

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний аерокосмічний університет ім. М.Є. Жуковського
«Харківський авіаційний інститут»

Факультет радіоелектроніки, комп'ютерних систем та інфокомунікацій

Кафедра комп'ютерних систем, мереж і кібербезпеки (503)

Лабораторна робота № 7

*Робота з файлами та серіалізація. Робота з текстом та
регулярними виразами*

(назва лабораторної роботи)

з дисципліни

Технології програмування

(шифр)

XAI.503.525a.03O.123-Комп'ютерна інженерія, ПЗ №9629619

Виконав студент гр.
11.12.2022

525a
(№ групи)

Литвиненко А.В.
(П.І.Б.)

(підпис, дата)

Перевірив

канд. техн. наук, доцент

(підпис, дата)

Є. В. Бабешенко
(П.І.Б.)

Харків – 2022

Тема роботи: робота з файлами та серіалізація. Робота з текстом та регулярними виразами.

Мета роботи:

Вивчити можливості роботи з файлами. Освоїти практично переваги використання серіалізації.

Вивчити можливості роботи з рядками. Закріпити оброблення параметрів командного рядка. Освоїти використання класів `StreamReader` та `StreamWriter`. Навчитися працювати з регулярними виразами.

Варіант 5

Задача 1

Частина 1. Постановка завдання

Умова:

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

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

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

Реалізувати такі функції:

- додавання нового запису до колекції;
- видалення запису з колекції;
- перегляд усіх записів;
- зміна будь-якого запису;
- пошук записів по кожному з полів;
- експорт будь-якого запису до текстового файлу.

Програма має зберігати всі введені/змінені дані під час виходу з програми, якщо обрано пункт меню «Вихід». При запуску програма має перевіряти наявність файлу з даними та завантажувати їх автоматично, якщо файл, створений у попередньому сеансі роботи, доступний та коректний. Збереження даних має бути реалізоване з використанням механізму серіалізації, завантаження – з використанням механізму десеріалізації.

Також у програмі необхідно передбачити оброблення можливих винятків (наприклад, файл з даними відсутній, пошкоджений, введені некоректні дані користувачем тощо).

Умова з додатка:

5.	Курс валют Монобанк на поточну дату	JSON	https://api.monobank.ua/bank/currency
----	-------------------------------------	------	---

<https://api.monobank.ua/bank/currency> - лінк на файл

Частина 2. Схема класів

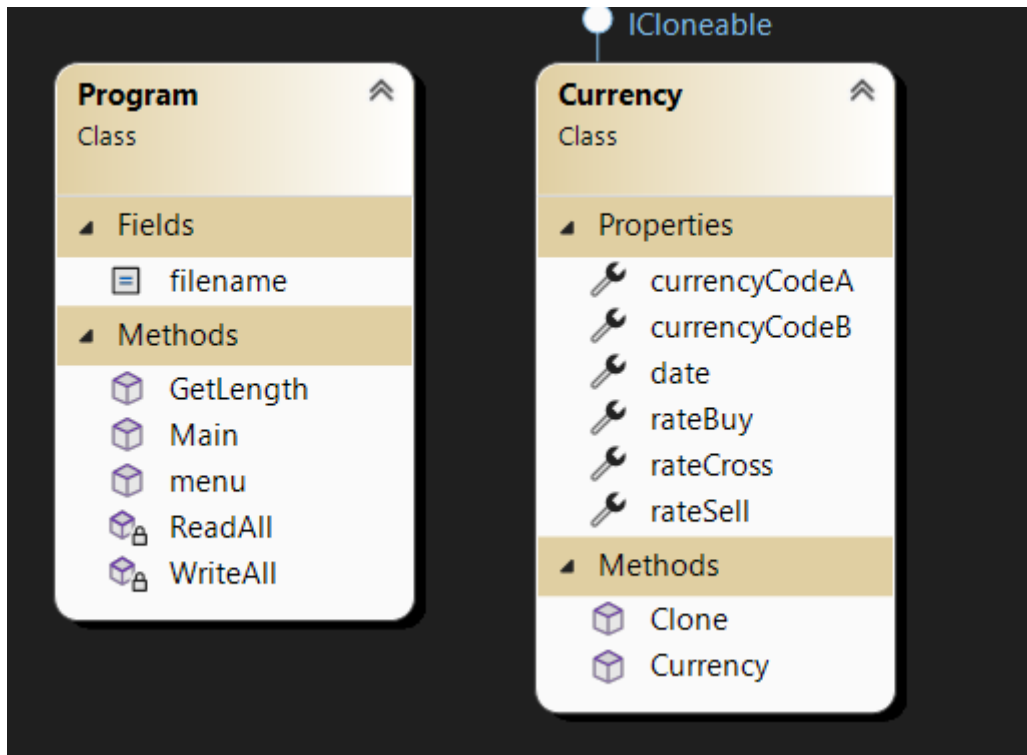


Рисунок 1 - Схема класів

Частина 3. Текст програми

Відповідно до розробленого алгоритму в середовищі Microsoft Visual Studio була написана програма, яка наведена нижче.

Program.cs

```
using Newtonsoft.Json;
using System.IOEnumeration;
using System.Net.Http.Json;
using System.Reflection.Metadata.Ecma335;
using System.Runtime.Serialization.Json;

namespace Task_1
{
    internal class Program
    {
        public const string filename = "currency.json";

        static void WriteAll(List<Currency> instances, string file)
        {
            string serializedUsers = JsonConvert.SerializeObject(instances);

            File.WriteAllText(file, serializedUsers);
        }

        static List<Currency> ReadAll()
        {
            string json = File.ReadAllText(filename);

            List<Currency> instances =
            JsonConvert.DeserializeObject<List<Currency>>(json)!;

            return instances;
        }

        static public int GetLength(Currency[] arr)
        {
            int i = 0;
            while (arr[i] != null)
            {
                i++;
            }
            return i;
        }

        static public void menu()
```

```

{
    Console.WriteLine("\t===== Menu =====");
    Console.WriteLine("\t0. This menu"); // +
    Console.WriteLine("\t1. Show content"); // +
    Console.WriteLine("\t2. Add record"); // +
    Console.WriteLine("\t3. Delete record"); // +
    Console.WriteLine("\t4. Change record"); // +
    Console.WriteLine("\t5. Find record"); // +
    Console.WriteLine("\t6. Export to another file");
    Console.WriteLine("\t7. Save and exit");
    Console.WriteLine("\t8. Exit (without saving)"); // +
    Console.WriteLine("\t=====");
}

public static void Main(string[] args)
{
    List<Currency> array = ReadAll();

    // Infinity cycle or MENU
    int op = 0;
    int i = 0;
    char yn;
    int user_value;
    int field_id = 0;
    string value = "";
    Currency t = new Currency(0,0,0,0,0,0);
    while (true)
    {
        switch (op)
        {

            // Prints menu
            case 0:
                menu();
                break;

            // Show structured file content
            case 1:
                i = 0;
                foreach(Currency c in array)
                {

```

```

Console.WriteLine(String.Format("===== {0} =====", i));
    Console.WriteLine("CurrencyCodeA = " +
c.currencyCodeA.ToString());
    Console.WriteLine("CurrencyCodeB = " +
c.currencyCodeB.ToString());
    Console.WriteLine("Date = " + c.date.ToString());
    if (c.rateCross == 0)
    {
        Console.WriteLine("RateBuy = " + c.rateBuy.ToString());
        Console.WriteLine("RateSell = " + c.rateSell.ToString());
    }
    else
    {
        Console.WriteLine("RateCross = " + c.rateCross.ToString());
    }
    Console.WriteLine("===== \n");

    i++;
}
Console.WriteLine("Totally: " + i.ToString());
break;

// Add record to array
case 2:
    Console.WriteLine("\t[Add a record]");
    Console.WriteLine("If you want to skip a field, enter: 0");

    Console.Write("currencyCodeA = ");
    t.currencyCodeA = int.Parse(Console.ReadLine());

    Console.Write("currencyCodeB = ");
    t.currencyCodeB = int.Parse(Console.ReadLine());

    Console.Write("date = ");
    t.date = int.Parse(Console.ReadLine());

    Console.Write("rateBuy = ");
    t.rateBuy = double.Parse(Console.ReadLine());

    Console.Write("rateSell = ");
    t.rateSell = double.Parse(Console.ReadLine());

```

```

        Console.WriteLine("rateCross = ");
        t.rateCross = double.Parse(Console.ReadLine());

        array.Add((Currency) t.Clone());

        Console.WriteLine("[OK]");

        break;

// Remove record in array
case 3:
    Console.WriteLine("[Enter a record index] >> ");
    user_value = int.Parse(Console.ReadLine());

    // numbers = numbers.Where((val, idx) => idx !=
numIndex).ToArray();
    array.Remove(array[user_value]);

    Console.WriteLine("[OK]");

    break;

// Change record in array
case 4:
    Console.WriteLine("[Enter a record index] >> ");
    user_value = int.Parse(Console.ReadLine());

    t = array[user_value];

    // Print object information
    Console.WriteLine("==== [ INFO ] =====");
    Console.WriteLine("1. CurrencyCodeA = " +
t.currencyCodeA.ToString());
    Console.WriteLine("2. CurrencyCodeB = " +
t.currencyCodeB.ToString());
    Console.WriteLine("3. Date = " + t.date.ToString());
    if (t.rateCross == 0)
    {
        Console.WriteLine("4. RateBuy = " + t.rateBuy.ToString());
        Console.WriteLine("5. RateSell = " + t.rateSell.ToString());
    }
    else

```

```

    {
        Console.WriteLine("4. RateCross = " + t.rateCross.ToString());
    }
    Console.WriteLine("=====\n");

```

```

Console.Write("[Field's number] >> ");
field_id = int.Parse(Console.ReadLine());
Console.Write("[Field's value] >> ");
value = Console.ReadLine();

```

```

switch (field_id)
{
    case 1:
        t.currencyCodeA = int.Parse(value);
        break;
    case 2:
        t.currencyCodeB = int.Parse(value);
        break;
    case 3:
        t.date = int.Parse(value);
        break;
    case 4:
        if (t.rateBuy == 0)
        {
            t.rateCross = double.Parse(value);
        }
        else
        {
            t.rateBuy = double.Parse(value);
        }
        break;
    case 5:
        if (t.rateSell == 0)
        {
            break;
        }
        else
        {
            t.rateSell = double.Parse(value);
        }
        break;
    default:
        Console.WriteLine("[ERR] This field doesn't exist!");

```



```

        break;
    }

    Console.WriteLine("[OK]");

    break;

// Find record in array
case 5:
    Console.WriteLine("Fields to search:");
    Console.WriteLine("\t1. currencyCodeA");
    Console.WriteLine("\t2. currencyCodeB");
    Console.WriteLine("\t3. date");
    Console.WriteLine("\t4. rateBuy");
    Console.WriteLine("\t5. rateSell");
    Console.WriteLine("\t6. rateCross");

    Console.Write("[Field's number] >> ");
    field_id = int.Parse(Console.ReadLine());

    Console.Write("[Value to find] >> ");
    value = Console.ReadLine();

    bool flag = true;
    i = 0;
    foreach(Currency c in array)
    {
        if (!flag)
        {
            break;
        }

        switch (field_id)
        {
            // currencyCodeA
            case 1:
                if (c.currencyCodeA == int.Parse(value))
                {
                    flag = false;
                }
                break;

            // currencyCodeB
            case 2:

```

```

        if (c.currencyCodeB == int.Parse(value))
        {
            flag = false;
        }
        break;

// date
case 3:
    if (c.date == int.Parse(value))
    {
        flag = false;
    }
    break;

// rateBuy
case 4:
    if (c.rateBuy == double.Parse(value))
    {
        flag = false;
    }
    break;

// rateSell
case 5:
    if (c.rateSell == double.Parse(value))
    {
        flag = false;
    }
    break;

// rateCross
case 6:
    if (c.rateCross == double.Parse(value))
    {
        flag = false;
    }
    break;
default:
    Console.WriteLine("[ERR] Invalid field's number!");
    break;
}

if (!flag)

```

```

        {
            Console.WriteLine();
            Console.WriteLine(">>> [Found!] <<<<");
            Console.WriteLine("Index: " + i.ToString());
            Console.WriteLine("CurrencyCodeA = " +
t.currencyCodeA.ToString());
            Console.WriteLine("CurrencyCodeB = " +
t.currencyCodeB.ToString());
            Console.WriteLine("Date = " + t.date.ToString());
            if (t.rateCross == 0)
            {
                Console.WriteLine("RateBuy = " + t.rateBuy.ToString());
                Console.WriteLine("RateSell = " + t.rateSell.ToString());
            }
            else
            {
                Console.WriteLine("RateCross = " + t.rateCross.ToString());
            }
            Console.WriteLine("\n");
        }
        i++;
    }
    if (flag)
    {
        Console.WriteLine("[ERR] 404 not found\n");
    }

    break;

// Export file content to another file
case 6:
    Console.Write("Enteer filename to export: ");
    value = Console.ReadLine();

    WriteAll(array,value);
    Console.WriteLine("OK");
    break;

// Save data and exit
case 7:
    // saving user data to currency.json
    WriteAll(array, filename);

```

```

        Console.WriteLine("Ok");
        return;

// Just exit without saving the file
case 8:
    Console.Write("[INFO] You may lost of your data. Are you sure?
(y/N): ");
    yn = Console.ReadLine()[0];

    if (yn == 'y')
    {
        return;
    }
    else
    {
        Console.WriteLine("[INFO] Your data was saved from destruction!
:");
    }

    break;
default:
    Console.WriteLine("[INFO] Invalid operation!");
    break;
}
Console.Write("[OP] >> ");
try
{
    op = int.Parse(Console.ReadLine());
}
catch (FormatException)
{
    Console.WriteLine("[ERR] Invalid format!");
    op = 0;
}
}

return;
}
}
}

```

Class1.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Runtime.Serialization.Json;
using System.Runtime.Serialization;
using System.Text.Json.Serialization;
using System.ComponentModel;

namespace Task_1
{
    class Currency : ICloneable
    {
        public int currencyCodeA { get; set; }
        public int currencyCodeB { get; set; }
        public int date { get; set; }
        public double rateBuy { get; set; }
        public double rateSell { get; set; }
        public double rateCross { get; set; }

        public Currency(int a, int b, int d, double buy, double sell, double cross)
        {
            currencyCodeA = a;
            currencyCodeB = b;
            date = d;
            rateBuy = buy;
            rateSell = sell;
            rateCross = cross;
        }

        public object Clone()
        {
            return this.MemberwiseClone();
        }
    }
}
```

Частина 4. Тестування

```
C:\Users\user\Desktop\Sharp_7\Sharp_7\bin\Debug\net6.0\Task_1.exe

===== Menu =====
0. This menu
1. Show content
2. Add record
3. Delete record
4. Change record
5. Find record
6. Export to another file
7. Save and exit
8. Exit (without saving)
=====
[OP] >> _
```

Рисунок 2 – Меню програми

```
=====111=====
CurrencyCodeA = 886
CurrencyCodeB = 980
Date = 1543715495
RateCross = 0.112
=====

=====112=====
CurrencyCodeA = 710
CurrencyCodeB = 980
Date = 1670685757
RateCross = 2.1907
=====

=====113=====
CurrencyCodeA = 894
CurrencyCodeB = 980
Date = 1670623274
RateCross = 0.0021
=====

Totally: 114
[OP] >>
```

Рисунок 3 – Показ усіх значень

<pre>[OP] >> 2 [Add a record] If you want to skip a field, enter: 0 currencyCodeA = 111 currencyCodeB = 222 date = 354354332 rateBuy = 0.2344242 rateSell = 2.525234 rateCross = 0 [OK] [OP] >> 2 [Add a record] If you want to skip a field, enter: 0 currencyCodeA = 555 currencyCodeB = 666 date = 234243423 rateBuy = 0 rateSell = 0 rateCross = 9.35332 [OK] [OP] >> .</pre>	<pre>=====113===== CurrencyCodeA = 894 CurrencyCodeB = 980 Date = 1670623274 RateCross = 0.0021 ===== =====114===== CurrencyCodeA = 111 CurrencyCodeB = 222 Date = 354354332 RateBuy = 0.2344242 RateSell = 2.525234 ===== =====115===== CurrencyCodeA = 555 CurrencyCodeB = 666 Date = 234243423 RateCross = 9.35332 =====</pre>
---	---

Рисунок 4 – Додавання значень та перевірка

<pre>[OP] >> 3 [Enter a record index] >> 114 [OK] [OP] >></pre>	<pre>=====113===== CurrencyCodeA = 894 CurrencyCodeB = 980 Date = 1670623274 RateCross = 0.0021 ===== =====114===== CurrencyCodeA = 555 CurrencyCodeB = 666 Date = 234243423 RateCross = 9.35332 ===== Totally: 115 [OP] >></pre>
---	---

Рисунок 5 – Видалення значення з індексом 114

```
[OP] >> 4
[Enter a record index] >> 114
===== [ INFO ] =====
1. CurrencyCodeA = 555
2. CurrencyCodeB = 666
3. Date = 234243423
4. RateCross = 9.35332
=====

[Field's number] >> 3
[Field's value] >> 111222333
[OK]
[OP] >>
```

Рисунок 6 – Зміна значення date елемента з індексом 114

<pre>[OP] >> 4 [Enter a record index] >> 113 ===== [INFO] ===== 1. CurrencyCodeA = 894 2. CurrencyCodeB = 980 3. Date = 1670623274 4. RateCross = 0.0021 ===== [Field's number] >> 1 [Field's value] >> 777 [OK] [OP] >></pre>	<pre>=====113===== CurrencyCodeA = 777 CurrencyCodeB = 980 Date = 1670623274 RateCross = 0.0021 ===== =====114===== CurrencyCodeA = 555 CurrencyCodeB = 666 Date = 111222333 RateCross = 9.35332 ===== Totally: 115 [OP] >></pre>
---	---

Рисунок 7 – Зміна значення currencyCodeA елемента з індексом 113

<pre>[OP] >> 5 Fields to search: 1. currencyCodeA 2. currencyCodeB 3. date 4. rateBuy 5. rateSell 6. rateCross [Field's number] >> 1 [Value to find] >> 777 >>> [Found!] <<<< Index: 113 CurrencyCodeA = 777 CurrencyCodeB = 980 Date = 1670623274 RateCross = 0.0021</pre>	<pre>[OP] >> 5 Fields to search: 1. currencyCodeA 2. currencyCodeB 3. date 4. rateBuy 5. rateSell 6. rateCross [Field's number] >> 3 [Value to find] >> 111222333 >>> [Found!] <<<< Index: 114 CurrencyCodeA = 555 CurrencyCodeB = 666 Date = 111222333 RateCross = 9.35332</pre>
---	---

Рисунок 8 – Пошук за значенням


```

=====112=====
CurrencyCodeA = 710
CurrencyCodeB = 980
Date = 1670685757
RateCross = 2.1907
=====

=====113=====
CurrencyCodeA = 555
CurrencyCodeB = 666
Date = 111222333
RateBuy = 9.35332
RateSell = 0
=====

=====114=====
CurrencyCodeA = 555
CurrencyCodeB = 666
Date = 111222333
RateCross = 9.35332
=====

```

Рисунок 9 – Як виглядають останні елементи після змін

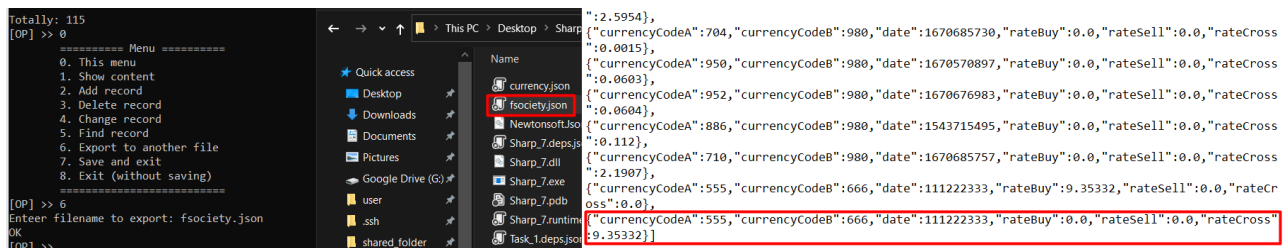


Рисунок 10 – Експорт в інший файл

```
currency.json - Notepad
File Edit Format View Help
{
  "date": 1670570897,
  "rateCross": 0.0603
},
{
  "currencyCodeA": 952,
  "currencyCodeB": 980,
  "date": 1670676983,
  "rateCross": 0.0604
},
{
  "currencyCodeA": 886,
  "currencyCodeB": 980,
  "date": 1543715495,
  "rateCross": 0.112
},
{
  "currencyCodeA": 710,
  "currencyCodeB": 980,
  "date": 1670685757,
  "rateCross": 2.1907
},
{
  "currencyCodeA": 894,
  "currencyCodeB": 980,
  "date": 1670623274,
  "rateCross": 0.0021
}
}
```

Рисунок 11 – Файл до збереження

```
Microsoft Visual Studio Debug Console
=====114=====
CurrencyCodeA = 555
CurrencyCodeB = 666
Date = 111222333
RateCross = 9.35332
=====
Totally: 115
[OP] >> 0
===== Menu =====
0. This menu
1. Show content
2. Add record
3. Delete record
4. Change record
5. Find record
6. Export to another file
7. Save and exit
8. Exit (without saving)
=====
[OP] >> 6
Enter filename to export: fsociety.json
Ok
[OP] >> 7
Ok

C:\Users\User\Desktop\Sharp_7\Sharp_7\bin\Debug\
To automatically close the console when debugging
To manually close the console when debugging stops.
Press any key to close this window

currency.json - Notepad
File Edit Format View Help
{"currencyCodeA":949,"currencyCodeB":980,"date":1670686299,"rateBuy":0.0,"rateSell":0.0,"rateCross":2.0111},
{"currencyCodeA":901,"currencyCodeB":980,"date":1670680475,"rateBuy":0.0,"rateSell":0.0,"rateCross":1.2224},
{"currencyCodeA":834,"currencyCodeB":980,"date":1670613800,"rateBuy":0.0,"rateSell":0.0,"rateCross":0.0163},
{"currencyCodeA":800,"currencyCodeB":980,"date":1670655931,"rateBuy":0.0,"rateSell":0.0,"rateCross":0.0101},
{"currencyCodeA":858,"currencyCodeB":980,"date":1670630809,"rateBuy":0.0,"rateSell":0.0,"rateCross":0.9473},
{"currencyCodeA":860,"currencyCodeB":980,"date":1670686047,"rateBuy":0.0,"rateSell":0.0,"rateCross":0.0033},
{"currencyCodeA":937,"currencyCodeB":980,"date":1670623274,"rateBuy":0.0,"rateSell":0.0,"rateCross":2.5954},
{"currencyCodeA":704,"currencyCodeB":980,"date":1670685730,"rateBuy":0.0,"rateSell":0.0,"rateCross":0.0015},
{"currencyCodeA":950,"currencyCodeB":980,"date":1670570897,"rateBuy":0.0,"rateSell":0.0,"rateCross":0.0603},
{"currencyCodeA":952,"currencyCodeB":980,"date":1670676983,"rateBuy":0.0,"rateSell":0.0,"rateCross":0.0604},
{"currencyCodeA":886,"currencyCodeB":980,"date":1543715495,"rateBuy":0.0,"rateSell":0.0,"rateCross":0.112},
{"currencyCodeA":710,"currencyCodeB":980,"date":1670685757,"rateBuy":0.0,"rateSell":0.0,"rateCross":2.1907},
{"currencyCodeA":555,"currencyCodeB":666,"date":111222333,"rateBuy":9.35332,"rateSell":0.0,"rateCross":0.0},
{"currencyCodeA":894,"currencyCodeB":980,"date":1670623274,"rateBuy":0.0,"rateSell":0.0,"rateCross":0.0021}
}
```

Рисунок 12 – Файл після збереження

Задача 2

Частина 1. Постановка завдання

Умова:

Розробити консольну програму, призначену для перевірки рядків на відповідність заданому шаблону. Результатом роботи програми має бути виведення інформації про те, які рядки відповідають заданому шаблону, а які – ні.

Якщо користувач вказав ключ /s, програма має перевірити всі слова, передані після ключа в командному рядку.

Якщо користувач вказав ключ /f та ім'я файлу, програма має перевірити всі слова у вказаному файлі.

Якщо користувач вказав ключ /? або /h, програма має вивести довідкову інформацію про її призначення та опис усіх ключів.

Якщо користувач запустив програму без ключів, вона має вимагати рядки у користувача та виводити результати перевірки; введення рядків завершується, якщо користувач ввів рядок "end".

Шаблон повинен бути поставлений за допомогою регулярного виразу.

Варіанти індивідуальних завдань наведено у додатку Б.

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

Умова з додатка:

5.	Рядки, що містять коректні номери телефонів мобільних операторів України. Приклади рядків, які підходять: «(050) 310-20-89», «0971125678», «073-123-45-76». Приклади рядків, які не підходять: «0112345678», «(077) 212-45-08», «09548567нуль».
----	---

Частина 2. Схема класів

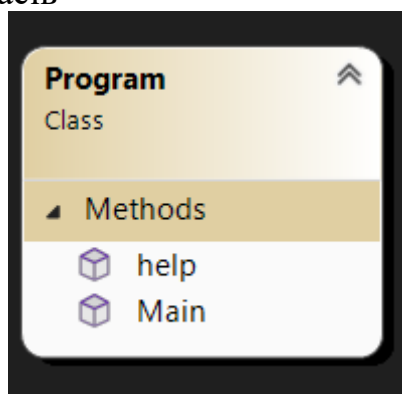


Рисунок 13 - Схема класів

Частина 3. Розробка тестів

Таблиця 3 – Тестові набори

№	Вхідні дані	Очікуваний результат
1	(050) 310-20-89	OK
2	0971125678	OK
3	073-123-45-76	OK
4	0112345678	BAD
5	(077)212-45-08	BAD
6	09548567нуль	BAD

Частина 4. Текст програми

Program.cs

```
using System.Text.RegularExpressions;

namespace Task_2
{
    internal class Program
    {
        static public void help()
        {
            Console.WriteLine("Task 2 made by Andrii Lytvynenko (C) 2022" +
                "\n\t - /s -- search in a list of words" +
                "\n\t - /f -- search in a list of files" +
                "\n\t - /? /h -- this help message");
        }

        public static void Main(string[] args)
        {
            string pattern = @"(\(050\)\\d{3}-\\d{2}-\\d{2})|(097\\d{7})|(073-\\d{3}-\\d{2}-\\d{2})";
            string argument;
            string tmp = "";

            try
            {
                argument = args[0];
            }
            catch (IndexOutOfRangeException)
            {
                Console.WriteLine("[ERR] No command and values!");
                return;
            }

            List<string> values = args.ToList();
            values = values.GetRange(1, values.Count-1);

            if(values.Count == 0 && !(argument == "/" || argument == "/h"))
            {
                // user must enter values
                Console.WriteLine("Enter values. If you want to exit, you'll enter 'end' key :)");
                while(true)
                {
                    tmp = Console.ReadLine();

                    if(tmp == "end")
```

```

        {
            break;
        }
        values.Add(tmp);
    }
}

Console.WriteLine("\n===== Regex check begins! =====\n");
switch (argument)
{
    case "/s":
        foreach (string word in values)
        {
            if (Regex.IsMatch(word, pattern))
            {
                Console.WriteLine(String.Format("{0}' is good! :)",
word));
            }
            else
            {
                Console.WriteLine(String.Format("{0}' is BAAAAAAD! >:(",
word));
            }
        }
        break;

    case "/f":
        foreach (string file in values)
        {
            try
            {
                string[] text = File.ReadAllLines(file);
                foreach (string word in text)
                {
                    if (Regex.IsMatch(word, pattern))
                    {
                        Console.WriteLine(String.Format("{0}' is good!
:)", word));
                    }
                    else
                    {
                        Console.WriteLine(String.Format("{0}' is
BAAAAAAD! >:(", word));
                    }
                }
            }
            catch (FileNotFoundException)
            {
                Console.WriteLine(String.Format("[ERR] '{0}' not found!",
file));
            }
            catch (ArgumentException)
            {
                Console.WriteLine("[ERR] Empty filename!");
            }
        }
        break;

    case "/?":
    case "/h":
        help();
        break;
}

```

```
        }  
  
        return;  
    }  
}
```

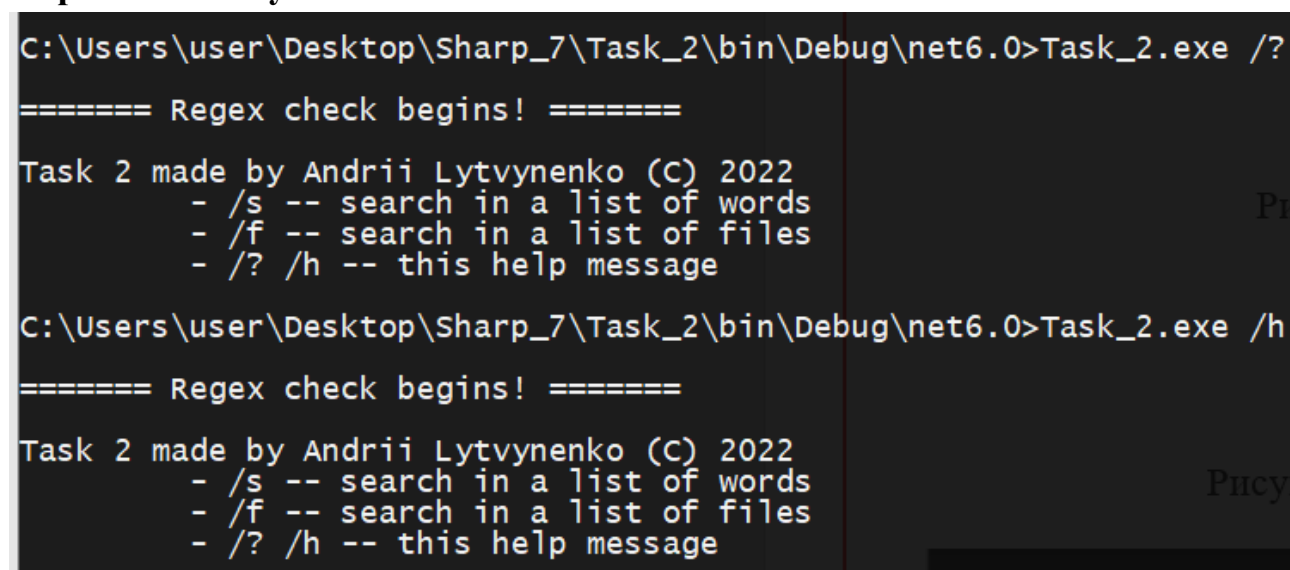
Частина 5. Тестування

Результати тестування наведені в таблиці 4.

Таблиця 4 – Результати тестування

Номер тесту	Вхідні дані	Очікуваний результат	Отриманий результат	Результат тестування
1	(050) 310-20-89	OK	OK	OK
2	0971125678	OK	OK	OK
3	073-123-45-76	OK	OK	OK
4	0112345678	BAD	BAD	OK
5	(077)212-45-08	BAD	BAD	OK
6	09548567нуль	BAD	BAD	OK

Скриншот тестування:



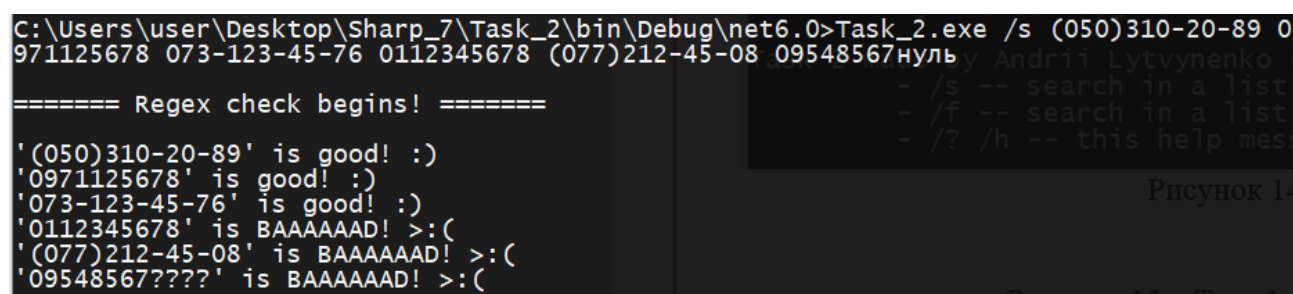
```
C:\Users\user\Desktop\Sharp_7\Task_2\bin\Debug\net6.0>Task_2.exe /?
===== Regex check begins! =====

Task 2 made by Andrii Lytvynenko (C) 2022
- /s -- search in a list of words
- /f -- search in a list of files
- /? /h -- this help message

C:\Users\user\Desktop\Sharp_7\Task_2\bin\Debug\net6.0>Task_2.exe /h
===== Regex check begins! =====

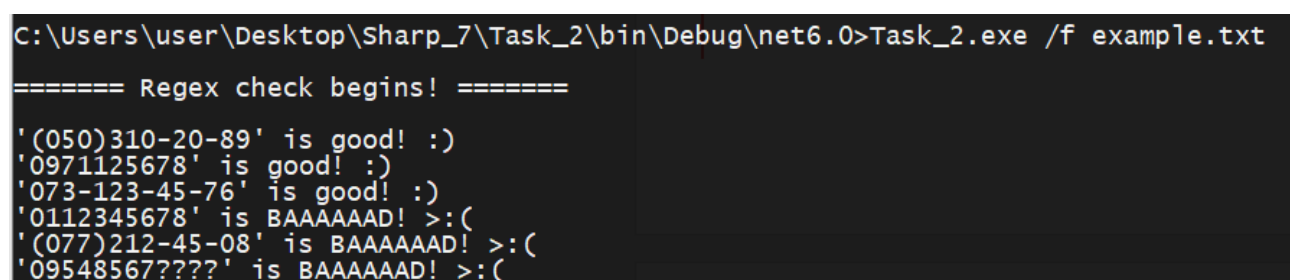
Task 2 made by Andrii Lytvynenko (C) 2022
- /s -- search in a list of words
- /f -- search in a list of files
- /? /h -- this help message
```

Рисунок 14 – Довідка



```
C:\Users\user\Desktop\Sharp_7\Task_2\bin\Debug\net6.0>Task_2.exe /s (050)310-20-89 0
971125678 073-123-45-76 0112345678 (077)212-45-08 09548567нуль
===== Regex check begins! =====
'(050)310-20-89' is good! :)
'0971125678' is good! :)
'073-123-45-76' is good! :)
'0112345678' is BAAAAAAD! >:(
'(077)212-45-08' is BAAAAAAD! >:(
'09548567????' is BAAAAAAD! >:(
```

Рисунок 15 – Тест 1-6 з командного рядка



```
C:\Users\user\Desktop\Sharp_7\Task_2\bin\Debug\net6.0>Task_2.exe /f example.txt
===== Regex check begins! =====
'(050)310-20-89' is good! :)
'0971125678' is good! :)
'073-123-45-76' is good! :)
'0112345678' is BAAAAAAD! >:(
'(077)212-45-08' is BAAAAAAD! >:(
'09548567????' is BAAAAAAD! >:(
```

Рисунок 16 – Тест 1-6 з файлу

```

C:\Users\user\Desktop\Sharp_7\Task_2\bin\Debug\net6.0>Task_2.exe /s
Enter values. If you want to exit, you'll enter 'end' key :)
(050)310-20-89
0971125678
073-123-45-76
0112345678
(077)212-45-08
09548567нуль
end

===== Regex check begins! =====

'(050)310-20-89' is good! :)
'0971125678' is good! :)
'073-123-45-76' is good! :)
'0112345678' is BAAAAAAD! >:(
'(077)212-45-08' is BAAAAAAD! >:(
'09548567????' is BAAAAAAD! >:(

```

Рисунок 17 – Тест 1-6 з введення у програмі

```

Enter values. If you want to exit, you'll enter 'end' key :)
example.txt
egoejroigieg.txt
lalalala.txt
xd.txt
end

===== Regex check begins! =====

'(050)310-20-89' is good! :)
'0971125678' is good! :)
'073-123-45-76' is good! :)
'0112345678' is BAAAAAAD! >:(
'(077)212-45-08' is BAAAAAAD! >:(
'09548567????' is BAAAAAAD! >:(
[ERR] 'egoejroigieg.txt' not found!
[ERR] 'lalalala.txt' not found!
[ERR] Empty filename!
[ERR] 'xd.txt' not found!

```

Рисунок 18 – Тест 1-6 з декількох файлів файлу

```

C:\Users\user\Desktop\Sharp_7\Task_2\bin\Debug\net6.0>Task_2.exe
[ERR] No command and values!

```

Рисунок 19 – Перехоплення помилки

Висновки

Під час цієї лабораторної роботи я навчився користуватися файлами, зумів їх відкривати, читати, перезаписувати та закривати. Також закріпив практичні переваги використання серіалізації. У другій частині я покращив свої знання з використання рядків, закріпив оброблення параметрів командного рядка і навчився користуватися регулярними виразами.