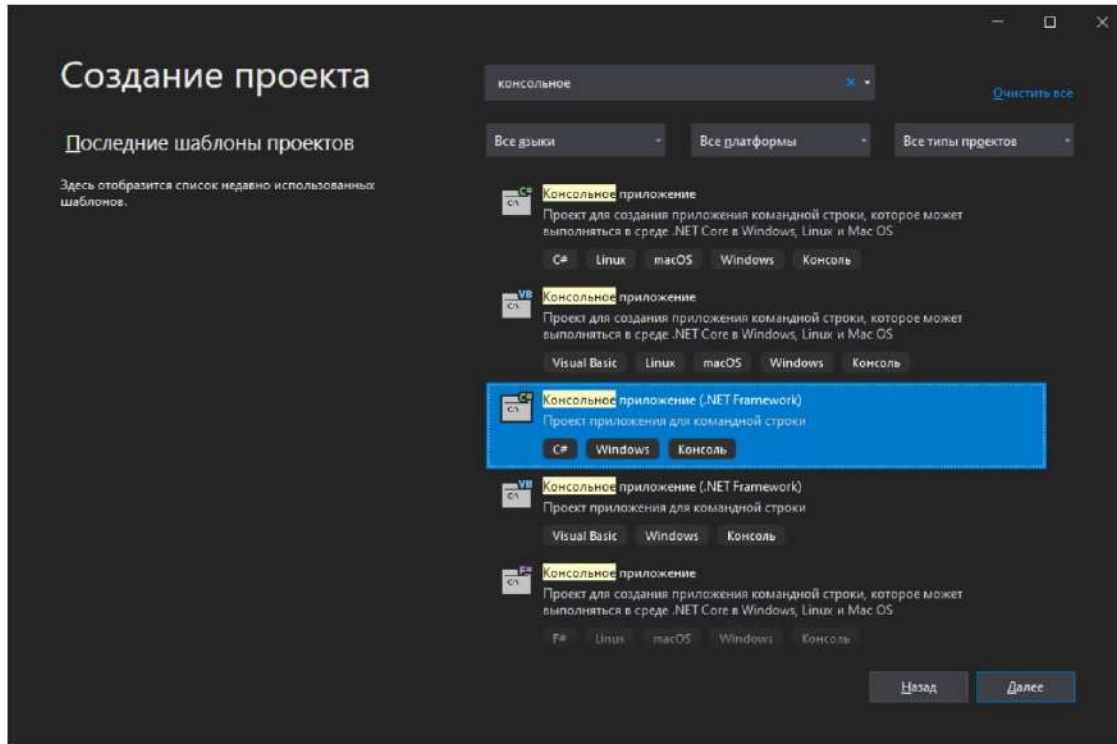


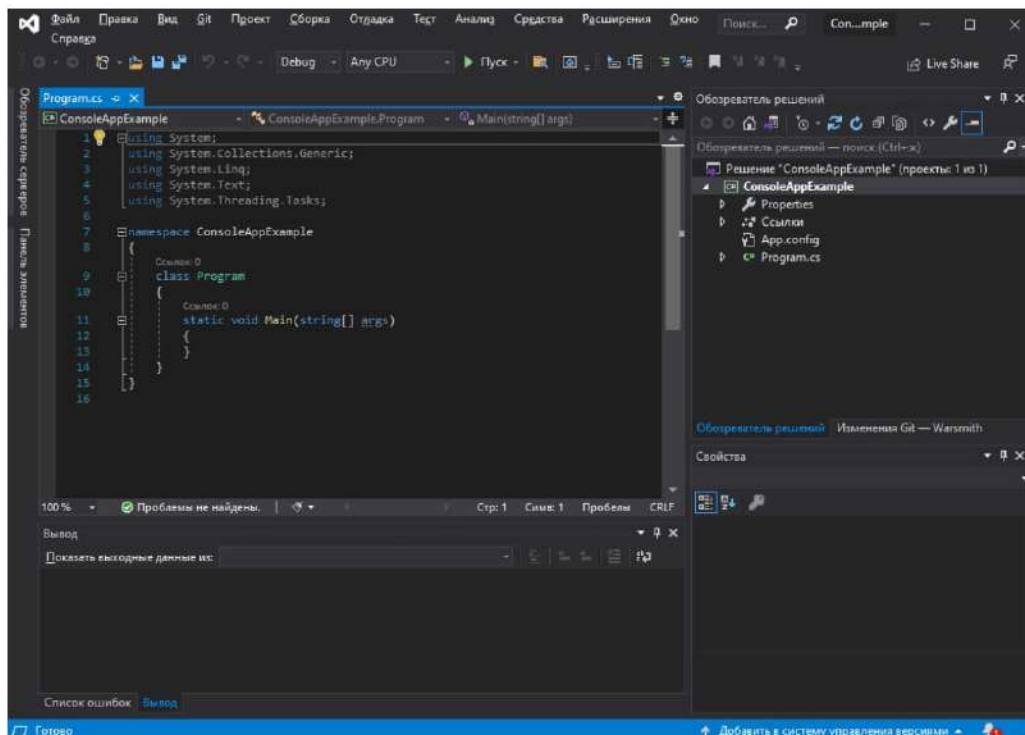
Для выполнения практических и расчетно-графической работ необходимо наличие установленного пакета Microsoft Visual Studio.

1. Создадим проект консольного приложения.

- Первый шаг при разработке любого приложения - создание проекта. Для этого запускаем Visual Studio -> Создание проекта -> Консольное приложение (.NET Framework)



- Вводим название проекта, расположение, имя решения и платформу (при необходимости) и нажимаем «Создать».
- После создания проекта должно появиться следующее окно, содержащее текст программы Program.cs на языке C#:



В данной программе объявление `using System` дает возможность ссылаться на классы, которые находятся в пространстве имен `System`, так что их можно использовать, не добавляя "System." перед именем типа.

Пространство имен `System` содержит много полезных классов. Одним из них является и класс `Console`, который используется при создании консольных приложений.

Класс проекта - `Program` размещен в пространстве имен, имеющем по умолчанию то же самое имя (`ConsoleAppExample`), что и решение, содержащее единственный проект.

Поскольку в C# нет глобальных функций, поэтому в данном примере объявляется класс `Program`, который содержит функцию `static Main()`, служащую начальной точкой выполнения программы.

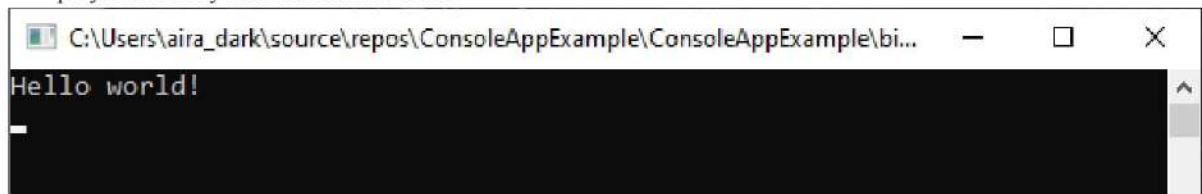
Функция `Main` может быть объявлена без параметров или с параметром, представляющий собой массив строк. Так как функция `Main` является точкой входа, она должна быть статической (`static`) функцией, т.е. она не связана с конкретным объектом класса, в котором объявлена. Тип данных `void` означает, что данная функция ничего не возвращает.

- Добавим внутри функции `Main` следующие строки:

```
Console.WriteLine("Hello world!");  
Console.ReadKey();
```

которая использует метод `WriteLine` класса `Console` для вывода строки "Hello world!".

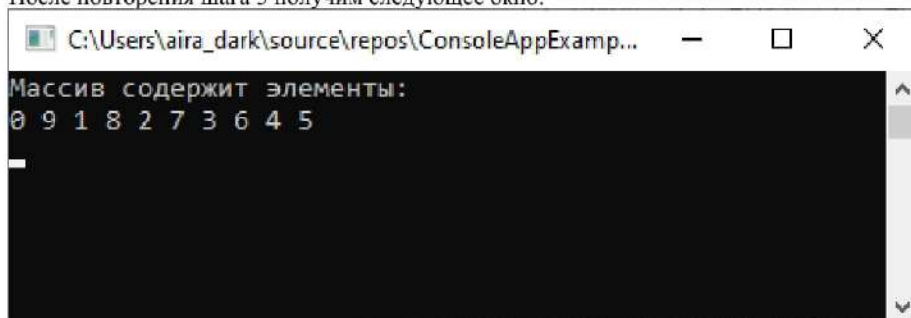
- Запуск приложения. Для можно использовать либо кнопку «Пуск», либо сочетание клавиш `<Ctrl+F5>`. В результате получаем консольное окно:



- Усложним приложение. Для этого добавим в функцию `Main` следующий код для вывода в консольное окно содержимого массива целых чисел:

```
static void Main(string[] args)  
{  
    int[] ArInt = { 0, 9, 1, 8, 2, 7, 3, 6, 4, 5 };  
    Console.WriteLine("Массив содержит элементы:");  
    foreach (int item in ArInt)  
    {  
        Console.Write("{0} ", item);  
    }  
    Console.WriteLine("");  
    Console.ReadKey();  
}
```

После повторения шага 3 получим следующее окно:



- Создадим второе консольное приложение, которое по заданным значениям коэффициентов a, b и c квадратного уравнения (значения вводятся с клавиатуры пользователем) вычисляет и отображает на экране корни уравнения. Уравнение имеет вид: $ax^2 + bx + c = 0$

Для данного приложения потребуются следующие методы:

- `string Console.ReadLine()` - чтение строки символов из входного потока.
- `double Convert.ToDouble(string)` - преобразование строки символов в число с плавающей запятой двойной точности.
- `double Math.Pow(double, double)` - возведение числа в степень.
- `double Math.Sqrt(double)` - извлечение квадратного корня числа.

1. Добавим внутри функции `Main` следующие строки:

```
string[] name_k = { "a", "b", "c"};
double[] value_k = new double[name_k.Length];

for (int i = 0; i < name_k.Length; i++)
{
    Console.WriteLine("Введите коэффициент {0}:", name_k[i]);
    value_k[i] = Convert.ToDouble(Console.ReadLine());
}

var d = Math.Pow(value_k[1], 2) - 4 * value_k[0] * value_k[2];
var x1 = (-value_k[1] + Math.Sqrt(d)) / (2 * value_k[0]);
var x2 = (-value_k[1] - Math.Sqrt(d)) / (2 * value_k[0]);

Console.WriteLine("Первый корень уранения равен {0}, второй равен {1}", x1, x2);
Console.ReadKey();
```

2. В результате получаем консольное окно:



```
C:\Users\aira_dark\source\repos\ConsoleAppExample\ConsoleAppExample\bin\Debug\ConsoleAppExample.exe
Введите коэффициент а:
1
Введите коэффициент b:
1
Введите коэффициент с:
-6
Первый корень уранения равен 2, второй равен -3
```

1. Работа с массивами.

Далее разработаем две программы, которые продемонстрируют, каким образом выполняется *инициализация* и работа с прямоугольными и ломаными массивами в C#.

1. Прямоугольный массив:

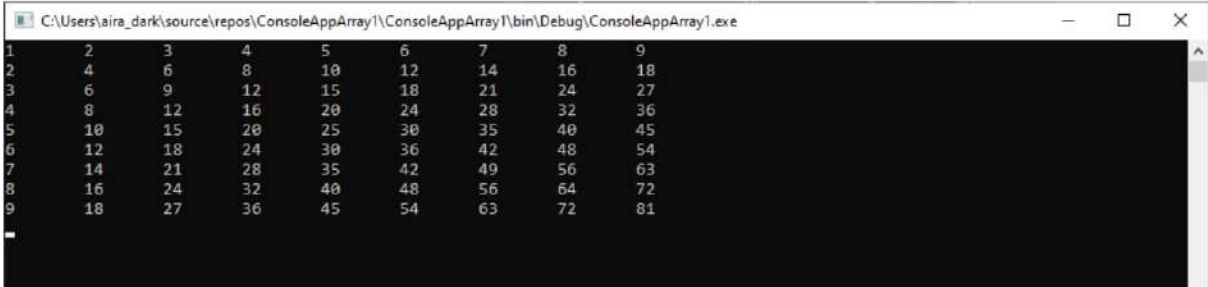
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleAppArray1
{
    class Program
    {
        static void Main(string[] args)
        {
            // Прямоугольный многомерный массив
            int[,] multMatr;
            multMatr = new int[10, 10];

            // Заполнение массива 9 на 9:
            for (int i = 1; i <= 9; i++)
                for (int j = 1; j <= 9; j++)
                    multMatr[i, j] = i * j;

            // Вывод элементов многомерного массива
            for (int i = 1; i <= 9; i++)
            {
                for (int j = 1; j <= 9; j++)
                {
                    Console.Write(multMatr[i, j] + "t");
                }
                Console.WriteLine();
            }
            Console.ReadKey();
        }
    }
}
```

В результат выполнения программы получим таблицу умножения следующего вида:



1	2	3	4	5	6	7	8	9
2	4	6	8	10	12	14	16	18
3	6	9	12	15	18	21	24	27
4	8	12	16	20	24	28	32	36
5	10	15	20	25	30	35	40	45
6	12	18	24	30	36	42	48	54
7	14	21	28	35	42	49	56	63
8	16	24	32	40	48	56	64	72
9	18	27	36	45	54	63	72	81

2. Ломаный массив.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```



```

using System.Threading.Tasks;

namespace ConsoleAppArray1
{
    class Program
    {
        static void Main(string[] args)
        {
            // Ломаный многомерный массив из пяти внутренних массивов разного размера
            int[][] JagArr = new int[10][];

            // Инициализация генератора случайных чисел
            Random rand = new Random();

            // Динамическое создание ломаного массив
            for (int i = 0; i < JagArr.Length; i++)
            {
                JagArr[i] = new int[i + rand.Next(10)];
            }

            // Вывод строк на консоль
            // Каждый элемент по умолчанию имеет значение, равное
            for (int i = 0; i < 10; i++)
            {
                // Свойство Length массива возвращает его размер
                Console.WriteLine("Length of row {0} is {1}:\t", i, JagArr[i].Length);
                for (int j = 0; j < JagArr[i].Length; j++)
                {
                    Console.Write(JagArr[i][j] + " ");
                }
                Console.WriteLine();
            }
            Console.ReadKey();
        }
    }
}

```

В результат выполнения программы получим таблицу следующего вида:

```

C:\Users\aira_dark\source\repos\ConsoleAppArray1\ConsoleAppArray1\bin\Debug\ConsoleAppArray1.exe
Length of row 0 is 5: 0 0 0 0 0
Length of row 1 is 7: 0 0 0 0 0 0 0
Length of row 2 is 7: 0 0 0 0 0 0 0
Length of row 3 is 10: 0 0 0 0 0 0 0 0 0 0
Length of row 4 is 9: 0 0 0 0 0 0 0 0 0
Length of row 5 is 11: 0 0 0 0 0 0 0 0 0 0 0
Length of row 6 is 8: 0 0 0 0 0 0 0 0
Length of row 7 is 14: 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Length of row 8 is 10: 0 0 0 0 0 0 0 0 0 0
Length of row 9 is 11: 0 0 0 0 0 0 0 0 0 0 0

```

2.Свойства и методы класса `Array`.

Следующий пример демонстрирует использование некоторых свойств и методов класса `Array`:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

```

```

namespace ConsoleAppArray1
{
    class Program
    {
        static void Main(string[] args)
        {
            // Массив символьных строк
            string[] Brands = new string[]
            {
                "Audi",
                "BMW",
                "Buick",
                "Chevrolet",
                "Citroen",
                "Dodge",
                "Ferrari",
                "Fiat",
                "Ford",
                "Honda",
                "Hyundai",
                "Cherokee",
                "Cherry",
                "Lada",
                "Lamborghini",
                "Lincoln",
                "Mazda",
                "Mercedes",
                "Mitsubishi",
                "Nissan",
                "Opel",
                "Peugeot",
                "Plymoth",
                "Pontiac",
                "Renault",
                "Rover",
                "Saab",
                "Subaru",
                "Suzuki",
                "Toyota",
                "Volkswagen",
                "Volvo"
            };

            // Вывод марок автомобилей в соответствии с порядком элементов в массиве

            Console.WriteLine("Here is the array of car brands:");

            for (int i = 0; i < Brands.Length; i++)
                Console.Write(Brands[i] + "\t");

            Console.WriteLine("\n\n");

            // Сортировка элементов в обратном порядке

            Array.Reverse(Brands);

            // Вывод автомарок
            Console.WriteLine("Here is the list once reversed:");
            for (int i = 0; i < Brands.Length; i++)
                Console.Write(Brands[i] + "\t");

            Console.WriteLine("\n\n");
        }
    }
}

```

```

        // Остаются только первый и последний

        Console.WriteLine("Only two remain: ");
        Array.Clear(Brands, 1, Brands.Length - 2);

        for (int i = 0; i < Brands.Length; i++)
        {
            if (Brands[i] != null)
                Console.Write(Brands[i] + "\t\n");
        }
        Console.ReadKey();
    }
}

```

Результат работы программы:

```

C:\Users\aira_dark\source\repos\ConsoleAppArray1\ConsoleAppArray1\bin\Debug\ConsoleAppArray1.exe
Here is the array of car brands:
Audi    BMW    Buick  Chevrolet  Citroen Dodge  Ferrari Fiat  Ford  Honda  Hyundai Cherokee  Cherry
Lada    Lamborghini Lincoln Mazda  Mercedes Mitsubishi Nissan Opel  Peugeot Plymouth Pontiac
Renault Rover  Saab    Subaru Suzuki Toyota Volkswagen Volvo

Here is the list once reversed:
Volvo Volkswagen Toyota Suzuki Subaru Saab Rover Renault Pontiac Plymouth Peugeot Opel Nissan Mitsubishi
hi Mercedes Mazda Lincoln Lamborghini Lada Cherry Cherokee Hyundai Honda Ford Fiat
Ferrari Dodge Citroen Chevrolet Buick BMW Audi

Only two remain:
Volvo
Audi

```

3. Использование класса StringBuilder:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleAppArray1
{
    class Program
    {
        static void Main(string[] args)
        {
            StringBuilder WordList = new StringBuilder("Строка ");

            WordList.Append("Для ");
            Console.WriteLine(WordList);
        }
    }
}

```

```

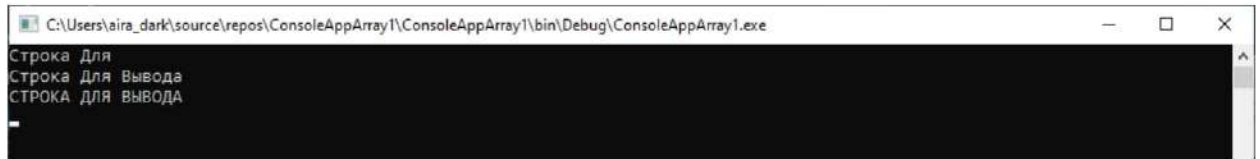
        WordList.Append("Вывода");
        Console.WriteLine(WordList);

        // Сделать все буквы прописными
        string Spectrum = WordList.ToString().ToUpper();
        Console.WriteLine(Spectrum);

        Console.ReadKey();
    }
}

```

Результат работы программы:



4. Использование класса `ArrayList`.

Использование класса `ArrayList` из пространства имен `System.Collections` позволяет эффективно реализовать работу с массивами объектов, поскольку многие возможности, необходимые для этого реализованы изначально, в частности методы вставки, удаления и нумерации элементов.

Для использования возможностей `ArrayList` используется не обычное наследование, а модель включения в виде делегирования вызовов на выполнение различных действий классу производному от `ArrayList`:

```

using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleAppArray1
{
    class NBooks : IEnumerable
    {
        // nbList — внутренний класс, который будет делать всю работу
        private ArrayList nbList;

        // Создаем объект класса nbList при помощи конструктора NBooks
        public NBooks() { nbList = new ArrayList(); }

        // Реализуем нужные нам методы для приема вызовов извне и передачи их nbList

        // Метод для вставки объекта NBook
        public void AddNBook(NBook nb)
        { nbList.Add(nb); }

        // Метод для удаления объекта NBook
        public void RemoveNBook(int nbToRemove)
        { nbList.RemoveAt(nbToRemove); }

        // Свойство, возвращающее количество объектов NBook
        public int NBookCount
        { get { return nbList.Count; } }
    }
}

```



```

        // Метод для очистки объекта — удаления всех объектов NBook
        public void ClearAllNBooks()
        { nbList.Clear(); }

        // Метод, который отвечает на вопрос — есть ли уже в наборе такой объект NBook
        public bool NBookIsPresent(NBook c)
        { return nbList.Contains(c); }

        // Все, что связано с реализацией IEnumerator, перенаправляется в nbList
        public IEnumerator GetEnumerator()
        { return nbList.GetEnumerator(); }
    }
}

```

Реализация класса `NBook` и код программы, использующей класс `NBooks` представлены ниже.

```

using System;
using System.Text;

namespace ConsoleAppArray1
{
    public class NBook
    {
        // Конструктор класса NBook

        public string Model;
        public string CPU_model;
        public int CPU_clock;
        public int RAM_size;

        public NBook(string mname, string CPU, int Clock, int RAM)
        {
            Model = mname;
            CPU_model = CPU;
            CPU_clock = Clock;
            RAM_size = RAM;
        }
    }

    class Program
    {
        static void Main(string[] args)
        {
            NBooks nbLot = new NBooks();

            // Создание списка объектов NBook

            nbLot.AddNBook(new NBook("ASUS A7Sn", "Intel Core 2 Duo T8300", 2400, 2048));
            nbLot.AddNBook(new NBook("Acer Aspire 5530G-803G25Mi", "AMD Turion X2 Ultra ZM80", 2100,
3072));
            nbLot.AddNBook(new NBook("Fujitsu Amilo Si 2636", "Intel Core 2 Duo T8300", 2400, 2048));
            nbLot.AddNBook(new NBook("HP Pavilion tx2650er", "AMD Turion X2 Ultra ZM82", 2200, 4096));

            // Выводим информацию о каждом объекте при помощи конструкции foreach

            Console.WriteLine("You have {0} in the lot: \n", nbLot.NBookCount);
            foreach (NBook nb in nbLot)
            {
                Console.WriteLine("Model: {0}", nb.Model);
            }
        }
    }
}

```

```

        Console.Write("CPU: {0}", nb.CPU_model);
        Console.WriteLine(" {0} GHz", nb.CPU_clock);
        Console.WriteLine("RAM: {0} GB\n", nb.RAM_size);
    }

    // Удаляем один из ноутбуков

    nbLot.RemoveNBook(3);
    Console.WriteLine("You have {0} in the lot.\n", nbLot.NBookCount);

    // Добавляем еще один ноутбук и проверяем его наличие в наборе

    NBook temp = new NBook("ASUS M51Ta", "AMD Turion™ X2 Ultra ZM84", 2300, 4096);
    nbLot.AddNBook(temp);

    if (nbLot.NBooksIsPresent(temp))
        Console.WriteLine(temp.Model + " is already in the lot.");

    // Удалить все

    nbLot.ClearAllNBooks();
    Console.WriteLine("You have {0} in the lot.\n", nbLot.NBookCount);

    Console.ReadKey();
}
}
}

```

Результат выполнения программы:

```

C:\Users\aira_dark\source\repos\ConsoleAppArray1\ConsoleAppArray1\bin\Debug\ConsoleAppArray1.exe
You have 4 in the lot:

Model: ASUS A75n
CPU: Intel Core 2 Duo T8300 2400 GHz
RAM: 2048 GB

Model: Acer Aspire 5530G-803G25Mi
CPU: AMD Turion X2 Ultra ZM80 2100 GHz
RAM: 3072 GB

Model: Fujitsu Amilo S1 2636
CPU: Intel Core 2 Duo T8300 2400 GHz
RAM: 2048 GB

Model: HP Pavilion tx2650er
CPU: AMD Turion X2 Ultra ZM82 2200 GHz
RAM: 4096 GB

You have 3 in the lot.

ASUS M51Ta is already in the lot.
You have 0 in the lot.

```

Контрольное задание

Необходимо разработать *консольное приложение* для ввода с клавиатуры массива строк и поиска среди них строк, содержащих заданный *строковый* фрагмент.

Для поиска потребуется использование метода `IndexOf(string findThisString)` для строковых элементов массива. Метод возвращает позицию начала искомой подстроки от начала строки, либо *значение -1* при отсутствии соответствия.