}**

Для вставки об'єкта використовується sql-вираз INSERT, яке має наступний синтаксис:

INSERT INTO названіе\_табліци (столбец1, столбец2, столбецN) VALUES (значення1, значення2, значеніеN)

В даному випадку ми знаємо, що в базі даних у нас є таблиця Users, в якій є три стовпці - Id і Age, що зберігають ціле число, і Name, який зберігає рядок. Тому відповідно ми додаємо для стовпця Name значення 'Tom', а для стовпця Age число 18.

Тут метод ExecuteNonOuery () повертає число змінених рядків (в даному випадку доданих в таблицю об'єктів). Хоча нам необов'язково повертати результат методу, але даний результат може використовуватися в якості перевірки, що операція, зокрема, додавання, пройшла успішно.

Оновлення об'єктів

Оновлення буде відбуватися аналогічно, тільки тепер буде використовуватися sql-вираз UPDATE, яке має наступний синтаксис:

UPDATE названіе\_табліци

SET столбец1 = значення1, столбец2 = значення2, столбецN = значеніеN

WHERE некоторий\_столбец = некоторое\_значеніе

Застосуємо цей вираз:

**string connectionString = @ "Data Source =. \ SQLEXPRESS; Initial Catalog = usersdb; Integrated Security = True";**

**string sqlExpression = "UPDATE Users SET Age = 20 WHERE Name = 'Tom'";**

**using (SqlConnection connection = new SqlConnection (connectionString))**

**{**

**connection.Open ();**

**SqlCommand command = new SqlCommand (sqlExpression, connection);**

**int number = command.ExecuteNonQuery ();**

**Console.WriteLine ( "Оновлене об'єктів: {0}", number);**

**}**

Тут оновлюється рядок, в якій Name = Tom, тобто вище доданий об'єкт. Якщо в таблиці буде кілька рядків, у яких Name = Tom, то оновляться всі ці рядки.

Видалення

Видалення проводиться за допомогою sql-вирази DELETE, яке має наступний синтаксис:

DELETE FROM таблиця

WHERE стовпець = значення

Видалимо, наприклад, всіх користувачів, у яких ім'я Tom:

**string connectionString = @ "Data Source =. \ SQLEXPRESS; Initial Catalog = usersdb; Integrated Security = True";**

**string sqlExpression = "DELETE FROM Users WHERE Name = 'Tom'";**

**using (SqlConnection connection = new SqlConnection (connectionString))**

**{**

**connection.Open ();**

**SqlCommand command = new SqlCommand (sqlExpression, connection);**

**int number = command.ExecuteNonQuery ();**

**Console.WriteLine ( "Вилучено об'єктів: {0}", number);**

**}**

У всіх трьох випадках фактично змінюється тільки sql-вираз, а інша логіка залишається незмінною. І ми також можемо виконувати відразу кілька операцій:

**static void Main (string [] args)**

**{**

**string connectionString = @ "Data Source =. \ SQLEXPRESS; Initial Catalog = usersdb; Integrated Security = True";**

**Console.WriteLine ( "Введіть ім'я:");**

**string name = Console.ReadLine ();**

**Console.WriteLine ( "Введіть вік:");**

**int age = Int32.Parse (Console.ReadLine ());**

**string sqlExpression = String.Format ( "INSERT INTO Users (Name, Age) VALUES ( '{0}', {1})", name, age);**

**using (SqlConnection connection = new SqlConnection (connectionString))**

**{**

**connection.Open ();**

**// додавання**

**SqlCommand command = new SqlCommand (sqlExpression, connection);**

**int number = command.ExecuteNonQuery ();**

**Console.WriteLine ( "Додано об'єктів: {0}", number);**

**// оновлення раніше доданого об'єкта**

**Console.WriteLine ( "Введіть нове ім'я:");**

**name = Console.ReadLine ();**

**sqlExpression = String.Format ( "UPDATE Users SET Name = '{0}' WHERE Age = {1}", name, age);**

**command.CommandText = sqlExpression;**

**number = command.ExecuteNonQuery ();**

**Console.WriteLine ( "Оновлене об'єктів: {0}", number);**

**}**

**Console.Read ();**

**}**

## Читання результатів запиту і SqlDataReader

У минулому пункті ми розглянули, як виконувати команди за допомогою методу ExecuteNonOuery (), однак якщо ми хочемо зчитувати дані, які зберігаються в таблиці, то нам буде потрібно інший метод - ExecuteReader (). Цей метод повертає об'єкт SqlDataReader, який використовується для читання даних. Так, отримаємо всі дані з таблиці Users і виведемо їх на консоль:

**static void Main (string [] args)**

**{**

**string connectionString = @ "Data Source =. \ SQLEXPRESS; Initial Catalog = usersdb; Integrated Security = True";**

**string sqlExpression = "SELECT \* FROM Users";**

**using (SqlConnection connection = new SqlConnection (connectionString))**

**{**

**connection.Open ();**

**SqlCommand command = new SqlCommand (sqlExpression, connection);**

**SqlDataReader reader = command.ExecuteReader ();**

**if (reader.HasRows) // якщо є дані**

**{**

**// виводимо назви стовпців**

**Console.WriteLine ( "{0} \ t {1} \ t {2}", reader.GetName (0), reader.GetName (1), reader.GetName (2));**

**while (reader.Read ()) // підрядник зчитуємо дані**

**{**

**object id = reader.GetValue (0);**

**object name = reader.GetValue (1);**

**object age = reader.GetValue (2);**

**Console.WriteLine ( "{0} \ t {1} \ t {2}", id, name, age);**

**}**

**}**

**reader.Close ();**

**}**

**Console.Read ();**

}

Для вибірки даних з БД використовується sql-вираз SELECT. В даному випадку ми вибираємо всі стовпці всіх рядків таблиці. Отримавши при виконанні запиту об'єкт SqlDataReader, ми можемо вважати всі отримані дані.

Але спочатку ми перевіряємо, а чи є взагалі дані за допомогою властивості HasRows. Якщо дані є, то виводимо заголовки таблиці за допомогою методів reader.GetName (). Причому ми отримуємо стовпці в вибірці саме в тому порядку, в якому вони визначені в таблиці. Тобто якщо другим в таблиці йде стовпець "Name", то щоб отримати його застосовується метод GetName (1) (так як нумерація стовпців йде з нуля).

Далі зчитуємо самі дані. За допомогою методу reader.Read () рідер переходить до наступного рядка і повертає логічне значення, яке вказує, чи є дані для зчитування.

У циклі while (reader.Read ()) в порядку проходження стовпів отримуємо дані за допомогою методу GetValue (), який повертає дані у вигляді об'єкта типу object. Наприклад, стовпець Id йде першим і представляє ціле число, тому для його отримання застосовується метод reader.GetValue (0). А стовпець Name йде другим, тому його значення отримуємо за допомогою reader.GetValue (1).

Після завершення роботи з SqlDataReader треба його закрити методом Close (). І поки один SqlDataReader не закритий, інший об'єкт SqlDataReader для одного і того ж підключення ми використовувати не зможемо.

В якості альтернативи ми могли б звертатися до даних через назву параметра:

while (reader.Read ())

{

object id = reader [ "id"];

object name = reader [ "name"];

object age = reader [ "age"];

Console.WriteLine ( "{0} \ t {1} \ t {2}", id, name, age);

}

В цьому випадку результат буде аналогічним.

Асинхронне читання

Для асинхронного читання, по-перше, застосовується метод ExecuteReaderAsync () класу SqlCommand, і по-друге, метод ReadAsync () класу SqlDataReader:

**static void Main (string [] args)**

**{**

**ReadDataAsync (). GetAwaiter ();**

**Console.Read ();**

**}**

**private static async Task ReadDataAsync ()**

**{**

**string connectionString = @ "Data Source =. \ SQLEXPRESS; Initial Catalog = usersdb; Integrated Security = True";**

**string sqlExpression = "SELECT \* FROM Users";**

**using (SqlConnection connection = new SqlConnection (connectionString))**

**{**

**await connection.OpenAsync ();**

**SqlCommand command = new SqlCommand (sqlExpression, connection);**

**SqlDataReader reader = await command.ExecuteReaderAsync ();**

**if (reader.HasRows)**

**{**

**// виводимо назви стовпців**

**Console.WriteLine ( "{0} \ t {1} \ t {2}", reader.GetName (0), reader.GetName (1), reader.GetName (2));**

**while (await reader.ReadAsync ())**

**{**

**object id = reader.GetValue (0);**

**object name = reader.GetValue (1);**

**object age = reader.GetValue (2);**

**Console.WriteLine ( "{0} \ t {1} \ t {2}", id, name, age);**

**}**

**}**

**reader.Close ();**

**}**

**}**

## Отримання скалярних значень

При відправці запитів ми можемо використовувати спеціальні вбудовані функції SQL, наприклад, Min, Max, Sum, Count і т.д., які не виконують операції з об'єктами і не витягують об'єкти, а повертають якесь певне значення. Наприклад, функція Count підраховує кількість об'єктів. І для роботи з такими функціями в SqlCommand визначений спеціальний метод ExecuteScalar. Наприклад, знайдемо число всіх об'єктів в таблиці і отримаємо мінімальний вік користувачів:

**string connectionString = @ "Data Source =. \ SQLEXPRESS; Initial Catalog = usersdb; Integrated Security = True";**

**string sqlExpression = "SELECT COUNT (\*) FROM Users";**

**using (SqlConnection connection = new SqlConnection (connectionString))**

**{**

**connection.Open ();**

**SqlCommand command = new SqlCommand (sqlExpression, connection);**

**object count = command.ExecuteScalar ();**

**command.CommandText = "SELECT MIN (Age) FROM Users";**

**object minAge = command.ExecuteScalar ();**

**Console.WriteLine ( "У таблиці {0} об'єктів", count);**

**Console.WriteLine ( "Мінімальний вік: {0}", minAge);**

**}**

Вираз "SELECT COUNT (\*) FROM Users" кількість об'єктів в таблиці Users, а вираз "SELECT MIN (Age) FROM Users" знаходить мінімальне значення стовпця Age. Як результат метод ExecuteScalar () повертає об'єкт типу object.

І якщо ми захочемо отримувати дані асинхронним чином, то ми можемо використовувати асинхронну версію цього методу - ExecuteScalarAsync ().

**ЗАВДАННЯ**

1. Базу даних, яку Ви розробили для проекту вбудувати в консольний застосунок.
2. Написати 4 запити (завдання для кожного студента – окремий запит), за допомогою засобів ADO.NET

**Лабораторна робота №15**

**Тема:LINQ**

**План**

1. Основи LINQ

2.Методи розширення LINQ

## 3.Фільтрація вибірки и проекція

## 4. Сортування

## 5. Робота с множинами

## 6.Агрегатні операції

## 7.Групування

## 8. З‘єднання колекцій. Метод Join, GroupJoin и Zip

## 9. Методы All и Any

**1.Основи LINQ**

LINQ (Language-Integrated Query) представляє просту і зручну мову запитів до джерела даних. Як джерело даних може виступати об'єкт, який реалізує інтерфейс IEnumerable (наприклад, стандартні колекції, масиви), набір даних DataSet, документ XML. Але незалежно від типу джерела LINQ дозволяє застосувати до всіх один і той же підхід для вибірки даних.

Існує кілька різновидів LINQ:

LINQ to Objects: застосовується для роботи з масивами і колекціями

LINQ to Entities: використовується при зверненні до баз даних через технологію Entity Framework

LINQ to Sql: технологія доступу до даних в MS SQL Server

LINQ to XML: застосовується при роботі з файлами XML

LINQ to DataSet: застосовується при роботі з об'єктом DataSet

Parallel LINQ (PLINQ): використовується для виконання паралельної запитів

У чому ж зручність LINQ? Подивимося на простому прикладі. Виберемо з масиву рядки, що починаються на певну букву і відсортуємо отриманий список:

**string [] teams = { "Баварія", "Боруссія", "Реал Мадрид", "Манчестер Сіті", "ПСЖ", "Барселона"};**

**var selectedTeams = new List <string> ();**

**foreach (string s in teams)**

**{**

**if (s.ToUpper (). StartsWith ( "Б"))**

**selectedTeams.Add (s);**

**}**

**selectedTeams.Sort ();**

**foreach (string s in selectedTeams)**

**Console.WriteLine (s);**

Тепер проведемо ті ж дії за допомогою LINQ:

**string [] teams = { "Баварія", "Боруссія", "Реал Мадрид", "Манчестер Сіті", "ПСЖ", "Барселона"};**

**var selectedTeams = from t in teams // визначаємо кожен об'єкт з teams як t**

**where t.ToUpper (). StartsWith ( "Б") // фільтрація за критерієм**

**orderby t // упорядковуємо по зростанню**

**select t; // вибираємо об'єкт**

**foreach (string s in selectedTeams)**

**Console.WriteLine (s);**

Щоб використовувати функціональність LINQ, переконаємося, що в файлі підключено простір імен System.LINQ.

Отже, код став менше і простіше. В принципі все вираз можна було б записати в один рядок: var selectedTeams = from t in teams where t.ToUpper (). StartsWith ( "Б") orderby t select t.

Найпростіше визначення запиту LINQ виглядає наступним чином:

from змінна in набор\_об'ектів

select змінна;

Отже, що робить цей запит LINQ? Вираз from t in teams проходить по всіх елементах масиву teams і визначає кожен елемент як t. Використовуючи змінну t ми можемо проводити над нею різні операції.

Незважаючи на те, що ми не вказуємо тип змінної t, вираження LINQ є строго типізованим. Тобто середовище автоматично розпізнає, що набір teams складається з об'єктів string, тому змінна t буде розглядатися як ще один рядок.

Далі за допомогою оператора where проводиться фільтрація об'єктів, і якщо об'єкт відповідає критерію (в даному випадку початкова буква повинна бути "Б"), то цей об'єкт передається далі.

Оператор orderby впорядковує по зростанню, тобто сортує вибрані об'єкти.

Оператор select передає обрані значення в результуючу вибірку, яка повертається LINQ-виразом.

В даному випадку результатом вираження LINQ є об'єкт IEnumerable <T>. Нерідко результуюча вибірка визначається за допомогою ключового слова var, тоді компілятор на етапі компіляції сам виводить тип.

Перевагою подібних запитів також є і те, що вони інтуїтивно схожі на запити мови SQL, хоча і мають деякі відмінності.

**2.Методи розширення LINQ**

Крім стандартного синтаксису from .. in .. select для створення запиту LINQ ми можемо застосовувати спеціальні методи розширення, які визначені для інтерфейсу IEnumerable. Як правило, ці методи реалізують ту ж функціональність, що і оператори LINQ типу where або orderby.

наприклад:

**string [] teams = { "Баварія", "Боруссія", "Реал Мадрид", "Манчестер Сіті", "ПСЖ", "Барселона"};**

**var selectedTeams = teams.Where (t => t.ToUpper (). StartsWith ( "Б")). OrderBy (t => t);**

**foreach (string s in selectedTeams)**

**Console.WriteLine (s);**

Запит teams.Where (t => t.ToUpper (). StartsWith ( "Б")). OrderBy (t => t) буде аналогічний попередньому. Він складається з ланцюжка методів Where і OrderBy. Як аргумент ці методи приймають делегат або лямбда-вираз.

Чи не кожен метод розширення має аналог серед операторів LINQ, але в цьому випадку можна поєднувати обидва підходи. Наприклад, використовуємо стандартний синтаксис linq і метод розширення Count (), який повертає кількість елементів у вибірці:

int number = (from t in teams where t.ToUpper (). StartsWith ( "Б") select t) .Count ();

Список використаних методів розширення LINQ

Select: визначає проекцію обраних значень

Where: визначає фільтр вибірки

OrderBy: впорядковує елементи по зростанню

OrderByDescending: упорядковуэ елементи по спаданню.

## 3.Фільтрація вибірки и проекція

Для вибору елементів з деякого набору за умовою використовується метод Where. Наприклад, виберемо всі парні елементи, які більше 10.

Фільтрація за допомогою операторів LINQ:

**int [] numbers = {1, 2, 3, 4, 10, 34, 55, 66, 77, 88};**

**IEnumerable <int> evens = from i in numbers**

**where i% 2 == 0 && i> 10**

**select i;**

**foreach (int i in evens)**

**Console.WriteLine (i);**

Тут використовується конструкція from: from i in numbers

Той же запит за допомогою методу розширення:

**int [] numbers = {1, 2, 3, 4, 10, 34, 55, 66, 77, 88};**

**IEnumerable <int> evens = numbers.Where (i => i% 2 == 0 && i> 10);**

Якщо вираз в методі Where для певного елемента дорівнюватиме true (в даному випадку вираз i% 2 == 0 && i> 10), то даний елемент потрапляє в результуючу вибірку.

## 4. Сортування

Для сортування набору даних по зростанню використовується оператор orderby:

**int [] numbers = {3, 12, 4, 10, 34, 20, 55, -66, 77, 88, 4};**

**var orderedNumbers = from i in numbers**

**orderby i**

**select i;**

**foreach (int i in orderedNumbers)**

**Console.WriteLine (i);**

Оператор orderby приймає критерій сортування. В даному випадку в якості критерію виступає саме число.

## 5. Робота с множинами

Крім методів вибірки LINQ має кілька методів для роботи з множинами: різниця, об'єднання і перетин.

Різниця множин

За допомогою методу Except можна отримати різницю двох множин:

**string [] soft = { "Microsoft", "Google", "Apple"};**

**string [] hard = { "Apple", "IBM", "Samsung"};**

**// різницю множин**

**var result = soft.Except (hard);**

**foreach (string s in result)**

**Console.WriteLine (s);**

В даному випадку з масиву soft забираються всі елементи, які є в масиві hard.

Перетин множин

Для отримання перетину множин, тобто загальних для обох наборів елементів, застосовується метод Intersect:

**string [] soft = { "Microsoft", "Google", "Apple"};**

**string [] hard = { "Apple", "IBM", "Samsung"};**

**// перетин множин**

**var result = soft.Intersect (hard);**

**foreach (string s in result)**

**Console.WriteLine (s);**

Так як обидва набору мають тільки один спільний елемент, то відповідно тільки він і потрапить в результуючу вибірку:

Об'єднання множин

Для об'єднання двох множин використовується метод Union. Його результатом є новий набір, в якому присутні елементи, як з одного, так і з другої множини. Повторювані елементи додаються в результат тільки один раз:

**string [] soft = { "Microsoft", "Google", "Apple"};**

**string [] hard = { "Apple", "IBM", "Samsung"};**

**// об'єднання множин**

**var result = soft.Union (hard);**

**foreach (string s in result)**

**Console.WriteLine (s);**

Якщо ж нам потрібно просте об'єднання двох наборів, то ми можемо використовувати метод Concat:

**var result = soft.Concat (hard);**

Видалення дублікатів

Для видалення дублів в наборі використовується метод Distinct:

**var result = soft.Concat (hard) .Distinct ();**

## 6.Агрегатні операції

До агрегатних операцій відносять різні операції над вибіркою, наприклад, отримання числа елементів, отримання мінімального, максимального і середнього значення у вибірці, а також підсумовування значень.

метод Aggregate

Метод Aggregate виконує загальну агрегацію елементів колекції в залежності від зазначеного вирази. наприклад:

**int [] numbers = {1, 2, 3, 4, 5};**

**int query = numbers.Aggregate ((x, y) => x - y);**

Змінна query представлятиме результат агрегації масиву. Як умова агрегації використовується вираз (x, y) => x - y, тобто спочатку з першого елемента віднімається другий, потім з отриманого значення віднімається третій і так далі. Тобто:

int query = 1 - 2 - 3 - 4 - 5

У підсумку ми отримаємо число -13. Відповідно ми б могли використовувати будь-які інші операції, наприклад, складання:

**int query = numbers.Aggregate ((x, y) => x + y); // аналогічно 1 + 2 + 3 + 4 + 5**

Отримання розміру вибірки. метод Count

Для отримання числа елементів у вибірці використовується метод Count ():

**int [] numbers = {1, 2, 3, 4, 10, 34, 55, 66, 77, 88};**

**int size = (from i in numbers where i% 2 == 0 && i> 10 select i) .Count ();**

**Console.WriteLine (size);**

Метод Count () в одній з версій також може приймати лямбда-вираз, який встановлює умова вибірки. Тому ми можемо в даному випадку не використовувати вираз Where:

**int size = numbers.Count (i => i% 2 == 0 && i> 10);**

**Console.WriteLine (size);**

Отримання суми

Для отримання суми значень застосовується метод Sum:

**int [] numbers = {1, 2, 3, 4, 10, 34, 55, 66, 77, 88};**

**List <User> users = new List <User> ()**

**{**

**new User {Name = "Tom", Age = 23},**

**new User {Name = "Sam", Age = 43},**

**new User {Name = "Bill", Age = 35}**

**};**

**int sum1 = numbers.Sum ();**

**decimal sum2 = users.Sum (n => n.Age);**

Максимальний, мінімальний і середній значення

Для знаходження мінімального значення застосовується метод Min (), для отримання максимального - метод Max (), а для знаходження середнього значення - метод Average (). Їх дія схожа на методи Sum і Count:

**int [] numbers = {1, 2, 3, 4, 10, 34, 55, 66, 77, 88};**

**List <User> users = new List <User> ()**

**{**

**new User {Name = "Tom", Age = 23},**

**new User {Name = "Sam", Age = 43},**

**new User {Name = "Bill", Age = 35}**

**};**

**int min1 = numbers.Min ();**

**int min2 = users.Min (n => n.Age); // мінімальний вік**

**int max1 = numbers.Max ();**

**int max2 = users.Max (n => n.Age); // максимальний вік**

**double avr1 = numbers.Average ();**

**double avr2 = users.Average (n => n.Age); //середній вік**

## Методи Skip и Take

Метод Skip () пропускає певну кількість елементів, а метод Take () витягує певне число елементів. Нерідко ці методи застосовуються разом для створення посторінкового виведення.

Винесемо три перших елемента:

**int [] numbers = {-3, -2, -1, 0, 1, 2, 3};**

**var result = numbers.Take (3);**

**foreach (int i in result)**

**Console.WriteLine (i);**

Виберемо всі елементи, крім перших трьох:

**var result = numbers.Skip (3);**

Схожим чином працюють методи TakeWhile і SkipWhile.

## 7.Групування

Для групування даних за певними параметрами застосовується оператор group by або метод GroupBy (). Припустимо, у нас є набір з об'єктів наступного типу:

**class Phone**

**{**

**public string Name {get; set; }**

**public string Company {get; set; }**

**}**

Даний клас представляє модель телефону, в якій визначені властивості для назви і компанії-виробника. Згрупуємо набір телефонів по виробнику:

**List <Phone> phones = new List <Phone>**

**{**

**new Phone {Name = "Lumia 430", Company = "Microsoft"},**

**new Phone {Name = "Mi 5", Company = "Xiaomi"},**

**new Phone {Name = "LG G 3", Company = "LG"},**

**new Phone {Name = "iPhone 5", Company = "Apple"},**

**new Phone {Name = "Lumia 930", Company = "Microsoft"},**

**new Phone {Name = "iPhone 6", Company = "Apple"},**

**new Phone {Name = "Lumia 630", Company = "Microsoft"},**

**new Phone {Name = "LG G 4", Company = "LG"}**

**};**

**var phoneGroups = from phone in phones**

**group phone by phone.Company;**

**foreach (IGrouping <string, Phone> g in phoneGroups)**

**{**

**Console.WriteLine (g.Key);**

**foreach (var t in g)**

**Console.WriteLine (t.Name);**

**Console.WriteLine ();**

**}**

Якщо у виразі LINQ останнім оператором, який виконує операції над вибіркою, є group, то оператор select не застосовується.

Оператор group приймає критерій за яким проводиться угруповання: group phone by phone.Company - в даному випадку угруповання по властивості Company.

Результатом оператора group є вибірка, яка складається з груп. Кожна група представляє об'єкт IGrouping <string, Phone>: параметр string вказує на тип ключа, а параметр Phone - на тип згрупованих об'єктів.

Кожна група має ключ, який ми можемо отримати через властивість Key: g.Key

Всі елементи групи можна отримати за допомогою додаткової ітерації. Елементи групи мають той же тип, що і тип об'єктів, які передавалися оператору group, тобто в даному випадку об'єкти типу Phone.

Аналогічний запит можна побудувати за допомогою методу розширення GroupBy.

## 8. З‘єднання колекцій. Метод Join, GroupJoin и Zip

З'єднання в LINQ використовується для об'єднання двох різнотипних наборів в один. Для з'єднання використовується оператор join або метод Join (). Як правило, дана операція застосовується до двох наборів, які мають один загальний критерій. Наприклад, у нас є два класи:

**class Player**

**{**

**public string Name {get; set; }**

**public string Team {get; set; }**

**}**

**class Team**

**{**

**public string Name {get; set; }**

**public string Country {get; set; }**

**}**

Об'єкти обох класів матиме один загальний критерій - назва команди. З'єднаємо за цим критерієм два набори цих класів:

**List <Team> teams = new List <Team> ()**

**{**

**new Team {Name = "Баварія", Country = "Німеччина"},**

**new Team {Name = "Барселона", Country = "Іспанія"}**

**};**

**List <Player> players = new List <Player> ()**

**{**

**new Player {Name = "Мессі", Team = "Барселона"},**

**new Player {Name = "Неймар", Team = "Барселона"},**

**new Player {Name = "Роббен", Team = "Баварія"}**

**};**

**var result = from pl in players**

**join t in teams on pl.Team equals t.Name**

**select new {Name = pl.Name, Team = pl.Team, Country = t.Country};**

**foreach (var item in result)**

**Console.WriteLine ( "{0} - {1} ({2})", item.Name, item.Team, item.Country);**

За допомогою виразу join t in teams on pl.Team equals t.Name об'єкт pl зі списку players з'єднується з об'єктом t зі списку teams, якщо значення властивості pl.Team збігається зі значенням властивості t.Name. Результатом з'єднання буде об'єкт анонімного типу, який буде містити три властивості.

Те ж саме діяння можна було б виконати за допомогою методу Join ():

**var result = players.Join (teams, // другий набір**

**p => p.Team, // властивість-селектор об'єкта з першого набору**

**t => t.Name, // властивість-селектор об'єкта з другого набору**

**(P, t) => new {Name = p.Name, Team = p.Team, Country = t.Country}); // результат**

Метод Join () приймає чотири параметри:

другий список, який з'єднуємо з поточним

властивість об'єкта з поточного списку, за яким йде з'єднання

властивість об'єкта з другого списку, за яким йде з'єднання

новий об'єкт, який виходить в результаті з'єднання

GroupJoin

Метод GroupJoin крім з'єднання послідовностей також виконує і угруповання. Наприклад, візьмемо вишеопределенние списки teams і players та згрупуємо всіх гравців по командам:

**var result2 = teams.GroupJoin (**

**players, // другий набір**

**t => t.Name, // властивість-селектор об'єкта з першого набору**

**pl => pl.Team, // властивість-селектор об'єкта з другого набору**

**(Team, pls) => new // результуючий об'єкт**

**{**

**Name = team.Name,**

**Country = team.Country,**

**Players = pls.Select (p => p.Name)**

**});**

**foreach (var team in result2)**

**{**

**Console.WriteLine (team.Name);**

**foreach (string player in team.Players)**

**{**

**Console.WriteLine (player);**

**}**

**Console.WriteLine ();**

**}**

Метод GroupJoin, також як і метод Join, приймає все ті ж параметри. Тільки тепер у останній параметр - делегат передаються об'єкт команди і набір гравців цієї команди.

Метод Zip

Метод Zip дозволяє поєднувати дві послідовності таким чином, що перший елемент з першої послідовності об'єднується з першим елементом з другої послідовності, другий елемент з першої послідовності з'єднується з другим елементом з другої послідовності і так далі:

**List <Team> teams = new List <Team> ()**

**{**

**new Team {Name = "Баварія", Country = "Німеччина"},**

**new Team {Name = "Барселона", Country = "Іспанія"},**

**new Team {Name = "Ювентус", Country = "Італія"}**

**};**

**List <Player> players = new List <Player> ()**

**{**

**new Player {Name = "Роббен", Team = "Баварія"},**

**new Player {Name = "Неймар", Team = "Барселона"},**

**new Player {Name = "Буффон", Team = "Ювентус"}**

**};**

**var result2 = players.Zip (teams,**

**(Player, team) => new**

**{**

**Name = player.Name,**

**Team = team.Name, Country = team.Country**

**});**

**foreach (var player in result2)**

**{**

**Console.WriteLine ( "{0} - {1} ({2})", player.Name, player.Team, player.Country);**

**Console.WriteLine ();**

**}**

Метод Zip в якості першого параметра приймає другу послідовність, з якою треба єднатися, а в якості другого параметра - делегат для створення нового об'єкта.

## 9. Методы All и Any

Методи All, Any і Contains дозволяють визначити, чи відповідає колекція певній умові, і в залежності від результату вони повертають true або false.

Метод All перевіряє, чи відповідають всі елементи умові. Наприклад, дізнаємося, чи у всіх користувачів вік перевищує 20 і ім'я починається з букви T:

**List <User> users = new List <User> ()**

**{**

**new User {Name = "Tom", Age = 23},**

**new User {Name = "Sam", Age = 43},**

**new User {Name = "Bill", Age = 35}**

**};**

**bool result1 = users.All (u => u.Age> 20); // true**

**if (result1)**

**Console.WriteLine ( "У всіх користувачів вік більше 20");**

**else**

**Console.WriteLine ( "Є користувачі з віком менше 20");**

**bool result2 = users.All (u => u.Name.StartsWith ( "T")); // false**

**if (result2)**

**Console.WriteLine ( "У всіх користувачів ім'я починається з T");**

**else**

**Console.WriteLine ( "Не у всіх користувачів ім'я починається з T");**

Оскільки у всіх користувачів вік більше 20, то змінна result1 буде дорівнює true. У той же час не у всіх користувачів ім'я начінаяется з літери T, тому друга змінна result2 буде дорівнює false.

Метод Any діє подібним чином, тільки дозволяє дізнатися, чи відповідає хоча б один елемент колекції певній умові:

**bool result1 = users.Any (u => u.Age <20); // false**

**if (result1)**

**Console.WriteLine ( "Є користувачі з віком менше 20");**

**else**

**Console.WriteLine ( "У всіх користувачів вік більше 20");**

**bool result2 = users.Any (u => u.Name.StartsWith ( "T")); // true**

**if (result2)**

**Console.WriteLine ( "Є користувачі, у яких ім'я починається з T");**

**else**

**Console.WriteLine ( "Відсутні користувачі, у яких ім'я починається з T");**

## Відкладене і негайне виконання запитів LINQ

Є два способи виконання запиту LINQ: відкладене і негайне виконання.

При відкладеному виконанні LINQ-вираз не виконується, поки не буде проведена ітерація або перебір по вибірці. Розглянемо відкладене виконання:

**string [] teams = { "Баварія", "Боруссія", "Реал Мадрид", "Манчестер Сіті", "ПСЖ", "Барселона"};**

**var selectedTeams = from t in teams where t.ToUpper (). StartsWith ( "Б") orderby t select t;**

**// виконання LINQ-запиту**

**foreach (string s in selectedTeams)**

**Console.WriteLine (s);**

Тобто фактичне виконання запиту відбувається не в рядку визначення: var selectedTeams = from t ..., а при переборі в циклі foreach.

Фактично LINQ-запит розбивається на три етапи:

Отримання джерела даних

створення запиту

Виконання запиту і отримання його результатів

Негайне виконання запиту

За допомогою ряду методів ми можемо застосувати негайне виконання запиту. Це методи, які повертають одне атомарному значення або один елемент. Наприклад, Count (), Average (), First () / FirstOrDefault (), Min (), Max () і т.д. Наприклад, метод Count () повертає числове значення, яке представляє кількість елементів в отриманої послідовності. А метод First () повертає перший елемент послідовності. Але щоб виконати ці методи, спочатку треба отримати саму послідовність, тобто результат запиту, і пройтися по ній циклом foreach, який викликається неявно всередині структури запиту.

Розглянемо приклад з методом Count (), який повертає число елементів послідовності:

**string [] teams = { "Баварія", "Боруссія", "Реал Мадрид", "Манчестер Сіті", "ПСЖ", "Барселона"};**

**// визначення і виконання LINQ-запиту**

**int i = (from t in teams**

**where t.ToUpper (). StartsWith ( "Б")**

**orderby t select t) .Count ();**

**Console.WriteLine (i); // 3**

**teams [1] = "Ювентус";**

**Console.WriteLine (i); // 3**

Результатом методу Count буде об'єкт int, тому спрацює негайне виконання.

**ЗАВДАННЯ.**

1. Для впудовваної в проект БД написати 5 LINQ-запитів. (Кожен студент окремо).

# Лабораторна робота №16

# Стандартні інтерфейси C#

**Мета роботи:**

1. Вивчення використання в програмах стандартних інтерфейсів

**Методичні рекомендації до виконання лабораторної роботи**

1. Уважно прочитайте лекцію 13 та цю лабораторну роботу.

2. Виконайте і проаналізуйте всі наведені в лекції 13 приклади.

3. Для поглибленого вивчення цього матеріалу прочитайте розділи 5, 10 книги [2].

**2. Варіанти завдань для Лабораторної роботи № 16**

**Номер варіанта відповідає номеру прізвища студента у списку групи**

|  |  |
| --- | --- |
| **№**  **варіан-ту** | **Зміст завдання** |
| 1 | 1. Створіть масив об'єктів класу **Тварина** (або похідних). Реалізуйте інтерфейс *IComparable* для порівняння тварин за вагою в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння тварин не тільки за вагою, але і за зростом.  3. Реалізуйте інтерфейс *IEnumerable. В*иведіть на консоль список тварин, впорядкований за вагою. |
| 2 | 1. Створіть масив об'єктів класу **Робітник**. Реалізуйте інтерфейс *IComparable* для порівняння робітників за віком в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння робітників не тільки за віком, але і за зарплатою.  3. Реалізуйте інтерфейс *IEnumerable. В*иведіть на консоль список робітників , впорядкований за зарплатою. |
| 3 | 1. Створіть масив об'єктів класу **Співробітник**. Реалізуйте інтерфейс *IComparable* для порівняння співробітників за віком в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння співробітників не тільки за віком, але і за стажем роботи на цьому підприємстві.  3. Реалізуйте інтерфейс *IEnumerable. В*иведіть на консоль список співробітників, впорядкований за стажем роботи.  . |
| 4 | 1. Створіть масив об'єктів класу **Виріб**. Реалізуйте інтерфейс *IComparable* для порівняння виробів за вагою в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння виробів за ціною і за якістю (створити свою шкалу якості).  3. Реалізуйте інтерфейс *IEnumerable. В*иведіть на консоль список виробів, впорядкований за ціною. |
| 5 | 1. Створіть масив об'єктів класу **Організація**. Реалізуйте інтерфейс *IComparable* для порівняння організацій за кількістю співробітників в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння організацій за кількістю співробітників і за рейтингом успішності (створити свою шкалу у 100 балів).  3. Реалізуйте інтерфейс *IEnumerable.* Виведіть на консоль список організацій , впорядкований за рейтингом. |
| 6 | 1. Створіть масив об'єктів класу **Журнал**. Реалізуйте інтерфейс *IComparable* для порівняння журналів за ціною в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння журналів за кількістю сторінок і за рейтингом продажів (створити свою шкалу 10 балів).  3. Реалізуйте інтерфейс *IEnumerable.* Виведіть на консоль список журналів, впорядкований за рейтингом продажів. |
| 7 | 1. Створіть масив об'єктів класу **Дерево**. Реалізуйте інтерфейс *IComparable* для порівняння порід дерев за ціною в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння дерев за висотою і за ціною.  3. Реалізуйте інтерфейс *IEnumerable.* Виведіть на консоль список порід дерев, впорядкований за ціною. |
| 8 | 1. Створіть масив об'єктів класу **Місто**. Реалізуйте інтерфейс *IComparable* для порівняння міст за територією в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння міст за територією і за кількістю населення.  3. Реалізуйте інтерфейс *IEnumerable.* Виведіть на консоль список міст, впорядкований за кількістю населення. |
| 9 | 1. Створіть масив об'єктів класу **Товар**. Реалізуйте інтерфейс *IComparable* для порівняння товарів за ціною в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння товарів за ціною і за розмірами.  3. Реалізуйте інтерфейс *IEnumerable.* Виведіть на консоль список товарів, впорядкований за ціною. |
| 10 | 1. Створіть масив об'єктів класу **Документ**. Реалізуйте інтерфейс *IComparable* для порівняння документів за кількістю сторінок в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння документів за кількістю сторінок і за таємністю (наприклад, 1- цілком таємно, 2 – таємно, 3 – для службового використання, 4 – для загального доступу).  3. Реалізуйте інтерфейс *IEnumerable.* Виведіть на консоль список документів, впорядкований за кількістю сторінок. |
| 11 | 1. Створіть масив об'єктів класу **Автомобіль**. Реалізуйте інтерфейс *IComparable* для порівняння автомобілів за ціною в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння автомобілів за ціною і за потужністю.  3. Реалізуйте інтерфейс *IEnumerable.* Виведіть на консоль список автомобілів, впорядкований за ціною і потужністю. |
| 12 | 1. Створіть масив об'єктів класу **Викладач**. Реалізуйте інтерфейс *IComparable* для порівняння викладачів за зарплатою в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння викладачів за зарплатою і за стажем роботи.  3. Реалізуйте інтерфейс *IEnumerable.* Виведіть на консоль список викладачів, впорядкований за за зарплатою і за стажем роботи. |
| 13 | 1. Створіть масив об'єктів класу **Овочі**. Реалізуйте інтерфейс *IComparable* для порівняння овочів за ціною в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння викладачів за зарплатою і за стажем роботи.  3. Реалізуйте інтерфейс *IEnumerable.* Виведіть на консоль список овочів, впорядкований за ціною. |
| 14 | 1. Створіть масив об'єктів класу **Книга**. Реалізуйте інтерфейс *IComparable* для порівняння книг за ціною в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння книг за ціною і за кількістю сторінок.  3. Реалізуйте інтерфейс *IEnumerable.* Виведіть на консоль список книг, впорядкований за ціною. |
| 15 | 1. Створіть масив об'єктів класу **Меблі**. Реалізуйте інтерфейс *IComparable* для порівняння **Меблів** за ціною в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння меблів за ціною і за габаритами.  3. Реалізуйте інтерфейс *IEnumerable.* Виведіть на консоль список меблів, впорядкований за ціною. |
| 16 | 1. Створіть масив об'єктів класу **Будинок**. Реалізуйте інтерфейс *IComparable* для порівняння будинків за ціною в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння будинків за ціною і за площею.  3. Реалізуйте інтерфейс *IEnumerable.* Виведіть на консоль список будинків, впорядкований спочатку за ціною, потім за площею. |
| 17 | 1. Створіть масив об'єктів класу **Літак**. Реалізуйте інтерфейс *IComparable* для порівняння літаків за кількістю годин нальоту в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння літаків за кількістю годин нальоту і за надійністю.  3. Реалізуйте інтерфейс *IEnumerable.* Виведіть на консоль список літаків, впорядкований спочатку за кількістю годин нальоту, потім за надійністю. |
| 18 | 1. Створіть масив об'єктів класу **Операційна система (ОС)**. Реалізуйте інтерфейс *IComparable* для порівняння ОС за ціною в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння ОС за ціною і за роками виходу у продаж.  3. Реалізуйте інтерфейс *IEnumerable.* Виведіть на консоль список ОС, впорядкований спочатку за ціною, потім за роками виходу у продаж. |
| 19 | 1. Створіть масив об'єктів класу **Комп'ютер**. Реалізуйте інтерфейс *IComparable* для порівняння Комп'ютерів за ціною в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння комп'ютерів за ціною і за розмірами жорсткого диска.  3. Реалізуйте інтерфейс *IEnumerable.* Виведіть на консоль список комп'ютерів, впорядкований спочатку за ціною, потім за розмірами жорсткого диска. |
| 20 | 1. Створіть масив об'єктів класу **Збірка**. Реалізуйте інтерфейс *IComparable* для порівняння збірок за ціною в методі *CompareTo().*  2. Реалізуйте в класі інтерфейс *IComparer* для порівняння збірок за ціною і за кількістю сторінок.  3. Реалізуйте інтерфейс *IEnumerable.* Виведіть на консоль список збірок, впорядкований за ціною. |

**Література**

1. Лекція 11

2.C# 2005 и платформа .Net 3.0 для профессионалов. Нейгел К., Ивьен Б. и др. –М.: ООО “И.Д. Вильямс", 2008. –1376 с.

**Лабораторна робота № 17**

**Масиви і стандартні класи C#**

**Мета роботи:**

1. Робота з масивами в C#

2. Оператор foreach

3. Генерація випадкових чисел

4. Робота з математичними функціями

5. Лінійний пошук

6. Двійковий пошук

**Методичні рекомендації до виконання лабораторної роботи**

1. Прочитайте теоретичну частину л/р. Спробуйте відповісти на наведені в кінці запитання

2. Прочитайте цю лабораторну роботу та виконайте наведені в ній приклади (вони всі працездатні)

3. При виконанні завдань зверніть увагу на приклад застосування до масивів циклу **foreach**

4. Також зверніть увагу на опис і реалізацію методу бісекцій (ділення навпіл) знаходження коренів нелінійних рівнянь. Те також задача на одновимірні масиви і цикли

5. Для поглибленого вивчення цього матеріалу прочитайте розділ 5 книги [1].

**Порядок виконання роботи**

1. Створити директорію Lab4, в якій будуть розміщуватися проекти цієї лабораторної роботи.

2. Виконати завдання свого варіанту у вигляді одного консольного проекту

3. Зафіксувати результати у Контрольній роботі 1.

**Приклади виконання завдань**

**1. Робота з масивами в C#**

Розглянемо приклад програми з лекції 5, яка визначає суму і кількість від'ємних елементів, а також максимальний елемент масиву, що складається з 6 цілочисельних елементів.

Тут елементи масиву **а** ініціалізуються при створенні масиву. Далі в циклі з параметром елементи масиву виводяться на консоль. Зверніть увагу, що для виводу використовується метод Console.Write а не Console.WriteLine, тому всі елементи виводяться в один рядок. Символ табуляциії "\t" в методі Console.Write розділяє елементи при виводі.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Console\_Lab4

{

class Program

{

static void Main(string[] args)

{

const int n = 6;

int[] a = new int[n] { 3, 12, 5, -9, 8, -4 };

Console.WriteLine( "Початковий масив:" );

for ( int i = 0; i < n; ++i )

Console.Write( "\t" + a[i] );

Console.WriteLine();

long sum = 0; // сума від'ємних елементів

int num = 0; // кількість від'ємних елементів

for ( int i = 0; i < n; ++i )

if ( a[i] < 0 )

{

sum += a[i];

++num;

}

Console.WriteLine( "Сума від'ємних = " + sum );

Console.WriteLine( "Кількість від'ємних = " + num );

int max = a[0]; // максимальний елемент

for ( int i = 1; i < n; ++i )

if ( a[i] > max ) max = a[i];

Console.WriteLine( "Максимальний елемент = " + max );

Console.ReadKey();

}

}

}

Розглянемо ще одну "класичну" задачу – генерація чисел Фібоначчі: послідовності чисел, яка задовольняє умовам

F1 = 1; F2 = 1; Fk = Fk-1 + Fk-2 для k>2.

Цю задачу можна реалізувати з використанням масивів. В наступному прикладі кількість чисел вводиться з консолі, потім створюється одновимірний масив вказаної розмірності, який заповнюється згенерованими числами.

**Генерація чисел Фібоначчі**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Console\_Lab4

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Введіть кількість чисел послідовності");

int n = int.Parse(Console.ReadLine());

int[] fibonachi = new int[n];

fibonachi[0] = 1;

fibonachi[1] = 1;

for (int i = 2; i < n; i++)

{

fibonachi[i] = fibonachi[i - 2] + fibonachi[i - 1];

}

for (int i = 0; i < n; i++)

Console.WriteLine("fibonachi[" + i + "]=" + fibonachi[i]);

Console.ReadKey();

}

}

}

**Генерація простих чисел**

**Алгоритм Ератосфена**

Простим є число, яке ділиться тільки на 1 і на себе.

Алгоритм пошуку послідовності простих чисел запропонував древньогрецький математик Ератосфен, і він отримав назву **Решето Ератосфена.**

**Ідея і загальний опис алгоритму.**

Є розташована в ряд за збільшенням послідовність цілих чисел. Спочатку в ній викреслюються усі числа кратні 2, окрім її самої, і так до N. Далі із списку, що вийшов, береться число, що йде за двійкою, - трійка, викреслюються усі кратні 3 числа, окрім її самої. У такому вигляді алгоритм триває для частини послідовності, що залишилася, і у результаті отримаємо усі прості числа у вказаному діапазоні.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 |
| 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 |
| 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 |
| 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 100 |

У таблиці наведені усі цілі числа від 2 до 100. Червоним помічені ті, які були видалені в процесі виконання алгоритму Решето Ератосфену.

Тепер розглянемо алгоритм детальніше, розбивши його на декілька частин. Отже, для пошуку простих чисел методом Решета Ератосфену треба:

1. Організувати список з чисел від 2 до N, а також логічний масив розмірністю N;
2. У вільну змінну R записати число 2;
3. Виключити усі числа кратні R, починаючи з R\*2;
4. Записати в R наступне за R не закреслене число;
5. Повторювати дії, описані в двох попередніх кроках.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace SimpleNumber

{

class Program

{

//Нахождение простых чисел. Алгоритм "Решето Эратосфена"

//Запишем подряд все числа от 2 до N.

//Дальше вычеркнем из этого списка все числа кратные 2,

// исключая саму двойку, потом вычеркнем все числа кратные 3,

//исключая само число 3, число 4 уже вычеркнуто, вычеркиваем числа кратные 5 и т.д.

//Продолжаем этот процесс, пока квадрат очередного числа не превысит N.

static void Main(string[] args)

{

bool[] table = new bool[100];

int i, j;

// Отмечаем все числа как простые

for (i = 0; i < table.Length; i++)

table[i] = true;

// Вычеркиваем лишнее

for (i = 2; i \* i < table.Length; i++)

if (table[i])

for (j = 2 \* i; j < table.Length; j += i)

table[j] = false;

// Выводим найденное

for (i = 2; i < table.Length; i++)

{

if (table[i])

Console.WriteLine(i);

}

Console.ReadKey();

}

}

}

**2. Оператор foreach**

Оператор **foreach** використовується для перебору елементів в масивах та інших колекціях і є новим типом циклу, реалізованому в C#. В цьому прикладі створюється *ступінчастий масив*, який заповнюється даними. За допомогою вкладених операторів циклу елементи масиву виводяться на консоль у вигляді матриці.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Console\_Lab4

{

class Program

{

static void Main(string[] args)

{

//оператор foreach і ступінчасті масиви

int[][] а = new int[3][];

а[0] = new int[5] { 24, 50, 18, 3, 16 };

а[1] = new int[3] { 7, 9, -1 };

а[2] = new int[4] { 6, 15, 3, 1 };

Console.WriteLine("Початковий масив:");

foreach (int[] mas1 in а)

{

foreach (int x in mas1)

Console.Write("\t" + x);

Console.WriteLine();

}

Console.ReadKey();

}

}

}

**3. Генерація випадкових чисел**

Потреба в генерації випадкових чисел часто виникає в програмуванні. В C# є *нестатичний* клас Random, методи якого дозволяють генерувати різні послідовності випадкових чисел. Нижче наведено приклад з лекції 4 в якому в методі ValsGenerator створюється об'єкт **aRand**. Далі в циклі викликається метод **aRand.Next(100),** який при кожному виклику генерує одне випадкове число від 1 до100 і призначає його черговому елементу масиву. Зверніть увагу, що метод **ValsGenerator** є статичним.

В методі **Main** створюється масив **Data,** викликається метод **ValsGenerator** для його заповнення, потім викликається метод Array.Sort(Data) для його сортування. Після чого елементи масиву виводяться в циклі на консоль.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Console\_Lab4

{

class Program

{

// генератор даних

static void ValsGenerator(int[] Vals)

{

// Random - клас для генерації випадкових чисел

Random aRand = new Random();

// заповнення масиву

for (int i = 0; i < Vals.Length; i++)

Vals[i] = aRand.Next(100);

}

static void Main(string[] args)

{

const int N = 10;

int[] Data = new int[N];

ValsGenerator(Data);

Array.Sort(Data);

Console.WriteLine("Друк відсортованих даних");

for (int i = 0; i < Data.Length; i++)

Console.WriteLine("Data[" + i + "] = " + Data[i]);

Console.ReadLine();

}

}

}

Зверніть увагу, що при виводі масиву

Console.WriteLine("Data[" + i + "] = " + Data[i]);

виконується неявне приведення з типу int до типу string.

**4. Робота з математичними функціями**

Клас Math містить методи для роботи з математичними функціями.

Приклад 4.9. з лекції 5 демонструє використання методів обчислення математичних функцій, які містяться в класі Math.

Розглянемо ще один приклад – використання масивів і математичних функцій в задачах обчислювальної математики.

**Постановка задачі**

Знайти дійсні корені рівняння 6x4-3x3+8x2-5=0 за методом бісекції (ділення навпіл) на відрізках [0, 1], [-1, 0].

**Алгоритм методу:**

Нехай [а,b] відрізок, на якому шукаються корені. Припустимо, що функція f(x) неперервна на [а,b] і на кінцях приймає значення різних знаків .



*Алгоритм* методу полягає в побудові послідовності вкладених відрізків, на кінцях яких функція приймає значення різних знаків. Кожний наступний відрізок отримують діленням навпіл попереднього. Опишемо один крок ітераційного методу. Нехай на *к-*ому кроці знайдено відрізок такий, що . Знайдемо середину відрізку . Якщо, то - корінь і задача вирішена. Якщо ні, то з двох половин відрізку вибираємо той, на кінцях якого функція має протилежні знаки:



, , якщо



, , якщо



*Критерій закінчення ітераційного процесу*: якщо довжина відрізку знаходження кореня менше 2, то ітерації припиняють і за значення кореня із заданою точністю приймають середину відрізку.



Код програми може бути таким:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace bicection

{

class Program

{ //Знайти корені нелінійного рівняння

//6x4-3x3+8x2-5=0

//x1=0.74213

//x2=-0.6365

static double f(double x)

{

//рівняння, для якого шукаємо корені

double y = 6 \* Math.Pow(x, 4) - 3 \* Math.Pow(x, 3) + 8 \* Math.Pow(x, 2) - 5;

return y;

}

// --------------------------------------------------------------------

static double bicect(double left, double right)

{

//метод бісекцій

double eps = 0.00001;

double center = 0;

while (right - left > eps \* 2)

{

center = (right - left) / 2 + left;

if (f(center) \* f(left) > 0)

left = center;

else

right = center;

}

return center;

}

// ----------------------------------------------------

static void Main(string[] args)

{

//метод бісекцій

double x1 = bicect(0, 1);

double x2 = bicect(-1, 0);

Console.WriteLine("Метод бісекцій");

Console.WriteLine("x1={0},x2= {1}", x1, x2);

Console.ReadKey();

}

}

}

В цій програмі є два методи: static double f(double x), в якому визначається рівняння, і метод static double bicect(double left, double right). Для обчислення ступеня в методі **f** використовується метод Math.Pow( ).

Другий метод static double bicect(double left, double right) реалізує ітераційний процес. В методі Main два рази викликається метод бісекцій для знаходження коренів на вказаних відрізках. Результати виводляться на консоль.

**5. Лінійний пошук в масиві**

**Лінійний, послідовний пошук** — алгоритм знаходження заданого значення довільної функції на деякому відрізку.

**Формальний запис алгоритму:**

1. Визначити елемент, який шукаємо ***key;***

2. Встановити границі області пошуку L, R для масиву A [a];

3. Якщо аi = key – елемент знайдено;

4. Інакше – переходимо до наступного елементу.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Search

{

class Program

{

static void Main(string[] args)

{

int i, n, key, nom;

bool x = false;

Console.WriteLine("Розмір масиву");

n = int.Parse(Console.ReadLine());

int[] mas = new int[n]; //визначення масиву

Console.WriteLine("Шуканий елемент");

key = int.Parse(Console.ReadLine());

// Random - клас для генерації випадкових чисел

Random aRand = new Random();

for (i = 0; i < n; i++)

{

//формування масиву, заповнення його випадковими числами

mas[i] = aRand.Next(n);

Console.Write(mas[i] + " ");

}

for (i = 0; i < n; i++)

{ //якщо цей елемент дорівнює шуканому

if (mas[i] == key) //то x призначаємо true

{ x = true; nom = i; break; } //і виходимо з циклу

}

if (x == true)

Console.WriteLine("Елемент знайдено");

else Console.WriteLine("\nЕлемент не знайдено");

Console.ReadKey();

}

}

}

**6. Двійковий (бінарний) пошук**

**Двійковий (бінарний) пошук** (також відомий як метод ділення навпіл і дихотомія) — класичний алгоритм пошуку елемента у відсортованому масиві.

**Формальний запис алгоритму:**

1. Масив ділитися на дві рівні частини, шляхом визначення першого (a), останнього (b) і середнього (c) елементів;
2. Середній елемент порівнюється з шуканим (s):

якщо **s<c.** Останньому елементу призначається значення середнього, тим самим ділянка пошуку зменшується удвічі: b=c;

якщо **s>c**. Першому елементу призначається значення середнього, і ділянка пошуку зменшується удвічі: a=c;

якщо **s=c**. Елемент знайдений, і робота алгоритму завершується**.**

1. Якщо для перевірки не залишився жодного елементу, то алгоритм завершується, інакше виконується перехід до пункту 2.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace BinarySearch

{

class Program

{

static void Main(string[] args)

{

int i, n, key, begin, end, c;

bool x = false;

Console.WriteLine("Розмір масиву");

n = int.Parse(Console.ReadLine());

int[] mas = new int[n]; //визначення масиву

Console.WriteLine("Шуканий елемент");

key = int.Parse(Console.ReadLine());

for (i = 0; i < n; i++)

{

//формування масиву, заповнення його числами кратними n (10,20,....)

mas[i] = n \* i;

Console.Write(mas[i] + " ");

}

begin = 0; end = n; //ліва і права границі масиву

while (begin < end)

{

// власне реалізація алгоритму

c = begin + (end - begin) / 2;

if (key < mas[c]) end = c;

else if (key > mas[c]) begin = c + 1;

else { x = true; break; }

}

if (x == true)

Console.WriteLine("Елемент знайдено");

else Console.WriteLine("\nЕлемент не знайдено");

Console.ReadKey();

}

}

}

**7. Варіанти завдань для Лабораторної роботи № 4**

**Номер варіанта відповідає номеру прізвища студента у списку групи**

Кожний студент повинен виконати по чотири завдання свого варіанту

|  |  |
| --- | --- |
| **№ варі-анту** | **Зміст завдання** |
| 1 | 1. Реалізувати функцію знаходження простих чисел у вказаному діапазоні. Кількість чисел ввести з консолі. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Створити масив для зберігання значень зросту студентів групи (20 чоловік). Заповнити масив за допомогою класу Random цілими числами в діапазоні від 160 до 190 включно. Відсортувати масив і вивести на консоль його елементи.  3. Є прямокутний масив розмірністю (5x5). Визначити суму всіх елементів третього рядка і суму всіх елементів першого стовпчика.  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння  6x4-3x3+8x2-25=0  5. Створити масив з трьох цілих чисел і ініцілізувати при створенні. Визначити мінімальне і максимальне число. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 2 | 1. Реалізувати функцію знаходження простих чисел у вказаному діапазоні. Кількість чисел ввести з консолі. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Згенерувати масив М випадкових чисел в діапазоні від 1 до 50. Обчислити суму елементів масиву, середнє арифметичне (мат.очікування) елементів M, квадратний корінь від М, min і max елементів масиву. Результати вивести на консоль.  3. В двовимірному масиві зберігається інформація про зарплату 18 співробітників за 12 місяців (за січень – першому стовпчику, лютий – другому, ...). Визначити загальний бюджет зарплати за рік, загальну і середню зарплату за квітень.  4. Реалізувати консольній застосунок знаходження методом Ньютона коренів нелінійного рівняння  (x2-6\*x)2 -2(x-3)2=81  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 3 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні 21 до 97. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Згенерувати масив випадкових чисел в діапазоні від 10 до 100. Обчислити добуток елементів масиву, середнє арифметичне (мат.очікування) елементів M, min і max елементів масиву. Результати вивести на консоль.  3. В двовимірному масиві зберігається інформація про зарплату 18 співробітників за 12 місяців (за січень – першому стовпчику, лютий – другому, ...). Визначити загальну зарплату кожного співробітника за рік і середню зарплату за кожний місяць.  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння  (x2+2x)2-(x+1)2=55  5. Створити масив з трьох цілих чисел і ініцілізувати при створенні. Визначити мінімальне і максимальне число. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 4 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 101 до 199. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Згенерувати масив випадкових дійсних чисел в діапазоні від 0 до 1. Відсортувати масив, знайти min і max елементи. Результати вивести на консоль.  3. В двовимірному масиві зберігається інформація про зарплату 30 працівників фірми за 12 місяців (за січень – першому стовпчику, лютий – другому, ...). Визначити середню зарплату фірми за кожний місяць року, а також середню зарплату за кожний місяць кожного співробітника.  4. Реалізувати консольній застосунок знаходження коренів нелінійного рівняння методом бісекції  (x2+x+1)( x2+x+1) -12=0  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 5 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 101 до 199. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Згенерувати дві серії випадкових чисел, що повторюються. Кількість чисел ввести з консолі. Результати вивести на консоль.  3. В двовимірному масиві зберігається інформація про оцінки студентів з двох дисциплін (з дисципліни ОС – в першому стовпчику, з дисципліни Мова С# - в другому). Визначити середню успішність студентів групи і середній бал (рейтинг) кожного студента з цих дисциплін.  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння  (x2-5\*x+7) 2- (x-2)(x-3)=0  5. Створити масив з трьох цілих чисел і ініцілізувати при створенні. Визначити мінімальне і максимальне число. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 6 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 1 до 99. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Згенерувати масив чисел Фібоначчі. Кількість чисел ввести з консолі. Результати вивести на консоль.  3. Фірма має 10 магазинів. Інформація про прибуток кожного магазину за кожний місяць зберігається в двовимірному масиві (першрго – в першому рядку, другого – в другому, і т.д.). Обчислити середній прибуток кожного магазину.  4. Реалізувати консольній застосунок знаходження методом Ньютона коренів нелінійного рівняння  x4-13x2+36=0  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 7 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 11 до 51. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Створити масив для зберігання значень ваги студентів групи (20 чоловік). Заповнити масив за допомогою класу Random цілими числами в діапазоні від 50 до 100 включно. Відсортувати масив і вивести на консоль його елементи.  3. В двовимірному масиві зберігається інформація про оцінки кожного із 20 студентів групи по кожній з 10 дисциплін. Перший рядок – інформація про оцінки першого студента, друга – другого і т.д. Обчислити середню успішність кожного студента.  4. Реалізувати консольній застосунок знаходження коренів нелінійного рівняння методом бісекції  2x8+x4-15=0  5. Створити масив з чотирьох дійсних чисел і ініцілізувати при створенні. Визначити мінімальне і максимальне число. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 8 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 1 до 101. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Створити масив з 10 елементів і заповнити його значеннями в коді. Всі його елементи:  - зменшити на 20;  - помножити на останній елемент  - збільшити на число В, яке ввести з консолі  3. В двовимірному масиві зберігається інформація про оцінки кожного із 20 студентів групи по кожній з 10 дисциплін. Перший рядок – інформація про оцінки першого студента, друга – другого і т.д. Обчислити середню успішність групи по кожній з дисциплін і в цілому успішність групи.  4. Реалізувати консольний застосунок знаходження методом бісекції коренів нелінійного рівняння  x4-13x2+36=0  5. Створити масив з чотирьох дійсних чисел і ініцілізувати при створенні. Визначити мінімальне і максимальне число. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 9 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 11 до 51. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Створити масив з 10 елементів і заповнити його значеннями в коді. Визначити суму всіх елементів масиву, суму квадратів всіх елементів масиву, середнє арифметичне всіх елементів масиву.  3. В двовимірному масиві 3x5 знайти суму елементів в кожному рядку і суму елементів кожного стовпчика.  4. Реалізувати консольній застосунок знаходження коренів нелінійного рівняння методом Ньютона  x4-2x3+x-132=0  5. Створити масив з чотирьох дійсних чисел і ініцілізувати при створенні. Визначити мінімальне і максимальне число. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 10 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 101 до 151. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Створити масив з 10 елементів і заповнити його значеннями в коді. Всі його елементи: збільшити у 2 рази, розділити на перший елемент, зменшити на число А, яке ввести з консолі.  3. Три групи студентів, в кожній з яких 20 чоловік, в сесію здавали по 4 екзамени. Відомості про оцінки кожної групи зберігаються в двовимірних масивах. Визначити кращу групу за середньою оцінкою  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння  8x4+6x3-13x2-x+3=0  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль.  6. Знайти методом лінійного пошуку кількість вказаних цифр в масиві {1,2,5,3,7,5,1,3,4}  7. Знайти методом лінійного пошуку максимальний і мінімальний елементи в масиві {4,5,2,3,8,7,6,1}  8. Знайти методом двійкового пошуку потрібний елемент у масиві розміру 100.  9. Модифікувати програму двійкового пошуку у масиві, використавши метод BinarySearch класу Array |
| 11 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 1 до 151. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена.  2. Створити масив з 8 елементів і заповнити його випадковими дійсними числами в діапазоні від 1 до 20. Результати вивести на консоль  3. Створити прямокутний масив і заповнити його випадковими числами від 1 до 40. Знайти суму елементів другого стовпчика    4. Реалізувати консольній застосунок знаходження методом Ньютона коренів нелінійного рівняння  x3+6x+4x2+3=0  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль. |
| 12 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 1 до 131. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена  2. Створити масив з 8 елементів і заповнити його випадковими дійсними числами в діапазоні від 21 до 40. Результати вивести на консоль  3. Створити прямокутний масив і заповнити його випадковими числами від 1 до 40. Знайти суму елементів третього рядка масиву, які менше 20  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння  2x4-x3-9x2+13x-5=0  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль. |
| 13 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 1 до 97. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена  2. Створити масив і заповнити його 20-ма першими цілими числами, які діляться на 13 або на 17 і знаходяться в інтервалі, ліва границя якого рівна 300.  3. Створити прямокутний масив і заповнити його випадковими числами від 1 до 40. Знайти суму елементів першого рядка масиву і кількість його не нульових елементів  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння  (x-1)3+(2x+3)3=27x3+8  5. Створити масив з чотирьох дійсних чисел і ініцілізувати при створенні. Визначити мінімальне і максимальне число. Застосувати методи класу Math. Результати вивести на консоль. |
| 14 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 1 до 97. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена  2. Створити масив і заповнити його 30-ма простими числами (простим є число, яке ділиться на 1 і самого себе)  3. Створити прямокутний масив цілих чисел і  заповнити його випадковими числами від 1 до 100. Визначити суму елементів масиву, які більше 30 і середнє арифметичне всіх елементів  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння  x4-4x3-19x2+106x-120=0  5. Створити масив з чотирьох дійсних чисел і ініцілізувати при створенні. Визначити мінімальне і максимальне число. Застосувати методи класу Math. Результати вивести на консоль. |
| 15 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 11 до 101. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена  2. Створити масив і заповнити його 20-ма випадковими числами в діапазоні 20 до 100. Вивести на консоль елемент за його індексом в масиві.  3. Створити прямокутний масив цілих чисел і заповнити його випадковими числами від 1 до 60. Визначити суму елементів масиву, які більше 10 і середнє арифметичне елементів першого рядка  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння  x4-2x3+x-132=0  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль. |
| 16 | 1. Реалізувати функцію знаходження всіх простих чисел у вказаному діапазоні (ввести з консолі). Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена  2. Створити масив, розмір масиву ввести з консолі, заповнити його випадковими числами від 1 до 50. Відсортувати елементи. Вивести на консоль в прямому і зворотному порядку.  3. Створити прямокутний масив цілих чисел і ініціалізувати його при створенні. Для кожного рядка масиву обчислити суму від'ємних елементів.  4. Реалізувати консольній застосунок знаходження методом Ньютона коренів нелінійного рівняння  6x3-x2-20x+12=0  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль. |
| 17 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 1 до 101. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена  2. Створити масив, розмір масиву ввести з консолі, заповнити його випадковими дійсними числами від 1 до 50. Обчислити корінь квадратний кожного з його елементів і середнє арифметичне двох сусідніх елементів.  3. Створити прямокутний масив цілих чисел і ініціалізувати його при створенні. Для кожного рядка масиву обчислити суму додатних і від'ємних елементів.  4. Реалізувати консольній застосунок знаходження методом Ньютона коренів нелінійного рівняння  x4+1=2(1+x)4  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль. |
| 18 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 1 до 51. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена  2. Створити масив, в якому зберігаються дані про кількість опадів, які випали за кожний день місяця (наприклад, в листопаді). Обчислити загальну і середню кількість опадів за місяць.  3. Створити прямокутний масив цілих чисел і ініціалізувати його при створенні. Для кожного стовпчика масиву обчислити суму додатніх, від'ємних елементів.  4. Реалізувати консольній застосунок знаходження методом Ньютона коренів нелінійного рівняння  x4+1=2(1+x)4  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль. |
| 19 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 1 до 51. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена  2. Створити масив, в якому зберігаються дані про температуру в одному місяці. Обчислити середню температуру в місяці.  3. Створити прямокутний масив цілих чисел і ініціалізувати його при створенні. Для кожного рядка масиву обчислити кількість елементів, які більше 10.  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння.  x4+1=2(1+x)4  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль. |
| 20 | 1. Реалізувати функцію знаходження всіх простих чисел в діапазоні від 13 до 151. Згенеровані числа зберегти в масиві. Елементи масиву вивести на консоль. При реалізації використати алгоритм Ератосфена  2. Створити масив, в якому зберігаються дані про кількість опадів, що випали за кожний день місяця (наприклад, в травні). Обчислити в якій половині місяця (першій чи другій) випало більше опадів. Обчислити в якій декаді місяця кількість опадів була найбільшою  3. Створити прямокутний масив цілих чисел і ініціалізувати його при створенні. Знайти максимальний і мінімальний елементи кожного рядка.  4. Реалізувати консольній застосунок знаходження методом бісекції коренів нелінійного рівняння  x4+1=2(1+x)4  5. Визначити мінімальне і максимальне число з трьох чисел. Числа ввести з консолі. Застосувати методи класу Math. Результати вивести на консоль. |

**Література**

1. Лекція 4

2.C# 2005 и платформа .Net 3.0 для профессионалов. Нейгел К., Ивьен Б. и др. –М.: ООО “И.Д. Вильямс", 2008. –1376 с.

3. Д.М. Златопольский. Сборник задач по программированию. 2-е издание. // СПб.: БХВ-Петербург, 2007.-240 с.

**Лабораторна робота №18**

**ТЕМА: ENTITY FRAMEWORK**

**План**

1. Про платформу
2. Перший додаток з Entity Framework. Підхід Code First
3. Підходи Code First до існуючої бази даних
4. Основні операції з даними

# LINQ to Entities

Entity Framework являє спеціальну об'єктно-орієнтовану технологію на базі фреймворка .NET для роботи з даними. Якщо традиційні засоби ADO.NET дозволяють створювати підключення, команди та інші об'єкти для взаємодії з базами даних, то Entity Framework являє собою більш високий рівень абстракції, який дозволяє абстрагуватися від самої бази даних і працювати з даними незалежно від типу сховища. Якщо на фізичному рівні ми оперуємо таблицями, індексами, первинними і зовнішніми ключами, то на концептуальному рівні, який нам пропонує Entity Framework, ми вже працює з об'єктами.

Центральної концепцією Entity Framework є поняття сутності або entity. Сутність представляє набір даних, асоційованих з певним об'єктом. Тому дана технологія передбачає роботу не з таблицями, а з об'єктами і їх наборами.

Будь-яка сутність, як і будь-який об'єкт з реального світу, має низку властивостей. Наприклад, якщо сутність описує людину, то ми можемо виділити такі властивості, як ім'я, прізвище, зріст, вік, вага. Властивості необов'язково представляють прості дані типу int, а й можуть представляти більш комплексні структури даних. І у кожної сутності може бути одна або кілька властивостей, які будуть відрізняти цю сутність від інших і будуть унікально визначати цю сутність. Подібні властивості називають ключами.

При цьому суті можуть бути пов'язані асоціативним зв'язком один-до-багатьох, один-до-одного і багато-до-багатьох, подібно до того, як в реальній базі даних відбувається зв'язок через зовнішні ключі.

Відмінною рисою Entity Framework є використання запитів LINQ для вибірки даних з БД. За допомогою LINQ ми можемо не тільки отримувати певні рядки, що зберігають об'єкти, з бд, а й отримувати об'єкти, пов'язані різними асоціативними зв'язками.

Іншим ключовим поняттям є Entity Data Model. Ця модель зіставляє класи сутностей з реальними таблицями в БД.

Entity Data Model складається з трьох рівнів: концептуального, рівень сховища і рівень зіставлення (маппінга).

На концептуальному рівні відбувається визначення класів сутностей, які використовуються в додатку.

Рівень сховища визначає таблиці, стовпці, відносини між таблицями і типи даних, з якими порівнюється використовувана база даних.

Рівень зіставлення (маппінга) служить посередником між попередніми двома, визначаючи зіставлення між властивостями класу суті і стовпцями таблиць.

Таким чином, ми можемо через класи, визначені у додатку, взаємодіяти з таблицями з бази даних.

Способи взаємодії з БД

Entity Framework передбачає три можливі способи взаємодії з базою даних:

Database first: Entity Framework створює набір класів, які відображають модель конкретної бази даних

Model first: спочатку розробник створює модель бази даних, по якій потім Entity Framework створює реальну базу даних на сервері.

Code first: розробник створює клас моделі даних, які будуть зберігатися в бд, а потім Entity Framework за цією моделлю генерує базу даних і її таблиці

**2.Перший додаток з Entity Framework. Підхід Code First**

В середовищі Visual Studio створюємо консольний застосунок.

Тепер насамперед додамо новий клас, який буде описувати дані. Нехай наш додаток буде присвячено роботі з користувачами. Тому додамо в проект новий клас User:

public class User

{

public int Id {get; set; }

public string Name {get; set; }

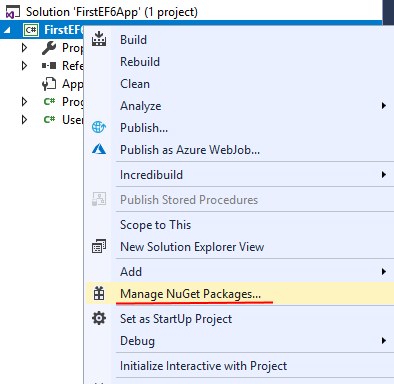
public int Age {get; set; }

}

Це звичайний клас, який містить певну кількість автовластивостей. Кожна властивість буде зіставлятися з окремим стовпцем в таблиці з бд.

Треба відзначити, що Entity Framework при роботі з Code First вимагає визначення ключа елемента для створення первинного ключа в таблиці в бд. За замовчуванням при генерації бд EF в якості первинних ключів буде розглядати властивості з іменами Id або [ім'я\_класу] Id (тобто UserId). Якщо ж ми хочемо назвати ключову властивість інакше, то нам потрібно буде внести додаткову логіку на c #.

Тепер для взаємодії з бд нам потрібен контекст даних. Це свого роду посередник між бд і класами, що описують дані. Але, у нас за замовчуванням ще не додана бібліотека для EF. Щоб її додати, натиснемо на проект в Explorer Solution правою кнопкою миші і виберемо в контекстному меню Manage NuGet Packages ...:



Потім у вікні управління NuGet-пакетами в вікні пошуку введемо слово "Entity" і виберемо пакет Entity Framework і встановимо його.

Після установки пакета додамо в проект новий клас UserContext:

using System;

using System.Collections.Generic;

using System.Data.Entity;

namespace FirstEF6App

{

class UserContext: DbContext

{

public UserContext ()

: Base ( "DbConnection")

{}

public DbSet <User> Users {get; set; }

}

}

Основу функціональності Entity Framework складають класи, що знаходяться в просторі імен System.Data.Entity. Серед усього набору класів цього простору імен слід виділити наступні:

DbContext: визначає контекст даних, який використовується для взаємодії з базою даних.

DbModelBuilder: зіставляє класи на мові C # з сутностями в базі даних.

DbSet / DbSet <TEntity>: представляє набір сутностей, що зберігаються в базі даних

У будь-якому додатку, що працює з БД через Entity Framework, нам потрібен буде контекст (клас похідний від DbContext) і набір даних DbSet, через який ми зможемо взаємодіяти з таблицями з БД. В даному випадку таким контекстом є клас UserContext.

У конструкторі цього класу викликається конструктор базового класу, в який передається рядок "DbConnection" - це ім'я майбутнього рядка підключення до бази даних. В принципі ми можемо не використовувати конструктор, тоді в цьому випадку рядок підключення носив би ім'я самого класу контексту даних.

І також в класі визначено одна властивість Users, яка буде зберігати набір об'єктів User. У класі контексту даних набір об'єктів представляє клас DbSet <T>. Через це властивість буде здійснюватися зв'язок з таблицею об'єктів User в бд.

І тепер нам треба встановити підключення до бази даних. Для установки підключення зазвичай використовується файл конфігурації програми. У проектах для десктопних додатків файл конфігурації називається App.config (як в нашому випадку), в проектах веб-додатків - web.config. У нашому випадку, оскільки у нас консольний додаток, це файл App.config. Після додавання Entity Framework він виглядає приблизно наступним чином:

<? Xml version = "1.0" encoding = "utf-8"?>

<Configuration>

<ConfigSections>

<Section name = "entityFramework" type = "System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version = 6.0.0.0, Culture = neutral, PublicKeyToken = b77a5c561934e089" requirePermission = "false" />

<! - For more information on Entity Framework configuration, visit http://go.microsoft.com/fwlink/?LinkID=237468 ->

</ ConfigSections>

<Startup>

<SupportedRuntime version = "v4.0" sku = ". NETFramework, Version = v4.7.1" />

</ Startup>

<EntityFramework>

<DefaultConnectionFactory type = "System.Data.Entity.Infrastructure.LocalDbConnectionFactory, EntityFramework">

<Parameters>

<Parameter value = "mssqllocaldb" />

</ Parameters>

</ DefaultConnectionFactory>

<Providers>

<Provider invariantName = "System.Data.SqlClient" type = "System.Data.Entity.SqlServer.SqlProviderServices, EntityFramework.SqlServer" />

</ Providers>

</ EntityFramework>

</ Configuration>

Вміст файлу в кожному конкретному випадку може відрізнятися. Але в будь-якому випадку після додавання EntityFramework в проект в ньому міститиметься елемент configSections. І після закриває тега </ configSections> додамо наступний елемент:

<ConnectionStrings>

<Add name = "DBConnection" connectionString = "data source = (localdb) \ MSSQLLocalDB; Initial Catalog = userstore; Integrated Security = True;"

providerName = "System.Data.SqlClient" />

</ ConnectionStrings>

Все підключення до джерел даних встановлюються в секції connectionStrings, а кожне окреме підключення представляє елемент add. У конструкторі класу контексту UserContext ми передаємо в якості назви підключення рядок "DbConnection", тому дана назва вказується в атрибуті name = "DBConnection".

Налаштування рядка підключення задає атрибут connectionString. В даному випадку ми встановлюємо назву бази даних, з якою будемо взаємодіяти - userstore.

Тепер перейдемо до файлу Program.cs і змінимо його зміст наступним чином:

using System;

namespace FirstEF6App

{

class Program

{

static void Main (string [] args)

{

using (UserContext db = new UserContext ())

{

// створюємо два об'єкти User

User user1 = new User {Name = "Tom", Age = 33};

User user2 = new User {Name = "Sam", Age = 26};

// додаємо їх в бд

db.Users.Add (user1);

db.Users.Add (user2);

db.SaveChanges ();

Console.WriteLine ( "Об'єкти успішно збережені");

// отримуємо об'єкти з бд і виводимо на консоль

var users = db.Users;

Console.WriteLine ( "Список об'єктів:");

foreach (User u in users)

{

Console.WriteLine ( "{0}. {1} - {2}", u.Id, u.Name, u.Age);

}

}

Console.Read ();

}

}

}

Так як клас UserContext через батьківський клас DbContext реалізує інтерфейс IDisposable, то для роботи з UserContext з автоматичним закриттям даного об'єкта ми можемо використовувати конструкцію using.

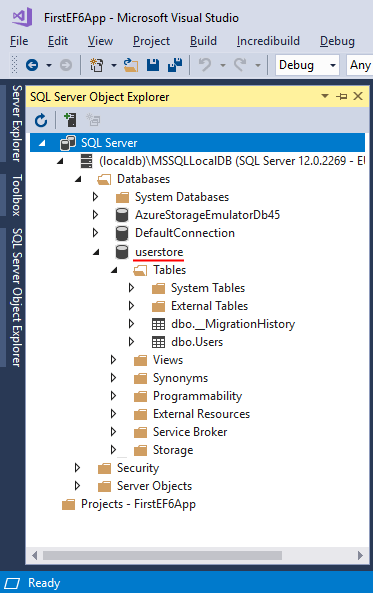
У конструкції using створюються два об'єкти User і додаються в базу даних. Для їх збереження нам досить використовувати метод Add: db.Users.Add (user1)

Щоб отримати список даних з бд, досить скористатися властивістю Users контексту даних: db.Users

Таким чином, Entity Framework забезпечує просте і зручне управління об'єктами з бази даних. При тому в даному випадку нам не треба навіть створювати базу даних і визначати в ній таблиці. Entity Framework все зробить за нас на основі визначення класу контексту даних і класів моделей. І якщо база даних вже є, то EF не буде повторно створювати її.

Наше завдання - тільки визначити модель, яка буде зберігатися в базі даних, і клас контексту. Тому даний підхід називається Code First - спочатку пишеться код, а потім по ньому створюється база даних і її таблиці.

Виникає питання, а де ж знаходиться БД? Щоб фізично побачити базу даних, ми можемо підключитися до неї з Visual Studio через вікно View-> SQL Server Object Explorer. Після цього ми можемо побачити в SQL Server Object Explorer створену базу даних, подивитися її будову, таблиці, відкрити і навіть змінити дані в таблицях:



**3. Підходи Code First до існуючої бази даних**

Так, цілком часто поширена ситуація, коли база даних вже є. І тут знову ж таки допоможе Code First.

База даних готова. Тепер нам треба додати підключення в файл конфігурації програми. У Solution Explorer знайдемо файл App.config і відкриємо його. Перед закриваючим тегом </ configuration> додамо нову секцію connectionStrings:

<? Xml version = "1.0" encoding = "utf-8"?>

<Configuration>

<! - решта вмісту ->

<ConnectionStrings>

<Add name = "UserDB" connectionString = "data source = (localdb) \ MSSQLLocalDB; Initial Catalog = userstoredb; Integrated Security = True;"

providerName = "System.Data.SqlClient" />

</ ConnectionStrings>

</ Configuration>

Тепер визначимо класи моделі даних і контексту. Додамо клас моделі User:

public class User

{

public int Id {get; set; }

public string Name {get; set; }

public int Age {get; set; }

}

І також додамо клас контексту даних:

using System;

using System.Collections.Generic;

using System.Data.Entity;

namespace CodeSecond

{

public class UserContext: DbContext

{

public UserContext ():

base ( "UserDB")

{}

public DbSet <User> Users {get; set; }

}

}

У конструкторі контексту даних ми передаємо в конструктор базового класу ім'я рядка з‘єднання із файла конфігурації App.config. Так як ми визначили там рядок підключення UserDB (<add name = "UserDB"), те саме це значення і використовується в конструкторі.

Однак, як варіант, ми могли не використовувати конструктор в класі контексту даних, а визначити в якості імені рядка підключення назву цього класу, наприклад: <add name = "UserContext" connectionString = "....

І для отримання даних визначимо наступний код в консольному додатку:

using (UserContext db = new UserContext ())

{

var users = db.Users;

foreach (User u in users)

{

Console.WriteLine ( "{0}. {1} - {2}", u.Id, u.Name, u.Age);

}

}

1. **Основні операції з даними**

Більшість операцій з даними являють собою CRUD-операції (Create, Read, Update, Delete), тобто отримання даних, створення, оновлення і видалення. Entity Framework дозволяє легко проводити дані операції.

Для прикладів з операціями візьмемо простеньку модель Phone:

public class Phone

{

public int Id {get; set; }

public string Name {get; set; }

public int Price {get; set; }

}

І наступний клас контексту даних:

public class PhoneContext: DbContext

{

public PhoneContext (): base ( "DefaultConnection")

{}

public DbSet <Phone> Phones {get; set; }

}

Додавання

Для додавання застосовується метод Add () у об'єкта DbSet:

using (PhoneContext db = new PhoneContext ())

{

Phone p1 = new Phone {Name = "Samsung Galaxy S7", Price = 20000};

Phone p2 = new Phone {Name = "iPhone 7", Price = 28000};

// додавання

db.Phones.Add (p1);

db.Phones.Add (p2);

db.SaveChanges (); // збереження змін

var phones = db.Phones.ToList ();

foreach (var p in phones)

Console.WriteLine ( "{0} - {1} - {2}", p.Id, p.Name, p.Price);

}

Після додавання треба зберегти всі зміни за допомогою методу SaveChanges ().

Редагування

Контекст даних здатний відстежувати зміни об'єктів, тому для редагування об'єкта досить змінити його властивості і після цього викликати метод SaveChanges ():

using (PhoneContext db = new PhoneContext ())

{

// отримуємо перший об'єкт

Phone p1 = db.Phones.FirstOrDefault ();

p1.Price = 30000;

db.SaveChanges (); // зберігаємо зміни

}

Але розглянемо іншу ситуацію:

Phone p1;

using (PhoneContext db = new PhoneContext ())

{

p1 = db.Phones.FirstOrDefault ();

}

using (PhoneContext db = new PhoneContext ())

{

if (p1! = null)

{

p1.Price = 60000;

db.SaveChanges ();

}

}

Так як об'єкт Phone отриманий в одному контексті, змінюється для іншого контексту, який його не відслідковує. В результаті зміни не будуть збережені. Щоб зміни збереглися, нам явно треба встановити для його стану значення EntityState.Modified:

using (PhoneContext db = new PhoneContext ())

{

if (p1! = null)

{

p1.Price = 60000;

db.Entry (p1) .State = EntityState.Modified;

db.SaveChanges ();

}

}

Видалення

Для видалення об'єкта застосовується метод Remove() об'єкта DbSet:

using (PhoneContext db = new PhoneContext ())

{

Phone p1 = db.Phones.FirstOrDefault ();

if (p1! = null)

{

db.Phones.Remove (p1);

db.SaveChanges ();

}

}

Але як і в випадку з оновленням тут ми можемо зіткнутися з подібною проблемою, коли об'єкт отримуємо з бази даних в межах одного контексту, а намагаємося видалити в іншому контексті. І в цьому випадку нам треба встановити вручну у об'єкта стан EntityState.Deleted:

using (PhoneContext db = new PhoneContext ())

{

if (p1! = null)

{

db.Entry (p1) .State = EntityState.Deleted;

db.SaveChanges ();

}

}

Метод Attach

Якщо об'єкт отриманий в одному контексті, зберігається в іншому, то ми можемо встановлювати у нього вручну стан EntityState.Updated або EntityState.Deleted. Але є ще один спосіб: за допомогою методу Attach у об'єкта DbSet ми можемо прикріпити об'єкт до поточного контексту даних:

|  |
| --- |
| Phone p1;  using (PhoneContext db = new PhoneContext())  {      p1 = db.Phones.FirstOrDefault();  }  // редактирование  using (PhoneContext db = new PhoneContext())  {      if(p1!=null)      {          db.Phones.Attach(p1);          p1.Price = 999;          db.SaveChanges();      }  }  // удаление  using (PhoneContext db = new PhoneContext())  {      if(p1!=null)      {          db.Phones.Attach(p1);          db.Phones.Remove(p1);          db.SaveChanges();      }  } 5. LINQ to Entities Раніше ми використовували ряд операцій для отримання даних з БД. В основі подібних операцій лежить технологія LINQ (Language Integrated Query), або точніше LINQ to Entities. LINQ to Entities пропонує простий і інтуїтивно зрозумілий підхід для отримання даних за допомогою виразів, які за формою близькі виразами мови SQL.  Хоча при роботі з базою даних ми оперуємо запитами LINQ, але база даних розуміє тільки запити на мові SQL. Тому між LINQ to Entities і базою даних є провідник, який дозволяє їм взаємодіяти. Цим провідником є провайдер EntityClient. Він створює інтерфейс для взаємодії з провайдером ADO.NET для SQL Serverа.  Для початку взаємодії з базою даних створюється об'єкт EntityConnection. Через об'єкт EntityCommand він відправляє запити, а за допомогою об'єкта EntityDataReader зчитує витягнуті з БД дані. Однак розробнику не треба безпосередньо взаємодіяти з цими об'єктами, фреймворк все зробить за нього. Завдання ж розробника зводиться в основному до написання запитів до бази даних за допомогою LINQ.  Перш ніж приступити до огляду основних запитів в LINQ to Entities, для роботи з матеріалом цього розділу створимо нові моделі по зв'язку один-до-багатьох:  public class Company  {  public int Id {get; set; }  public string Name {get; set; }  public ICollection <Phone> Phones {get; set; }  public Company ()  {  Phones = new List <Phone> ();  }  }  public class Phone  {  public int Id {get; set; }  public string Name {get; set; }  public int Price {get; set; }  public int CompanyId {get; set; }  public Company Company {get; set; }  }  У нас тут модель телефону і модель компанії-виробника. Тепер створимо контекст даних і ініціалізатор бази даних початковими даними:  class PhoneContext: DbContext  {  static PhoneContext ()  {  Database.SetInitializer (new MyContextInitializer ());  }  public PhoneContext (): base ( "DefaultConnection")  {}  public DbSet <Company> Companies {get; set; }  public DbSet <Phone> Phones {get; set; }  }  class MyContextInitializer: DropCreateDatabaseAlways <PhoneContext>  {  protected override void Seed (PhoneContext db)  {  Company c1 = new Company {Name = "Samsung"};  Company c2 = new Company {Name = "Apple"};  db.Companies.Add (c1);  db.Companies.Add (c2);  db.SaveChanges ();  Phone p1 = new Phone {Name = "Samsung Galaxy S5", Price = 20000, Company = c1};  Phone p2 = new Phone {Name = "Samsung Galaxy S4", Price = 15000, Company = c1};  Phone p3 = new Phone {Name = "iPhone5", Price = 28000, Company = c2};  Phone p4 = new Phone {Name = "iPhone 4S", Price = 23000, Company = c2};  db.Phones.AddRange (new List <Phone> () {p1, p2, p3, p4});  db.SaveChanges ();  }  }  Щоб база даних вже містила деякі дані, в ініціалізаторі бд створюється кілька об'єктів. Щоб задіяти ініціалізатор, він викликається в статичному конструкторі контексту даних: Database.SetInitializer (new MyContextInitializer ());  Для створення запитів в Linq to Entities, так само, як і в Linq to Objects, ми можемо застосовувати оператори LINQ і методи розширення LINQ.  Наприклад, використовуємо деякі оператори LINQ:  using (PhoneContext db = new PhoneContext ())  {  var phones = from p in db.Phones  where p.CompanyId == 1  select p;  }  І той же запит за допомогою методів розширень LINQ:  using (PhoneContext db = new PhoneContext ())  {  var phones = db.Phones.Where (p => p.CompanyId == 1);  }  Обидва запити в результаті транслюються в один вираз sql:  SELECT [Extent1]. [Id] AS [Id],  [Extent1]. [Name] AS [Name],  [Extent1]. [Price] AS [Price],  [Extent1]. [CompanyId] AS [CompanyId]  FROM [dbo]. [Phones] AS [Extent1]  WHERE 1 = [Extent1]. [CompanyId]}  Важливо розуміти різницю між Linq to Entities і Linq to Objects:  var phones = db.Phones.Where (p => p.CompanyId == 1) .ToList (). Where (p => p.Id <10);  Тут використовуються два методи Where, але їх реалізація буде різною. У першому випадку, db.Phones.Where (p => p.CompanyId == 1) транслюється в вираз SQL, який було розглянуто вище. Далі метод ToList () за результатами запиту створює список в пам'яті комп'ютера. Після цього ми вже маємо справу зі списком в пам'яті, а не з базою даних. І далі виклик Where (p => p.Id <10) звертатиметься до списку в пам'яті і буде представляти Linq to Object. |

Вибірка і проекція з бази даних

Для вибірки застосовується метод Where. Виберемо з бд всі моделі, виробник яких - "Samsung":

using (PhoneContext db = new PhoneContext ())

{

var phones = db.Phones.Where (p => p.Company.Name == "Samsung");

foreach (Phone p in phones)

Console.WriteLine ( "{0}. {1} - {2}", p.Id, p.Name, p.Price);

}

Для вибірки одного об'єкта ми можемо використовувати метод Find (). Даний метод не є методом Linq, він визначений в класі DbSet:

Phone myphone = db.Phones.Find (3); // виберемо елемент з id = 3

Але в якості альтернативи ми можемо використовувати методи Linq First () / FirstOrDefault (). Вони отримують перший елемент вибірки, який відповідає певній умові. Використання методу FirstOrDefault () є більш гнучким, тому що якщо вибірка порожня, то він поверне значення null. А метод First () в тій же ситуації викине помилку.

Phone myphone = db.Phones.FirstOrDefault (p => p.Id == 3);

if (myphone! = null)

Console.WriteLine (myphone.Name);

Тепер зробимо проекцію. Припустимо, нам треба додати в результат вибірки назву компанії. Ми можемо використовувати метод Include для під'єднання до об'єкта пов'язаних даних з іншої таблиці: var phones = db.Phones.Include (p => p.Company). Але не завжди потрібні всі властивості об'єктів, що вибираються. У цьому випадку ми можемо застосувати метод Select для проекції витягнутих даних на новий тип:

using (PhoneContext db = new PhoneContext ())

{

var phones = db.Phones.Select (p => new

{

Name = p.Name,

Price = p.Price,

Company = p.Company.Name

});

foreach (var p in phones)

Console.WriteLine ( "{0} ({1}) - {2}", p.Name, p.Company, p.Price);

}

У підсумку метод Select з отриманих даних спроектує новий тип. В даному випадку ми отримаємо дані анонімного типу, але це також може бути визначений користувачем тип. наприклад:

public class Model

{

public string Name {get; set; }

public string Company {get; set; }

public int Price {get; set; }

}

І спроеціруємо вибірку на цей тип:

var phones = db.Phones.Select (p => new Model

{

Name = p.Name,

Price = p.Price,

Company = p.Company.Name

});

foreach (Model p in phones)

Console.WriteLine ( "{0} ({1}) - {2}", p.Name, p.Company, p.Price);

Linq to Entities підтримує звернення до вбудованих функцій SQL через спеціальні методи Count, Sum і т.д.

Кількість елементів у вибірці

Метод Count () дозволяє знайти кількість елементів у вибірці:

using (PhoneContext db = new PhoneContext ())

{

int number1 = db.Phones.Count ();

// знайдемо кількість моделей, які в назві містять Samsung

int number2 = db.Phones.Count (p => p.Name.Contains ( "Samsung"));

Console.WriteLine (number1);

Console.WriteLine (number2);

}

Мінімальне, максимальне і середнє значення

Для знаходження мінімального, максимального і середнього значень за вибіркою застосовуються функції Min (), Max () і Average () відповідно. Знайдемо мінімальну, максимальну і середню ціну за моделями:

using (PhoneContext db = new PhoneContext ())

{

// мінімальна ціна

int minPrice = db.Phones.Min (p => p.Price);

// максимальна ціна

int maxPrice = db.Phones.Max (p => p.Price);

// середня ціна на телефони фірми Samsung

double avgPrice = db.Phones.Where (p => p.Company.Name == "Samsung")

.Average (p => p.Price);

Console.WriteLine (minPrice);

Console.WriteLine (maxPrice);

Console.WriteLine (avgPrice);

}

Сума значень

Для отримання суми значень використовується метод Sum ():

using (PhoneContext db = new PhoneContext ())

{

// сумарна вартість всіх моделей

int sum1 = db.Phones.Sum (p => p.Price);

// сумарна вартість всіх моделей фірми Samsung

int sum2 = db.Phones.Where (p => p.Name.Contains ( "Samsung"))

.Sum (p => p.Price);

Console.WriteLine (sum1);

Console.WriteLine (sum2);

}

**ЗАВДАННЯ**

# 1.Встановити ENTITY framework для вашого проекту. Розробити підход Code First до Вашої бази даних. Написати 5 LINQ to Entities запитів. (Останнє завдання для кожного студента проекту окремо)

**Лабораторна робота №19**

**ТЕМА: Графіка в мові c#**

План

1. Простір імен System.Drawing
2. Клас - Graphics
3. Клас - Pen
4. Клас - Brush
5. Апроксимація β-сплайном.
6. Векторний добуток.
7. Інтерполяція поліномом Лангранжа.
8. Інтерполяція поліномом Ньютона.
9. Криві Безьє.

**Простір імен System.Drawing**

Простір імен System.Drawing забезпечує доступ до функціональних можливостей графічного інтерфейсу GDI +.

Простори імен:

System.Drawing.Drawing2D,

System.Drawing.Imaging,

System.Drawing.Text

забезпечують додаткові функціональні можливості.

Клас Graphics надає методи малювання на пристрої відображення.

Такі класи, як Rectangle і Point, інкапсулюють елементи GDI +.

Клас Pen використовується для малювання ліній і кривих, а класи, похідні від абстрактного класу Brush, використовуються для заливки фігур.

**Клас - Graphics**

Інкапсулює поверхню малювання GDI +. Цей клас не успадковується.

Методів в цьому класі величезна кількість, тому розглянемо деякі з них:

**AddMetafileComment** Додає коментар до поточного об'єкту Metafile.

**BeginContainer()** Зберігає графічний контейнер, що містить поточний стан даного об'єкта Graphics, а потім відкриває і використовує новий графічний контейнер.

**BeginContainer (Rectangle, Rectangle, GraphicsUnit)** Зберігає графічний контейнер, що містить поточний стан даного об'єкта Graphics, а також відкриває і використовує новий графічний контейнер із зазначеним перетворенням масштабу.

**BeginContainer (RectangleF, RectangleF, GraphicsUnit)** Зберігає графічний контейнер, що містить поточний стан даного об'єкта Graphics, а також відкриває і використовує новий графічний контейнер із зазначеним перетворенням масштабу.

**Clear** Очищає всю поверхню малювання і виконує заливку поверхні вказаним кольором фону.

**CopyFromScreen (Point, Point, Size)** Виконує передачу даних про колір, відповідних прямокутної області пікселів, блоками бітів з екрану на поверхню малювання об'єкта Graphics.

**CopyFromScreen (Point, Point, Size, CopyPixelOperation)** Виконує передачу даних про колір, відповідних прямокутної області пікселів, блоками бітів з екрану на поверхню малювання об'єкта Graphics.

**CopyFromScreen (Int32, Int32, Int32, Int32, Size)** Виконує передачу даних про колір, відповідних прямокутної області пікселів, блоками бітів з екрану на поверхню малювання об'єкта Graphics.

**Dispose** Звільняє всі ресурси, використовувані даним об'єктом Graphics.

**DrawArc (Pen, Rectangle, Single, Single)** Малює дугу, яка є частиною еліпса, заданого структурою Rectangle.

**DrawArc (Pen, Int32, Int32, Int32, Int32, Int32, Int32)** Малює дугу, яка є частиною еліпса, заданого парою координат, шириною і висотою.

**DrawArc (Pen, Single, Single, Single, Single, Single, Single)** Малює дугу, яка є частиною еліпса, заданого парою координат, шириною і висотою.

**DrawBezier (Pen, Point, Point, Point, Point)** Малює криву Безьє, яка визначається чотирма структурами Point.

**DrawLine (Pen, Point, Point)** Проводить лінію, що сполучає дві структури Point.

**DrawLine (Pen, Int32, Int32, Int32, Int32)** Проводить лінію, що сполучає дві точки, що задаються парами координат.

**DrawLine (Pen, Single, Single, Single, Single)** Проводить лінію, що сполучає дві точки, що задаються парами координат.

**DrawLines (Pen, Point [])** Малює набір сегментів ліній, які з'єднують масив структур Point.

**DrawPath** Малює об'єкт GraphicsPath.

**DrawPie (Pen, Rectangle, Single, Single)** Малює сектор, який визначається еліпсом, заданим структурою Rectangle і двома радіальними лініями.

**DrawPie (Pen, Int32, Int32, Int32, Int32, Int32, Int32)** Малює сектор, який визначається еліпсом, який заданий парою координат, шириною, висотою і двома радіальними лініями.

**DrawPie (Pen, Single, Single, Single, Single, Single, Single)** Малює сектор, який визначається еліпсом, який заданий парою координат, шириною, висотою і двома радіальними лініями.

**DrawPolygon (Pen, Point [])** Малює багатокутник, який визначається масивом структур Point.

**DrawRectangle (Pen, Rectangle)** Малює прямокутник, який визначається структурою Rectangle.

**DrawRectangle (Pen, Int32, Int32, Int32, Int32)** Малює прямокутник, який визначений парою координат, шириною і висотою.

**DrawRectangle (Pen, Single, Single, Single, Single)** Малює прямокутник, який визначений парою координат, шириною і висотою.

**DrawRectangles (Pen, Rectangle [])** Малює набір прямокутників, що визначаються структурами Rectangle.

DrawString (String, Font, Brush, PointF) Створює текстовий рядок що вказується, в заданому місці за допомогою визначених об'єктів Brush і Font.

**DrawString (String, Font, Brush, PointF, StringFormat)** Малює заданий текстовий рядок в заданому місці за допомогою визначених об'єктів Brush і Font, використовуючи атрибути форматування заданого формату StringFormat.

DrawString (String, Font, Brush, Single, Single) Створює текстовий рядок, що вказується , в заданому місці за допомогою визначених об'єктів Brush і Font.

**Equals (Object)** Визначає, чи рівний заданий об'єкт поточного об'єкту. (Успадкованих від Object.)

**ExcludeClip (Rectangle)** Обновляє вирізану область даного об'єкта Graphics, щоб виключити з неї частину, яка визначається структурою Rectangle.

**ExcludeClip (Region)** Обновляє вирізану область даного об'єкта Graphics, щоб виключити з неї частину, яка визначається структурою Region.

**FillClosedCurve (Brush, Point [])** Заповнює внутрішню частину замкнутої фундаментальної кривої, яка визначається масивом структур Point.

**FillClosedCurve (Brush, Point [], FillMode)** Заповнює внутрішню частину замкнутої фундаментальної сплайновой кривої, яка визначається масивом структур Point, використовуючи зазначений режим заливки.

**FillClosedCurve (Brush, Point [], FillMode, Single)** Заповнює внутрішню частину замкнутої фундаментальної кривої, яка визначається масивом структур Point, використовуючи зазначений режим заливки.

**FillEllipse (Brush, Rectangle)** Заповнює внутрішню частину еліпса, що визначається обмежуючим прямокутником, який заданий.

**FillPath** Заповнює внутрішню частину об'єкта GraphicsPath.

**FillPie (Brush, Rectangle, Single, Single)** Заповнює внутрішню частину сектора, що визначається еліпсом, який заданий структурою RectangleF, і двома радіальними лініями.

**FillPie (Brush, Int32, Int32, Int32, Int32, Int32, Int32)** Заповнює внутрішню частину сектора, що визначається еліпсом, який заданий парою координат, шириною, висотою і двома радіальними лініями.

**FillPie (Brush, Single, Single, Single, Single, Single, Single)** Заповнює внутрішню частину сектора, що визначається еліпсом, який заданий парою координат, шириною, висотою і двома радіальними лініями.

**FillPolygon (Brush, Point [])** Заповнює внутрішню частину багатокутника, обумовленого масивом точок, заданих структурами Point.

**FillPolygon (Brush, Point [], FillMode)** Заповнює внутрішню частину багатокутника, визначеного масивом точок, заданих структурами Point, використовуючи зазначений режим заливки.

**FillRectangle (Brush, Rectangle)** Заповнює внутрішню частину прямокутника, що визначається структурою Rectangle.

**FillRectangle (Brush, Int32, Int32, Int32, Int32)** Заповнює внутрішню частину прямокутника, який визначений парою координат, шириною і висотою.

**FillRectangle (Brush, Single, Single, Single, Single)** Заповнює внутрішню частину прямокутника, який визначений парою координат, шириною і висотою.

**FillRectangles (Brush, Rectangle [])** Заповнює внутрішню частину набору прямокутників, що визначаються структурами Rectangle.

**FillRegion** Заповнює внутрішню частину об'єкта Region.

**Flush()** Викликає примусове виконання всіх відкладених графічних операцій і негайно повертається, не чекаючи їх закінчення.

Flush (FlushIntention) Викликає примусове виконання всіх відкладених графічних операцій. При цьому відповідно до настройки метод чекає або не чекає закінчення операцій для повернення.

**Клас - Pen**

Визначає об'єкт, який використовується для малювання прямих ліній і кривих. Цей клас не успадковується.

**Pen (Color)** Ініціалізує новий екземпляр класу Pen з вказаним кольором.

**Pen (Color, Single)** Ініціалізує новий екземпляр класу Pen з зазначеними властивостями Color і Width. (Width - встановлює ширину пера Pen, в одиницях об'єкта Graphics, використовуваного для малювання)

**Клас - Brush**

Визначає об'єкти, які використовуються для заливки всередині графічних фігур, таких як прямокутники, еліпси, кола, багатокутники.

Це абстрактний базовий клас, який не може бути реалізованим. .Для створення об'єкта "перо" використовуються класи, похідні від Brush, такі як SolidBrush, TextureBrush і LinearGradientBrush.

Приклад:

Створюємо WindowForm проект. З панелі елементів кидаємо на форму наступні елементи: PictureBox, Button. Задаємо їх розміри. Метод, наведений нижче, повинен виконуватися по кліку на кнопку.

**private** **void** pictureBox1\_Click(**object** sender, EventArgs e)

{

*//Выбираем перо "myPen" черного цвета Black*

*//толщиной в 1 пиксель:*

Pen myWind = [new](http://www.google.com/search?q=new+msdn.microsoft.com) Pen(Color.Black, 2);

*//Объявляем объект "g" класса Graphics и предоставляем*

*//ему возможность рисования на pictureBox1:*

Graphics g = pictureBox1.CreateGraphics();

*// Создаем объекты для закрашивания фигур*

SolidBrush myCorp = [new](http://www.google.com/search?q=new+msdn.microsoft.com) SolidBrush(Color.Black);

SolidBrush myTrum = [new](http://www.google.com/search?q=new+msdn.microsoft.com) SolidBrush(Color.Green);

SolidBrush myTrub = [new](http://www.google.com/search?q=new+msdn.microsoft.com) SolidBrush(Color.Gray);

SolidBrush mySe = [new](http://www.google.com/search?q=new+msdn.microsoft.com) SolidBrush(Color.Blue);

*// Закрашиваем фигуры*

g.FillRectangle(myTrub,300,125,75,75);

g.FillRectangle(myTrub,480,125,75,75);

g.FillPolygon(myCorp, [new](http://www.google.com/search?q=new+msdn.microsoft.com) Point[] {

[new](http://www.google.com/search?q=new+msdn.microsoft.com) Point(100,300),[new](http://www.google.com/search?q=new+msdn.microsoft.com) Point(700,300),

[new](http://www.google.com/search?q=new+msdn.microsoft.com) Point(700,300),[new](http://www.google.com/search?q=new+msdn.microsoft.com) Point(600,400),

[new](http://www.google.com/search?q=new+msdn.microsoft.com) Point(600,400),[new](http://www.google.com/search?q=new+msdn.microsoft.com) Point(200,400),

[new](http://www.google.com/search?q=new+msdn.microsoft.com) Point(200,400),[new](http://www.google.com/search?q=new+msdn.microsoft.com) Point(100,300)

} );

g.FillRectangle(myTrum, 250, 200, 350, 100);

**int** x = 50;

**int** Radius = 50;

*// Море*

**while** (x <= pictureBox1.Width - Radius)

{

g.FillPie(mySe, 0 + x, 375, 50, 50,0, -180);

x += 50;

}

*// Окошки*

**for** (**int** y = 300; y <= 550; y += 50)

{

g.DrawEllipse(myWind, y, 240, 20, 20);

}

}

**Апроксимація β-сплайном.**

Кожен більш або менш складний малюнок складається не тільки з відрізків прямих ліній, кіл і їх дуг, але і з набору кривих ліній. Гладкі криві зручно будувати за допомогою методу згладжування кривої типу β-сплайна. β-сплайн - це гладка крива, або, точніше, крива з безперервними старшими похідними до *n*-ой, де n - порядок сплайна. Відмітимо, що лінія, апроксімована з β-сплайнів, не буде проходити точно через задані точки. Побудова лінії відбувається за допомогою ітераційної процедури.

x(t) = ((a3t + a2)t + a1)t + a0, для 0 <= t <= 1

y(t) = ((b3t + b2)t + b1)t + b0, для 0 <= t <= 1

a3 = (-xi-1 + 3xi - 3xi+1 + xi+2)/6

a2 = (xi-1 - 2xi + xi+1)/2

a1 = (-xi-1 + xi+1)/2

a0 = (xi-1 + 4xi+ xi+1)/6

b3 = (-yi-11 + 3yi - 3yi+1 + yi+2)/6

b2 = (yi-1 - 2yi + yi+1)/2

b1 = (-yi-1 + yi+1)/2

b0 = (yi-1 + 4yi+ yi+1)/6

**Векторний добуток.**

**Означення 1**. Векторним добутком векторів називається третій вектор такий, що

1) перпендикулярний до векторів ;

2) , утворюють праву трійку векторів;

3)

.

Де φ – кут між векторами .

**Інтерполяція поліномом Лангранжа**

Інтерполяція - у обчислювальній математиці це спосіб знаходження проміжних значень величини за наявністю дискретного набору відомих значень.

Задача: за заданим набором точок на площині необхідно побудувати графік функцій, які проходять через ці точки.

Існує кілька способів вирішення цього завдання.

Інтерполяційний многочлен Лангранжа - це многочлен найменшого ступеня, що приймає дані значення в даному наборі точок. Для *n + 1* пар чисел *(x0, y0), (x1, y1), ..., (xn, yn)*, де всі різні змінні, існує єдиний многочлен L(x) ступеня не більше *n*, для якого *L(xj) = yj*

В самому простому випадку *(n = 1)* - це лінійний поліном, графік якого - пряма, яка проходить через дві задані точки.

L (x) = Σyili (x), 1 <= i <= n, де *yi* - це координата i-ої точки, тобто значення функції, яку нам потрібно знайти, в точку *x*i, а li(x) - i-й основний поліном.

***P =*** 1 <= j <= n, j ≠ i П - знак добутка по всім j, задовольняючим заданным умовам.

**Інтерполяція поліномом Ньютона**

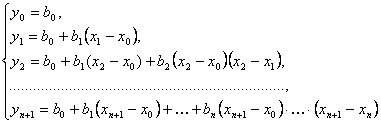
Многочлен Ньютона - як і інші інтерполяційні формули, служить для побудови многочлена n-ї ступеня, яка співпадає в (n+1) з відомими значеннями невідомої функції у=f(x).

Нехай в точках х0, х1, ..., хn + 1 значення функції y = f (x) дорівнюють відповідно у0=f(x0), y1=f(x1), ..., yn+1=f(xn+1).

Побудуємо інтерполяційний поліном Ньютона за допомогою методу невизначених коефіцієнтів. Для цього запишемо многочлен у вигляді:

Pn(x) = b0 + b1(x – x0) + b2(x – x0)(x – x1) + b3(x – x0)(x – x1)(x – x2) + … + bn(x – x0)…(x – xn).(1)

Послідовно підставляючи у формулу (1) замість х дані значення х0, х1, ..., хn + 1, отримаємо для знаходження невизначених коефіцієнтів b0, b1, ..., bn «трикутну» систему рівнянь.

******

При підстановці в равенство (1) замість х числа х0 в правій частині рівності всі доданки обернулися в нуль, крім першого.

Отриману систему зручно вирішити: з першого її рівняння знайдемо b0; підставивши його у друге, знайдемо коефіцієнт b1 і т.д.

***Завдання***

***Завдання на 1 бал***

***Мова програмування C#***

1. ***Використовуючи бібліотеку Drawing намалювати квадрат червоного кольору, товщиною 1 піксель.***
2. ***Використовуючи бібліотеку Drawing намалювати прямокутник синього кольору, товщиною 2 піксель***
3. ***Використовуючи бібліотеку Drawing намалювати квадрат зеленого кольору, товщиною 1 піксель***
4. ***Використовуючи бібліотеку Drawing намалювати коло червоного кольору, товщиною 3 піксель***
5. ***Використовуючи бібліотеку Drawing намалювати еліпс фіолетового кольору, товщиною 1 піксель***
6. ***Використовуючи бібліотеку Drawing намалювати еліпс червоного кольору, товщиною 2 піксель***
7. ***Використовуючи бібліотеку Drawing намалювати коло рожевого кольору, товщиною 1 піксель***
8. ***Використовуючи бібліотеку Drawing намалювати квадрат жовтого кольору, товщиною 4 піксель***
9. ***Використовуючи бібліотеку Drawing намалювати коло чорного кольору, товщиною 2 піксель***
10. ***Використовуючи бібліотеку Drawing намалювати прямокутник червоного кольору, товщиною 1 піксель***

***Завдання на 2 бали:***

***Натискаючи на кнопку міняти колір і товщину лінії:***

1. ***Використовуючи бібліотеку Drawing намалювати квадрат червоного кольору, товщиною 1 піксель.***
2. ***Використовуючи бібліотеку Drawing намалювати прямокутник синього кольору, товщиною 2 піксель***
3. ***Використовуючи бібліотеку Drawing намалювати квадрат зеленого кольору, товщиною 1 піксель***
4. ***Використовуючи бібліотеку Drawing намалювати коло червоного кольору, товщиною 3 піксель***
5. ***Використовуючи бібліотеку Drawing намалювати еліпс фіолетового кольору, товщиною 1 піксель***
6. ***Використовуючи бібліотеку Drawing намалювати еліпс червоного кольору, товщиною 2 піксель***
7. ***Використовуючи бібліотеку Drawing намалювати коло рожевого кольору, товщиною 1 піксель***
8. ***Використовуючи бібліотеку Drawing намалювати квадрат жовтого кольору, товщиною 4 піксель***
9. ***Використовуючи бібліотеку Drawing намалювати коло чорного кольору, товщиною 2 піксель***
10. ***Використовуючи бібліотеку Drawing намалювати прямокутник червоного кольору, товщиною 1 піксель***

***Завдання на 3 бали:***

1. ***Намалювати будиночок з подвір’ям.***
2. ***Намалювати зоряне небо.***
3. ***Намалювати денне небо.***
4. ***Намалювати потяг.***
5. ***Намалювати галактику.***
6. ***Намалювати кораблик.***
7. ***Намалювати куб з тінню.***
8. ***Намалювати шар з тінню.***
9. ***Намалювати піраміду.***
10. ***Намалювати конус.***

***Завдання на 4 бали.***

1. ***Дана множина точок:***

|  |  |
| --- | --- |
| ***x*** | ***y*** |
| ***0*** | ***0*** |
| ***0.5*** | ***0.25*** |
| ***1*** | ***1*** |
| ***1.1*** | ***1.21*** |
| ***1.2*** | ***1.44*** |
| ***2*** | ***4*** |
| ***3*** | ***9*** |

***Використовуючи апроксімацію β-сплайна побудувати криву.***

1. ***Дана множина точок:***

|  |  |
| --- | --- |
| ***x*** | ***y*** |
| ***0*** | ***0*** |
| ***0.5*** | ***0.25*** |
| ***1*** | ***1*** |
| ***1.1*** | ***1.21*** |
| ***1.2*** | ***1.44*** |
| ***2*** | ***4*** |
| ***3*** | ***9*** |

***Використовуючи інтерполяцію Лагранжа побудувати криву.***

1. ***Дана множина точок:***

|  |  |
| --- | --- |
| ***x*** | ***y*** |
| ***0*** | ***0*** |
| ***0.5*** | ***0.25*** |
| ***1*** | ***1*** |
| ***1.1*** | ***1.21*** |
| ***1.2*** | ***1.44*** |
| ***2*** | ***4*** |
| ***3*** | ***9*** |

***Використовуючи інтерполяцію Ньютона побудувати криву.***

1. ***Дана множина точок:***

|  |  |
| --- | --- |
| ***x*** | ***y*** |
| ***1*** | ***5*** |
| ***2*** | ***2.5*** |
| ***3*** | ***1.66*** |
| ***4*** | ***1.25*** |
| ***6*** | ***0.83*** |
| ***7*** | ***0.714*** |
| ***8*** | ***0.625*** |

***Використовуючи апроксімацію β-сплайна побудувати криву.***

1. ***Дана множина точок:***

|  |  |
| --- | --- |
| ***x*** | ***y*** |
| ***1*** | ***5*** |
| ***2*** | ***2.5*** |
| ***3*** | ***1.66*** |
| ***4*** | ***1.25*** |
| ***6*** | ***0.83*** |
| ***7*** | ***0.714*** |
| ***8*** | ***0.625*** |

***Використовуючи інтерполяцію Лагранжа побудувати криву.***

1. ***Дана множина точок:***

|  |  |
| --- | --- |
| ***x*** | ***y*** |
| ***1*** | ***5*** |
| ***2*** | ***2.5*** |
| ***3*** | ***1.66*** |
| ***4*** | ***1.25*** |
| ***6*** | ***0.83*** |
| ***7*** | ***0.714*** |
| ***8*** | ***0.625*** |

***Використовуючи інтерполяцію Ньютона побудувати криву.***

1. ***Дана множина точок:***

|  |  |
| --- | --- |
| ***x*** | ***y*** |
| ***1*** | ***2.72*** |
| ***2*** | ***7.39*** |
| ***3*** | ***20.09*** |
| ***4*** | ***54.6*** |
| ***5*** | ***148.41*** |
| ***6*** | ***403.43*** |

***Використовуючи апроксімацію β-сплайна побудувати криву.***

1. ***Дана множина точок:***

|  |  |
| --- | --- |
| ***x*** | ***y*** |
| ***1*** | ***2.72*** |
| ***2*** | ***7.39*** |
| ***3*** | ***20.09*** |
| ***4*** | ***54.6*** |
| ***5*** | ***148.41*** |
| ***6*** | ***403.43*** |

***Використовуючи інтерполяцію Лагранжа побудувати криву.***

1. ***Дана множина точок:***

|  |  |
| --- | --- |
| ***x*** | ***y*** |
| ***1*** | ***2.72*** |
| ***2*** | ***7.39*** |
| ***3*** | ***20.09*** |
| ***4*** | ***54.6*** |
| ***5*** | ***148.41*** |
| ***6*** | ***403.43*** |

***Використовуючи інтерполяцію Ньютона побудувати криву.***

СПИСОК ДОДАТКОВИХ ДЖЕРЕЛ

1. Christian Nagel. [[Professional C# 7 and .NET Core 2.0. Wrox; 7 edition (April 17, 2018).](https://www.amazon.com/Professional-NET-Core-Christian-Nagel/dp/1119449278?tag=uuid10-20)](https://www.amazon.com/Microsoft-Visual-Step-Developer-Reference/dp/1509307761?tag=uuid10-20) Pp.1440

2. John Sharp. Microsoft Visual C# Step by Step (9th Edition) (Developer Reference) 9th Edition**.** Microsoft Press; 9 edition (July 5, 2018). Pp.832.

3.John Paul Mueller.C# 7.0 All-in-One For Dummies 1st Edition . For Dummies; 1 edition (December 26, 2017). Pp.864.

# 4. Gaurav Aroraa. Learn C# in 7 days: Get up and running with C# 7 with async main, tuples, pattern matching, LINQ, regex, indexers, and more Paperback – October 6, 2017 .Pp.306

# 5. Enrico Buonanno .Functional Programming in C#: How to write better C# code 1st Edition. Publisher: Manning Publications; 1 edition (September 17, 2017).Pp. 408