

Министерство образования Московской области

ГБПОУ МО «XXXX»

09.02.07

Дисциплина: Основы алгоритмизации и программирования

ЛАБОРАТОРНЫЕ РАБОТЫ

ОТЧЁТ

ККОО.ОАXXXX.000

Студент:

Преподаватель:

Дата:

Оценка:

2021

СОДЕРЖАНИЕ

Лабораторная работа №10	3
Лабораторная работа №11	14
Лабораторная работа №12	23
Лабораторная работа №13.1	30
Лабораторная работа №13.2	42
Лабораторная работа №13.3	86
Лабораторная работа №14	92
Лабораторная работа № 15	103

Лабораторная работа №10

Тема. Строки

Цель: изучить встроенную поддержку C# работы со строками, изучить множество встроенных методов для сравнения, поиска, сортировки и управления строковыми значениями

Задание 1. Для закрепления теоретического материала по теме Строки необходимо изучить теоретический материал и выполнить предложенные задания.

При решении задач использовать тип string.

Задание. Разработать программу, которая позволяет ввести строку с экрана и для введенной строки s:

- 1) подсчитывает общее число вхождений символов x и y;

```
static void Print(char[] arr)
{
    foreach (char elem in arr)
    {
        Console.Write(elem);
    }
    Console.WriteLine();
}

static void Main(string[] args)
{
    string s;
    char x, y;

    char[] arr;

    int count = 0;

    x = 'a';
    y = 'n';

    Console.WriteLine("Введите строку: ");
    s = Convert.ToString(Console.ReadLine());

    arr = s.ToCharArray();

    Console.WriteLine("Ваш массив символов: ");
    Print(arr);

    for (int i = 0; i < arr.Length; i++)
    {
        if (arr[i] == x || arr[i] == y)
        {

```

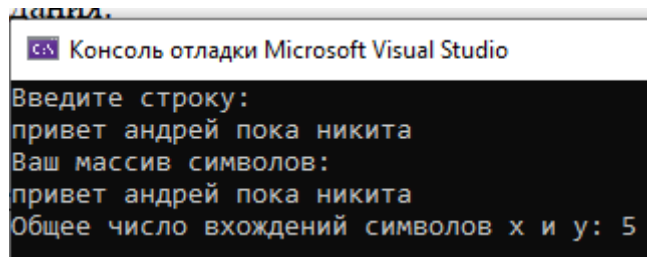
					KKOO.OAXXXX.000	Лист
						3
Изм.	Лист	№ докум.	Подпись	Дата		

```

        count++;
    }
}

Console.WriteLine($"Общее число вхождений символов x и y: {count}");
}

```



2) определяет, какой из двух заданных символов встречается в строке чаще всего;

```

static void Print(char[] arr)
{
    foreach (char elem in arr)
    {
        Console.Write(elem);
    }
    Console.WriteLine();
}

static void Main(string[] args)
{
    string s;
    char x, y;

    char[] arr;

    int countx = 0;
    int county = 0;

    x = 'a';
    y = 'п';

    Console.WriteLine("Введите строку: ");
    s = Convert.ToString(Console.ReadLine());

    arr = s.ToCharArray();

    Console.WriteLine("Ваш массив символов: ");
    Print(arr);

    for (int i = 0; i < arr.Length; i++)
    {
        if (arr[i] == x)
        {
            countx++;
        }

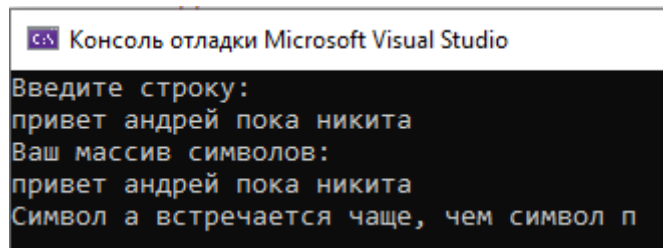
        if (arr[i] == y)
        {
            county++;
        }
    }
    if (countx > county)

```

```

{
    Console.WriteLine($"Символ {x} встречается чаще, чем символ {y}");
}
if (countx < county)
{
    Console.WriteLine($"Символ {y} встречается чаще, чем символ {x}");
}
else
{
    Console.WriteLine($"Символы встречаются одинаково.");
}
}
}

```



3) выводит на экран символы, которые наиболее часто встречается в строке;

```

class Program
{
    static void Main(string[] args)
    {
        string s;
        Console.Write("Введи строку: ");
        s = Console.ReadLine();
        char[] a = s.ToCharArray();
        int max = 0, chet;
        for (int i = 0; i < a.Length; i++)
        {
            chet = 0;
            for (int t = 0; t < a.Length; t++)
            {
                if (a[i] == a[t])
                {
                    chet++;
                }
            }
            if (chet > max)
            {
                max = chet;
            }
        }
        for (int i = 0; i < a.Length; i++)
        {
            chet = 0;
            for (int t = 0; t < a.Length; t++)
            {
                if (a[i] == a[t])
                {
                    chet++;
                }
            }
            if (chet == max)
            {

```

```

        Console.WriteLine($"Наиболее часто встречается символ - '{a[i]}'");
    }
}
Console.ReadKey();
}
}

```

4) выводит на экран символы, которые встречаются в строке только один раз;

```

class Program
{
    static void Main(string[] args)
    {
        string s;
        Console.Write("Введи строку: ");
        s = Console.ReadLine();
        char[] a = s.ToCharArray();
        int shet;
        for (int i = 0; i < a.Length; i++)
        {
            shet = 0;
            {
                for (int t = 0; t < a.Length; t++)
                {
                    if (a[i] == a[t])
                    {
                        shet++;
                    }
                }
                if (shet == 1)
                {
                    Console.WriteLine($"Символ '{a[i]}' встречается в строке только один раз");
                }
            }
        }
        Console.ReadKey();
    }
}

```

5) определяет, имеются ли в строке два соседствующих одинаковых символа;

```

class Program
{
    static void Main(string[] args)
    {
        string s;
        Console.Write("Введи строку: ");
        s = Console.ReadLine();
        char[] a = s.ToCharArray();
        for (int i = 0; i < a.Length; i++)
        {
            if (a[i] == a[i+1])
            {
                Console.WriteLine($"В строке есть два соседствующих одинаковых символа, это '{a[i]}' и '{a[i+1]}'");
            }
        }
        Console.ReadKey();
    }
}

```

6) определяет, является ли строка палиндромом;

7) определяет, упорядочены ли по алфавиту символы строки;

```
static void Print(char[] arr)
{
    foreach (char elem in arr)
    {
        Console.Write(elem);
    }
    Console.WriteLine();
}

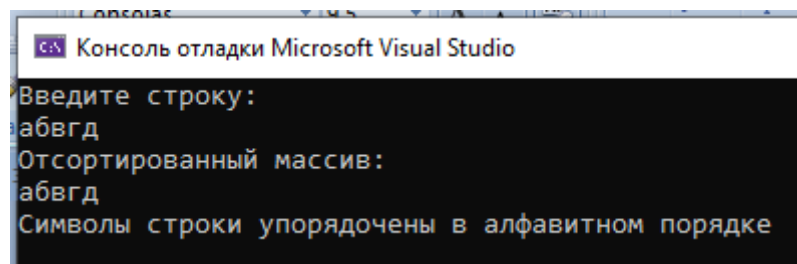
static void Main(string[] args)
{
    string s;
    bool tf = true;

    char[] arr_sort;
    char[] arr_nonesort;

    Console.WriteLine("Введите строку: ");
    s = Convert.ToString(Console.ReadLine());
    arr_nonesort = s.ToCharArray();
    arr_sort = s.ToCharArray();
    Array.Sort(arr_sort);
    Console.WriteLine("Отсортированный массив: ");
    Print(arr_sort);

    for (int i = 0; i < arr_sort.Length; i++)
    {
        if (arr_sort[i] == arr_nonesort[i])
        {
            tf = true;
        }
        else
        {
            tf = false;
        }
    }

    if (tf == true)
    {
        Console.WriteLine("Символы строки упорядочены в алфавитном порядке");
    }
    else
    {
        Console.WriteLine("Символы строки не упорядочены в алфавитном порядке");
    }
}
```



8) подсчитывает количество букв в строке;

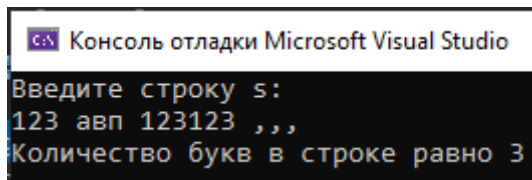
					ККОО.ОАXXXX.000	Лист
Изм.	Лист	№ докум.	Подпись	Дата		7

```

using System;

namespace ConsoleApp1
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Введите строку s:");
            string s = Console.ReadLine();
            char[] a;
            int sh1 = 0;
            a = s.ToCharArray();
            for (int i = 0; i < a.Length; i++)
            {
                if (char.IsLetter(a[i]))
                {
                    sh1++;
                }
            }
            Console.WriteLine($"Количество букв в строке равно {sh1}");
        }
    }
}

```



9) подсчитывает количество цифр в строке;

```

static void Print(char[] arr)
{
    foreach (char elem in arr)
    {
        Console.Write(elem);
    }
    Console.WriteLine();
}

static void Main(string[] args)
{
    string s;

    int count = 0;
    char[] arr;

    Console.WriteLine("Введите строку: ");
    s = Convert.ToString(Console.ReadLine());

    arr = s.ToCharArray();

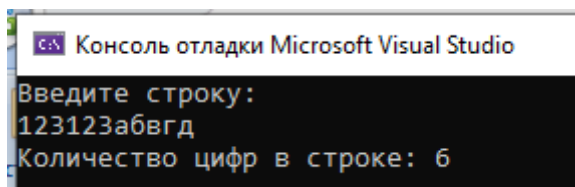
    for (int i = 0; i < arr.Length; i++)
    {
        if (char.IsDigit(arr[i]))
        {
            count++;
        }
    }

    Console.WriteLine($"Количество цифр в строке: {count}");
}

```

					ККОО.ОАХХХХ.000	Лист
Изм.	Лист	№ докум.	Подпись	Дата		8

}



10) подсчитывает сумму всех содержащихся в строке цифр;

```
static void Print(char[] arr)
{
    foreach (char elem in arr)
    {
        Console.Write(elem);
    }
    Console.WriteLine();
}

static void Main(string[] args)
{
    string s;
    int num = 0;

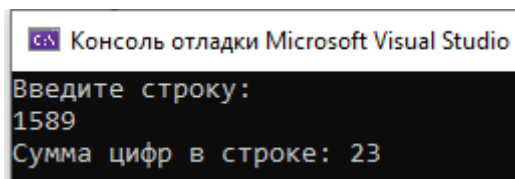
    char[] arr;

    Console.WriteLine("Введите строку: ");
    s = Convert.ToString(Console.ReadLine());

    arr = s.ToCharArray();

    for (int i = 0; i < arr.Length; i++)
    {
        if (char.IsDigit(arr[i]))
        {
            num += arr[i] - '0';
        }
    }

    Console.WriteLine($"Сумма цифр в строке: {num}");
}
```



11) выводит на экран последовательность символов, расположенных до первого двоеточия;

```
static void Print(char[] arr)
{
    foreach (char elem in arr)
    {
        Console.Write(elem);
    }
    Console.WriteLine();
}

static void Main(string[] args)
{

```

```

string s;
int num = 0;

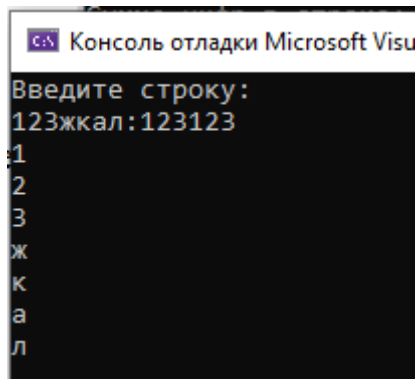
char[] arr;
char dvoethochie = ':';

Console.WriteLine("Введите строку: ");
s = Convert.ToString(Console.ReadLine());

arr = s.ToCharArray();

for (int i = 0; i < arr.Length; i++)
{
    if (dvoethochie == arr[i])
    {
        break;
    }
    else
    {
        Console.WriteLine(arr[i]);
    }
}
}

```



12) выводит на экран последовательность символов, расположенных после последнего двоеточия;

```

using System;

namespace ConsoleApp1
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.Write("Введите строку S: ");
            string s = Console.ReadLine();
            Console.Write("Строка после последнего двоеточия = ");
            int index = s.LastIndexOf(':');
            if (index >= 0)
            {
                Console.WriteLine(s.Substring(index+1));
            }
            else
            {
                Console.WriteLine(s);
            }
        }
    }
}

```

```
}
```

13) выводит на экран последовательность символов, расположенных между круглыми скобками (считается, что в строке ровно одна пара круглых скобок);

```
using System;

namespace ConsoleApp1
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.Write("Введите строку S: ");
            string s = Console.ReadLine();
            Console.Write("Между скобками = ");
            int index = s.IndexOf('(');
            int index2 = s.LastIndexOf('');
            if (index >= 0 && index2 >= 0)
            {
                Console.WriteLine(s.Substring(index+1,index2 - (index + 1)));
            }
            else
            {
                Console.WriteLine(s);
            }
        }
    }
}
```

14) находит самую длинную подстроку, состоящую только из цифр;

```
class Program
{
    static void Main(string[] args)
    {
        string s;
        Console.Write("Введи строку: ");
        s = Console.ReadLine();
        bool a;
        int chet = 0, chet2 = 0, ind = 0;
        Char[] sim = s.ToCharArray();
        for (int i = 0; i < s.Length; i++)
        {
            a = char.IsDigit(sim[i]);
            if (a == true)
            {
                chet++;
            }
            else
            {
                if (chet2 < chet)
                {
                    chet2 = chet;
                }
                chet = 0;
            }
        }
    }
}
```

					KKOO.OAXXXX.000	Лист
Изм.	Лист	№ докум.	Подпись	Дата		11

```

    }
    chet = 0;
    for (int i = 0; i < s.Length; i++)
    {
        a = char.IsDigit(sim[i]);
        if (a == true)
        {
            chet++;
        }
        else
        {
            chet = 0;
        }
        if (chet == chet2)
        {
            ind = i - chet + 1;
        }
    }
    Console.WriteLine($"{s.Substring(ind, chet2)}");
    Console.ReadKey();
}
}

```

15) находит самую длинную подстроку, состоящую из повторяющегося СИМВОЛА.

```

class Program
{
    static void Main(string[] args)
    {
        string s;
        Console.Write("Введи строку: ");
        s = Console.ReadLine();
        int chet = 0, chet2 = 0, ind = 0;
        Char[] sim = s.ToCharArray();
        for (int i = 1; i < s.Length; i++)
        {
            if (sim[i] == sim[i-1])
            {
                chet++;
            }
            else
            {
                if (chet2 < chet)
                {
                    chet2 = chet;
                }
                chet = 0;
            }
        }
        chet = 0;
        for (int i = 1; i < s.Length; i++)
        {
            if (sim[i] == sim[i-1])
            {
                chet++;
            }
            else
            {
                chet = 0;
            }
        }
    }
}

```

					ККОО.ОАXXXX.000	Лист
Изм.	Лист	№ докум.	Подпись	Дата		12

```

    }
    if (chet == chet2)
    {
        ind = i - chet + 1;
    }
    }
    Console.WriteLine($"{s.Substring(ind - 1, chet2 + 1)}");
    Console.ReadKey();
}
}

```

					ККОО.ОАXXXX.000	Лист
						13
Изм.	Лист	№ докум.	Подпись	Дата		

Лабораторная работа №11

Тема. Строки

Цель: изучить встроенную поддержку C# работы со строками, изучить множество встроенных методов для сравнения, поиска, сортировки и управления строковыми значениями

Задание 2. Для закрепления теоретического материала по теме Строки необходимо изучить теоретический материал и выполнить предложенные задания.

При решении задач использовать класс StringBuilder

Задание. Разработать программу, которая позволяет ввести строку с экрана и выполняет следующие действия с введенной строкой:

- 1) вставляет в строку символ x после каждого вхождения символа y;

```
static void Main(string[] args)
{
    Console.WriteLine("Введите любую строку: ");
    StringBuilder user_str = new StringBuilder(Console.ReadLine());

    Console.WriteLine("Введите символ, который хотите вставлять после каждого символа вашей строки: ");
    string x = Console.ReadLine();

    Console.WriteLine("Введите символ, после которого надо вставить предыдущий символ: ");
    char y = Convert.ToChar(Console.ReadLine());

    for (int i = 0; i < user_str.Length; i++)
    {
        if (user_str[i] == y)
        {
            user_str.Insert(i + 1, x);
        }
    }

    Console.WriteLine($"Измененная строка: {user_str}");
}
```

```
cs Консоль отладки Microsoft Visual Studio
Введите любую строку:
hello androllo
Введите символ, который хотите вставлять после каждого символа вашей строки:
!
Введите символ, после которого надо вставить предыдущий символ:
1
Измененная строка: hel!l!o androl!l!o
```

2) вставляет в строку подстроку x после каждого вхождения подстроки y;

```
static void Main(string[] args)
{
    Console.WriteLine("Введите любую строку: ");
    StringBuilder user_str = new StringBuilder(Console.ReadLine());

    Console.WriteLine("Введите подстроку: ");
    string x = Console.ReadLine();

    Console.WriteLine("Введите другую подстроку: ");
    string y = Console.ReadLine();

    user_str.Replace(x, x + y);

    Console.WriteLine($"Измененная строка: {user_str}");
}
```

```
cs Консоль отладки Microsoft Visual Studio
Введите любую строку:
привет андрей
Введите подстроку:
привет
Введите другую подстроку:
мой друг
Измененная строка: привет мой друг андрей
```

3) удваивает каждое вхождение заданного символа x;

```
static void Main(string[] args)
{
    Console.WriteLine("Введите любую строку: ");
    StringBuilder user_str = new StringBuilder(Console.ReadLine());

    Console.WriteLine("Введите символ, который встречается в вашей строке: ");
    string x = Console.ReadLine();

    user_str.Replace(x, x + x);

    Console.WriteLine($"Измененная строка: {user_str}");
}
```

```
cs Консоль отладки Microsoft Visual Studio
Введите любую строку:
привет андрей
Введите символ, который встречается в вашей строке:
и
Измененная строка: приивет андрей
```

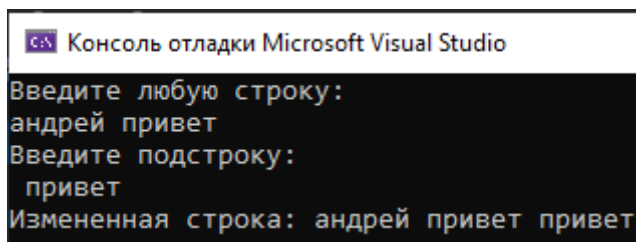
4) удваивает каждое вхождение заданной подстроки x;

```
static void Main(string[] args)
{
    Console.WriteLine("Введите любую строку: ");
    StringBuilder user_str = new StringBuilder(Console.ReadLine());

    Console.WriteLine("Введите подстроку: ");
    string x = Console.ReadLine();

    user_str.Replace(x, x + x);

    Console.WriteLine($"Измененная строка: {user_str}");
}
```



5) удаляет среднюю букву, если длина строки нечетная, и две средних, если длина строки четная;

```
static void Main(string[] args)
{
    Console.WriteLine("Введите любую строку: ");
    StringBuilder user_str = new StringBuilder(Console.ReadLine());
    StringBuilder edit_str = new StringBuilder();
    string tmp_user_str = Convert.ToString(user_str);

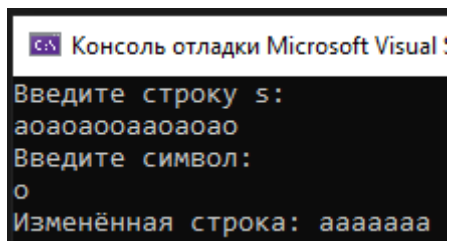
    string[] txtMass;
    txtMass = tmp_user_str.Split(' ');

    for (int i = 0; i <= txtMass.Length-1; i++)
    {
        char[] txtMassChar = txtMass[i].ToCharArray();
        Console.WriteLine(txtMassChar);
        for (int j = 0; j < txtMassChar.Length; j++)
        {
            int tmp_bool = txtMassChar.Length % 2;
            int middle = txtMassChar.Length / 2;
            if (tmp_bool == 0)
            {
                txtMass[i].ToString();
                txtMass[i].Remove(txtMassChar[middle]);
            }
            else
            {
                txtMass[i].ToString();
                txtMass[i].Remove(txtMassChar[middle]);
            }
        }
        edit_str.Append(txtMass[i]);
    }
    Console.WriteLine($"Измененная строка: {edit_str}");
}
```


6) удаляет все символы x;

```
using System;
using System.Text;

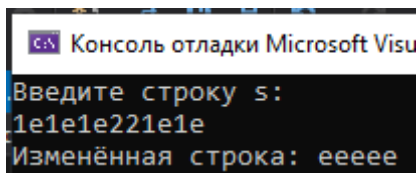
namespace ConsoleApp4
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Введите строку s:");
            StringBuilder s = new StringBuilder(Console.ReadLine());
            Console.WriteLine("Введите символ:");
            string x = Console.ReadLine();
            string y = string.Empty;
            s.Replace(x, y);
            Console.WriteLine($"Изменённая строка: {s}");
        }
    }
}
```



7) удаляет из строки все цифры;

```
using System;
using System.Text;

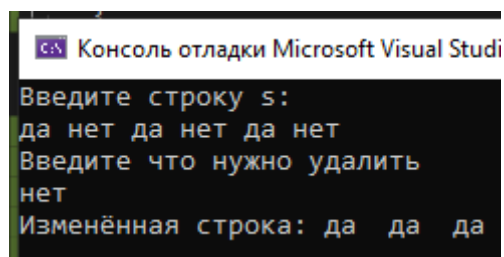
namespace ConsoleApp4
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Введите строку s:");
            StringBuilder s = new StringBuilder(Console.ReadLine());
            string y = "";
            for (int i = 0; i < s.Length; i++)
            {
                if (!char.IsDigit(s[i]))
                {
                    y += s[i];
                }
            }
            Console.WriteLine($"Изменённая строка: {y}");
        }
    }
}
```



8) удаляет все подстроки substr;

```
using System;
using System.Text;

namespace ConsoleApp4
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Введите строку s:");
            StringBuilder s = new StringBuilder(Console.ReadLine());
            Console.WriteLine("Введите что нужно удалить");
            string y = Console.ReadLine();
            s.Replace(y, null);
            Console.WriteLine($"Изменённая строка: {s}");
        }
    }
}
```



9) заменяет все вхождения подстроки str1 на подстроку str2 (при этом str1 может являться частью str2);

```
using System;
using System.Text;

namespace ConsoleApp4
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Введите строку s:");
            StringBuilder s = new StringBuilder(Console.ReadLine());
            Console.WriteLine("Введите что нужно удалить");
            string y = Console.ReadLine();
            Console.WriteLine("Введите что нужно вставить");
            string x = Console.ReadLine();
            s.Replace(y, x);
            Console.WriteLine($"Изменённая строка: {s}");
        }
    }
}
```

```

Консоль отладки Microsoft Visual Studio
Введите строку s:
да нет да нет да нет данет
Введите что нужно удалить
нет
Введите что нужно вставить
но
Изменённая строка: да но да но да но дано

```

10) заменяет все группы стоящих рядом точек на многоточие;

```

using System;
using System.Text;

namespace ConsoleApp4
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Введите строку s:");
            StringBuilder s = new StringBuilder(Console.ReadLine());
            string y = "...";
            string m = "...";
            s.Replace(y, m);
            Console.WriteLine($"Изменённая строка: {s}");
        }
    }
}

```

```

Консоль отладки Microsoft Visual Studio
Введите строку s:
1231..svsv.svsvsv.1.
Изменённая строка: 1231...svsv.svsvsv.1.

```

11) меняет местами первую букву со второй, третью с четвертой и
т.д.;

```

using System;
using System.Text;

namespace ConsoleApp4
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Введите строку s:");
            StringBuilder s = new StringBuilder(Console.ReadLine());
            for (int i = 0; i < s.Length - 1; i += 2)
            {
                char k = s[i + 1];
                s[i + 1] = s[i];
                s[i] = k;
            }
            Console.WriteLine($"Изменённая строка: {s}");
        }
    }
}

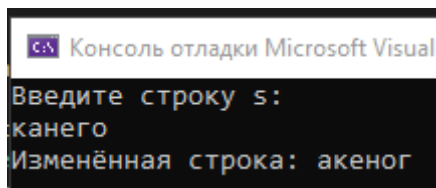
```

					ККОО.ОАXXXX.000	Лист
						19
Изм.	Лист	№ докум.	Подпись	Дата		

```

    }
}
}

```



12) меняет местами первую букву с последней, вторую с предпоследней и т.д.;

```

using System.Text;
class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("Введите строку:");
        StringBuilder s = new StringBuilder(Console.ReadLine());
        int i2 = s.Length-1;
        for (int i1 = 0; i1 < s.Length/2; i1++)
        {
            char s1 = s[i1];
            char s2 = s[i2];
            s.Remove(i1, 1);
            s.Insert(i1, s2);
            s.Remove(i2, 1);
            s.Insert(i2, s1);
            i2--;
        }
        Console.WriteLine($"Измененная строка: {s}");
        Console.ReadKey();
    }
}

```

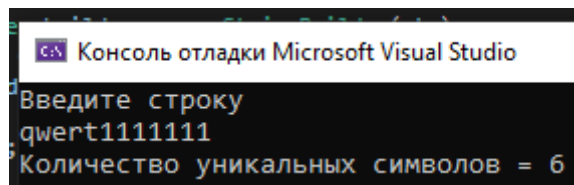
13) определяет, сколько различных символов встречается в строке;

```

using System;
using System.Text;

namespace ConsoleApp4
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Введите строку");
            string str = Console.ReadLine();
            StringBuilder builder = new StringBuilder(str);
            int count = 0;
            while (builder.Length > 0)
            {
                count++;
                builder.Replace(new string(builder[0], 1), "");
            }
            Console.WriteLine($"Количество уникальных символов = {count}");
        }
    }
}

```



14) удаляет из строки все подстроки, состоящие из цифр;

```
class Program
{
    static void Main(string[] args)
    {
        Console.Write("Введите строку:");
        StringBuilder s = new StringBuilder(Console.ReadLine());
        int k = 1, i = 0;
        bool f;
        for (int i2 = 0; i2 < s.Length; i2++)
        {
            k = 1;
            f = char.IsDigit(s[i2]);
            if (f == true)
            {
                i = i2;
                while (f == true)
                {
                    i2++;
                    f = char.IsDigit(s[i2]);
                    if (f == true)
                    {
                        k++;
                    }
                }
                s.Remove(i, k);
            }
            Console.WriteLine($"Измененная строка: {s}");
            Console.ReadKey();
        }
    }
}
```

15) удаляет из строки самую длинную подстроку, состоящую из повторяющегося символа.

```
using System.Text;

class Program
{
    static void Main(string[] args)
    {
        Console.Write("Введите строку:");
        StringBuilder s = new StringBuilder(Console.ReadLine());
        int chet = 0, chet2 = 0;
        for (int i = 0; i < s.Length; i++)
        {
            chet2 = 0;
            int i2 = i + 1;
            if (i2 != s.Length)
```

```

    {
        while (s[i] == s[i2])
        {
            chet2++;
            i2++;
        }
    }
    if (chet2 > chet)
    {
        chet = chet2;
    }
}
for (int i = 0; i < s.Length; i++)
{
    chet2 = 0;
    int i2 = i + 1;
    if (i2 != s.Length)
    {
        while (s[i] == s[i2])
        {
            chet2++;
            i2++;
        }
    }
    if (chet2 == chet)
    {
        s.Remove(i, chet + 1);
    }
}
Console.WriteLine($"Измененная строка: {s}");
Console.ReadKey();
}
}

```

Лабораторная работа №12

Тема. Вещественный тип данных

Цель: изучить алгоритмы решения уравнения $f(x)=0$; вычисления площади фигуры.

Задание 1. Для закрепления теоретического материала по теме Вещественный тип данных необходимо изучить теоретический материал и выполнить предложенные задания.

Для самостоятельной работы исследуйте функции с помощью созданной программы

$$\begin{aligned}x^2 - 6x + 5 &= 0 \\ x - \cos(x) &= 0 \\ x - \ln(x) - 2 &= 0 \\ 2x^3 - 9x^2 - 60x + 1 &= 0\end{aligned}$$

```
using System;

namespace ConsoleApp5
{
    class Program
    {
        public static double eps;

        static bool VichToch(double x, double y)
        {
            return Math.Abs(x - y) < eps;
        }

        static double F(double x)
        {
            return Math.Pow(x, 2) - 2;
        }

        static void Main(string[] args)
        {
            Console.WriteLine("Введите значение точности");
            eps = double.Parse(Console.ReadLine());
            Console.WriteLine("Введите нижнюю границу интервала");
            double a = double.Parse(Console.ReadLine());
            Console.WriteLine("Введите верхнюю границу интервала");
            double b = double.Parse(Console.ReadLine());
            if (F(a)*F(b) > 0)
            {
                Console.WriteLine("На этом интервале нет корней");
            }
            else
            {
                while (!VichToch(a, b))
                {
                    a = (a + b) / 2;
                    b = (a + b) / 2;
                }
            }
        }
    }
}
```

```

    {
        double c = (a + b) / 2;
        if (F(c) > 0)
        {
            b = c;
        }
        else
        {
            a = c;
        }
    }
    Console.WriteLine($"Корень равен {a}");
}
}
}

static double F(double x)
{
    return Math.Pow(x, 2) - 6 * x + 5;
}

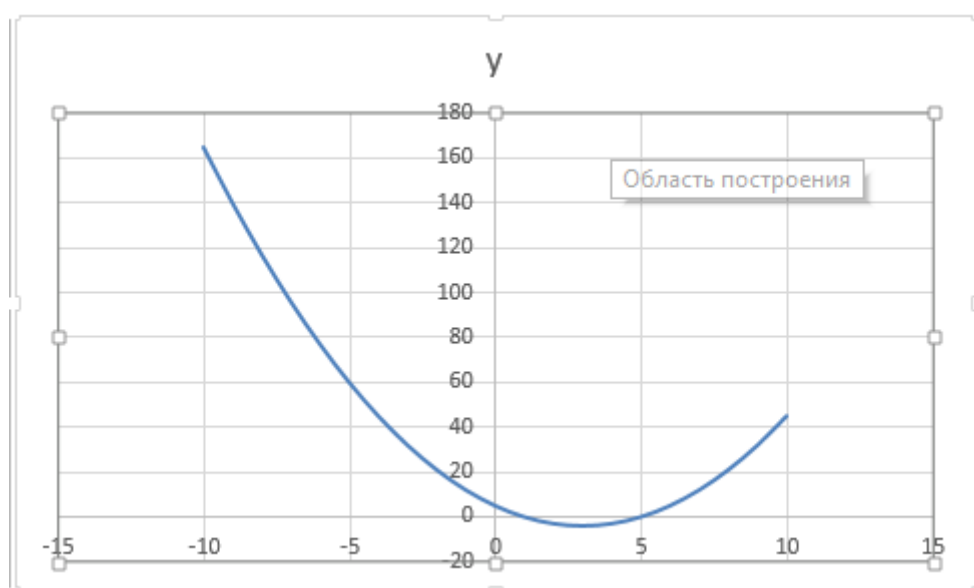
```

Консоль отладки Microsoft Visual Studio

```

Введите значение точности
0,001
Введите нижнюю границу интервала
1
Введите верхнюю границу интервала
7
Корень равен 4,999755859375

```



```

static double F(double x)
{
    return x - Math.Cos(x);
}

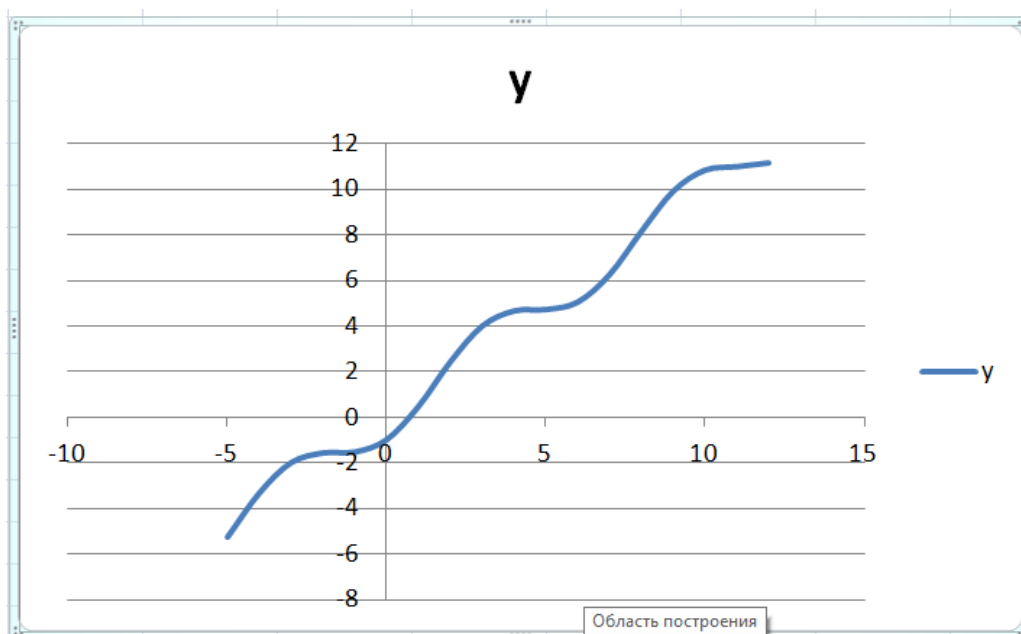
```


Консоль отладки Microsoft Visual Studio

```

Введите значение точности
0,001
Введите нижнюю границу интервала
0
Введите верхнюю границу интервала
4
Корень равен 0,73828125

```



```

static double F(double x)
{
    return x - Math.Log(x) - 2;
}

```

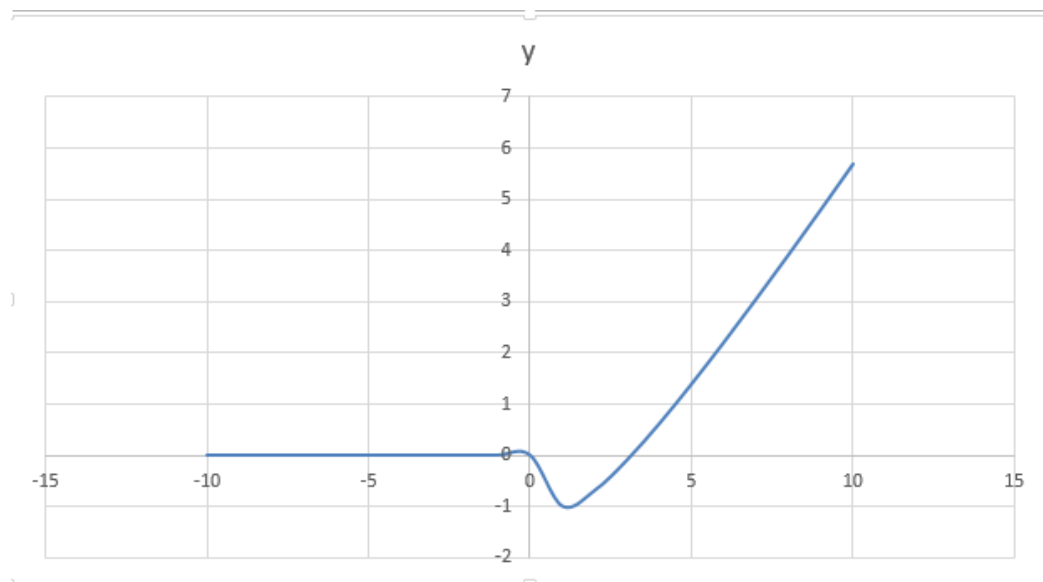
Консоль отладки Microsoft Visual Studio

Назад (Ctrl+-)

```

Введите значение точности
0,001
Введите нижнюю границу интервала
1
Введите верхнюю границу интервала
7
Корень равен 3,14599609375

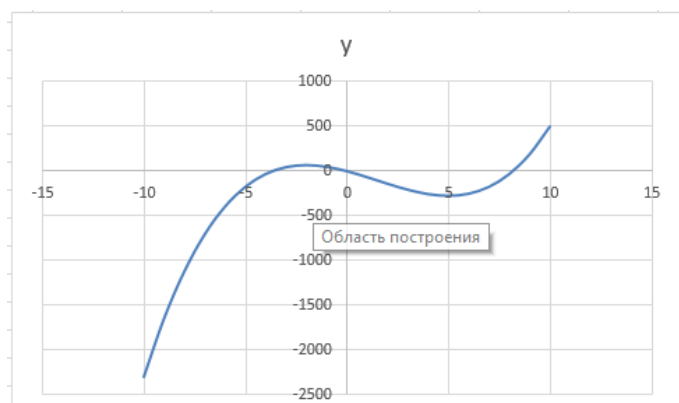
```



```
static double F(double x)
{
    return 2*Math.Pow(x, 3) - 9*Math.Pow(x, 2) - 60*x + 1;
}
```

```
Консоль отладки Microsoft Visual Studio
Введите значение точности
0,001
Введите нижнюю границу интервала
-5
Введите верхнюю границу интервала
-1
Корень равен -3,68359375
```

```
Консоль отладки Microsoft Visual Studio
Введите значение точности
0,001
Введите нижнюю границу интервала
5
Введите верхнюю границу интервала
10
Корень равен 8,1658935546875
```



1. Пусть $f(x)$ — непрерывная положительная функция на отрезке $[a, b]$ ($a < b$). Вычислим площадь фигуры, ограниченную графиком функции

$f(x)$, осью x и прямыми $x=a$ и $x=b$. Для этого разобьём отрезок $[a,b]$ на n равных отрезков одинаковой длины $\Delta x = (b - a)/n$. На каждом из отрезков $[x(i), x(i+1)]$ как на основании, построим прямоугольник с высотой $f(x(i))$. Площадь прямоугольников мы умеем считать. Сумма площадей всех прямоугольников даст приближенное значение площади фигуры. Общая формула:

$$S_{\text{прибл}} = (b - a)/n * (f(x(0)) + f(x(1)) + \dots + f(x(n - 1)))$$

Естественно, что чем «мельче» будет разбиение, тем точнее мы подсчитаем площадь фигуры.

Будем сравнивать два соседних приближенных значения площади для различных n . Если их разность больше заданной точности, то увеличиваем значение n и снова подсчитаем площадь. Начальное значение n взято 10, шаг изменения 100.

```
class Program
{
    const double eps=0.001;
    const int nb=10;
    const int ns = 100;
    static double a, b, ws, wn,n;

    static bool VichToch(double x, double y)
    {
        return Math.Abs(x - y) < eps;
    }

    static double F(double x)
    {
        return 1/(1+x);
    }
    static double Sq(double n)
    {
        double x,dx,s;
        x = a;
        dx = (b - a) / n;
        s = 0;
        for (int i = 1; i < n - 1; i++)
        {
            s += F(x);
            x += dx;
        }
        return (b - a) / n * s;
    }

    static void Main(string[] args)
    {
        Console.WriteLine("Введите нижнюю границу интервала");
```

					ККОО.ОАXXXX.000	Лист
Изм.	Лист	№ докум.	Подпись	Дата		27

```

a = double.Parse(Console.ReadLine());
Console.WriteLine("Введите верхнюю границу интервала");
b = double.Parse(Console.ReadLine());
n = nb;
wn = Sq(n);
do
{
    ws = wn;
    n += ns;
    wn = Sq(n);
} while (VichToch(ws,wn));
Console.WriteLine($"Количество частей {n} Значение площади {wn}");
Console.ReadKey();
}
}

```

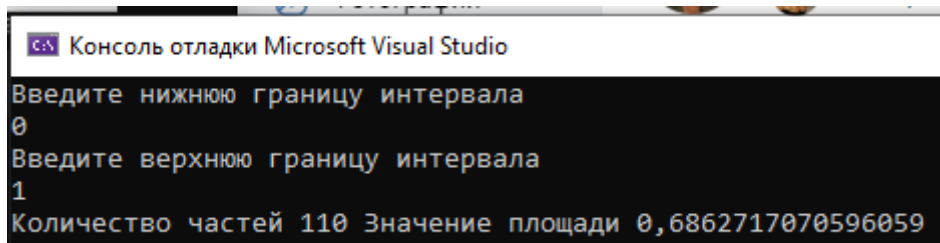
После запуска программы результат будет не тот. Найдите ошибки!

После запуска получаем, что при $n=810$ достигается требуемая точность вычислений, результат — 2.32654955.... Однако в уме мы подсчитали точнее. Изменим точность на $1.0E-6$. Ваш компьютер задумается на некоторое время, а результат $n=23310$ и площадь — 2.33309739... Для самоконтроля Вам предлагается вычислить площадь для следующих функций: $f(x)=1/(1+x)$, $[0,1]$; $f(x)=1/x$, $[1,3]$; $f(x)=\sin(x)$, $[0, \pi/2]$.

```

static double F(double x)
{
    return 1 / (1+x);
}

```



```

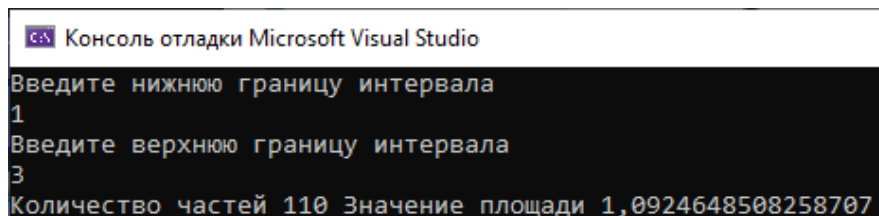
cs Консоль отладки Microsoft Visual Studio
Введите нижнюю границу интервала
0
Введите верхнюю границу интервала
1
Количество частей 110 Значение площади 0,6862717070596059

```

```

static double F(double x)
{
    return 1 / x;
}

```



```

cs Консоль отладки Microsoft Visual Studio
Введите нижнюю границу интервала
1
Введите верхнюю границу интервала
3
Количество частей 110 Значение площади 1,0924648508258707

```

```

static double F(double x)
{

```

					KKOO.OAXXXX.000	Лист
Изм.	Лист	№ докум.	Подпись	Дата		28

```
        return Math.Sin(x);  
    }  
    b = Math.PI / 2;
```

					ККОО.ОАXXXX.000	Лист
						29
Изм.	Лист	№ докум.	Подпись	Дата		

Лабораторная работа №13.1

Тема Реализация алгоритмов

Задание 1.

Написать программу, вычисляющую первые n элементов заданной последовательности:

$$11. b_1 = 5, b_n = \frac{b_{n-1}}{n^2 + n + 1}$$

```
class Program
{
    static double b = 5;
    static double bn = 0;

    static double F(int i)
    {
        bn = b / (Math.Pow(i, 2) + i + 1);
        b = bn;
        return bn;
    }
    static void Main(string[] args)
    {
        Console.WriteLine("Введите, сколько нужно вычислить n-элементов: ");
        int n = Convert.ToInt32(Console.ReadLine());

        for (int i = 2; i <= n; ++i)
        {
            bn = F(i);
            Console.WriteLine($"b{i} = {bn}");
        }
    }
}
```

```

Введите, сколько нужно вычислить n-элементов:
25
b2 = 0.7142857142857143
b3 = 0.054945054945054944
b4 = 0.0026164311878597592
b5 = 8.440100605999224E-05
b6 = 1.9628140944184243E-06
b7 = 3.443533498979692E-08
b8 = 4.71716917668451E-10
b9 = 5.183702391960999E-12
b10 = 4.670002154919819E-14
b11 = 3.5112798157291875E-16
b12 = 2.2364839590631766E-18
b13 = 1.2221223820017358E-20
b14 = 5.792049203799696E-23
b15 = 2.4033399185890854E-25
b16 = 8.803442925234745E-28
b17 = 2.8675709854184836E-30
b18 = 8.360265263610739E-33
b19 = 2.1942953447797214E-35
b20 = 5.212102956721428E-38
b21 = 1.125724180717371E-40
b22 = 2.2203632755766688E-43
b23 = 4.0151234639722766E-46
b24 = 6.680737876825752E-49
b25 = 1.0262270164094857E-51

```

Задание 2.

Для заданного натурального n и действительного x подсчитать следующие суммы:

$$11. S = 1! - 2! + 3! - \dots + (-1)^{n+1} n!$$

```

class Program
{
    static int factorial = 1;
    static int summ = 0;

    static int Factorial(int i)
    {
        factorial *= i;
        return factorial;
    }

    static int Summ(int factorial)
    {
        summ += factorial;
        return summ;
    }

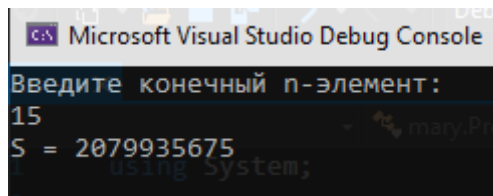
    static void Main(string[] args)
    {
        Console.WriteLine("Введите конечный n-элемент: ");
        int n = Convert.ToInt32(Console.ReadLine());
    }
}

```

```

for (int i = 1; i <= n; ++i)
{
    factorial = Factorial(i);
    factorial *= -1;
    summ = Summ(factorial);
}
Console.WriteLine($"S = {summ}");
}
}

```



Задание 3.

Для заданного натурального k и действительного x подсчитать следующее выражение:

$$11. P = \prod_{n=1}^k \left(1 + \frac{x^n}{n^2}\right)$$

```

class Program
{
    static double f = 1;
    static double p = 1;

    static double F(int i, double x)
    {
        f = Math.Pow(x, i) / i;
        return f;
    }

    static double P(double f)
    {
        p = 1 + f;
        return p;
    }

    static void Main(string[] args)
    {
        Console.WriteLine("Введите значение n: ");
        int n = Convert.ToInt32(Console.ReadLine());

        Console.WriteLine("Введите значение x: ");
        double x = Convert.ToDouble(Console.ReadLine());

        for (int i = 1; i <= n; ++i)
        {
            f = F(i, x);
            p *= P(f);
        }

        Console.WriteLine($"P={p}");
    }
}

```


}

```

Microsoft Visual Studio Debug Console
Введите значение n:
10
Введите значение x:
3
static double f = 1;
P=9.446711346845703E+19
  
```

Задание 4.

Вычислить бесконечную сумму ряда с заданной точностью ϵ ($\epsilon > 0$).

$$11. S = \sum_{i=1}^{\infty} \frac{(-1)^{2i}}{i(i+1)(i+2)}$$

Задание 5.

Вычислить и вывести на экран значение функции $F(x)$ на отрезке $[a, b]$ с шагом $h = 0.1$ и точностью E . Результат работы программы представить в виде следующей таблицы:

№	Значение x	Значение функции F(x)	Количество просуммированных слагаемых n
1			
2			
...			

$$11. F(x) = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots, x \in [0; 1]$$

```

class Program
{
    static double F = 1;
    static double a = -1;
    static double h = 0.1;
    static int count = 0;
    static int factorial = 1;

    static int Factorial(int i)
    {
        factorial *= i;
        return factorial;
    }

    static double PowX(double i, double h)
    {
        h = Math.Pow(h, i);
        return h;
    }

    static double Function(int factorial, double h)
    {
        F = h / factorial;
        return F;
    }

    static void Main(string[] args)
    {
  
```

```

string sx = "Значение x";
string sfx = "Значение F(x)";
string scout = "Количество слагаемых n";

Console.WriteLine("Введите точность e (прим. 0.001): ");
double e = Convert.ToDouble(Console.ReadLine());

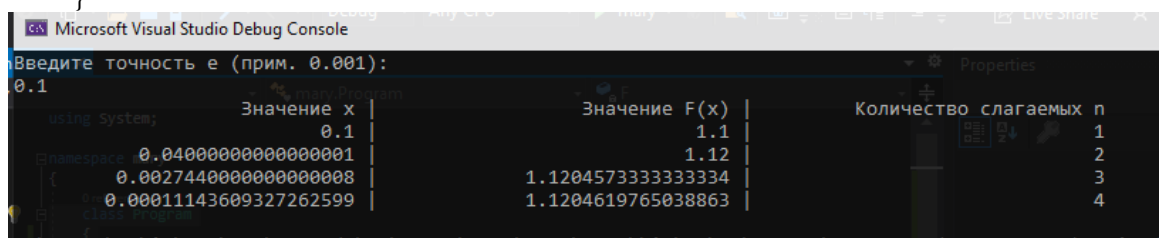
```

```

Console.WriteLine($"{sx, 30} | {sfx,30} | {scout,30}");
for (int i = 1; Math.Abs(a) >= e; ++i)
{
    factorial = Factorial(i);
    h = PowX(i, h);
    F += Function(factorial, h);
    count += 1;
    a /= -i;
    Console.WriteLine($"{h,30} | {F,30} | {count,30}");

    h += 0.1;
}
}
}

```



Задание 6

Для заданного натурального числа N. Определить, является ли заданное число простым; если нет, то вывести на экран все его делители;

```

class Program
{
    static bool IsPrime(double number)
    {
        for (int i = 2; i < number; i++)
        {
            if (number % i == 0)
                return false;
        }
        return true;
    }
    static void Main(string[] args)
    {
        double result = 0;
        Console.WriteLine("Введите значение N и нажмите Enter:");
        Console.Write("N = ");
        result = Convert.ToDouble(Console.ReadLine());
        if (IsPrime(result))
        {
            Console.WriteLine($"Число {result} является простым.");
        }
        else
        {
            for (int j = 1; j <= result; j++)
            {

```

```

        if (result % j == 0)
        {
            Console.WriteLine($"Найден делитель {j} числа {result}");
        }
        else
        {
            bool tmp = false;
        }
    }
}
}
}

```

Задание 7

Даны два натуральных числа а и b. Сократить дробь вида a/b

```

class Program
{
    static double a;
    static double b;
    static double del;
    static double ReduceFraction(double a, double b)
    {
        while (b != 0)
            b = a % (a = b);
        return a;
    }
    static void Main(string[] args)
    {
        Console.WriteLine("Введите числитель и знаменатель дроби(последовательно):");
        a = Convert.ToDouble(Console.ReadLine());
        b = Convert.ToDouble(Console.ReadLine());

        del = ReduceFraction(a, b);
        Console.WriteLine("a = " + a / del);
        Console.WriteLine("b = " + b / del);
        Console.ReadKey();
    }
}

```

Задание 8

Вывести на экран все числа из отрезка $[a;b]$, имеющие наибольшее количество делителей;

Задание 9

Дано натуральное число N . Вывести на экран:

Предшествующее по отношению к нему простое число

```
class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("Введите число:");

        int n = Convert.ToInt32(Console.ReadLine());
        Console.WriteLine($"Предшествующее к нему простое число: {PrimeLessN(n)}");
        Console.ReadKey();
    }
    static int PrimeLessN(int n)
    {
        int bufer;
        bool[] num = new bool[n + 1];
        for (int i = 0; i < n; i++)
        {
            num[i] = true;
        }

        int divMax = (int)Math.Floor(Math.Pow(n, 0.5));

        for (int i = 2; i < divMax; i++)
        {
            // если число является простым
            if (num[i])
            {
                bufer = 2 * i;
                while (bufer < n)
                {
                    num[bufer] = false;
                    bufer += i;
                }
            }
        }
        // поиск максимального простого, меньшего n
        for (int i = n; i >= 2; i--)
        {
            if (num[i]) return i;
        }
        Console.WriteLine("Число не найдено");
        return 0;
    }
}
```

```

C:\Users\8-bit\Desktop\My_Code\C#\Learn\mary\mary\mary\
Введите число:
558
Предшествующее к нему простое число: 557
0 references

```

Задание 10

На отрезке $[a;b]$ найти все пары соседних натуральных чисел: x и y ,
 Сумма которых является составным числом;

Контрольная работа

Реализация алгоритмов

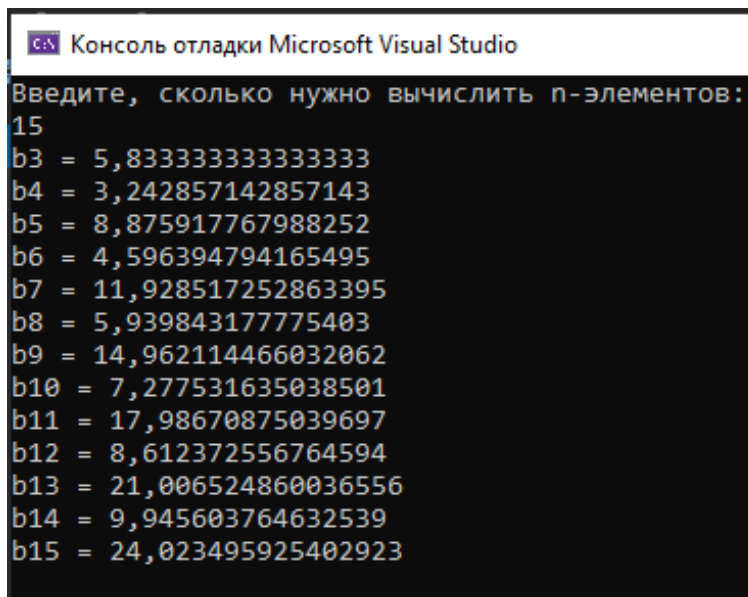
Задание 1. Написать консольное приложение, вычисляющее первые n элементов заданной последовательности

$$b_1 = 4, b_2 = 2, b_n = \frac{b_{n-2}}{n} + \frac{n^2}{b_{n-1}}$$

```
class Program
{
    static double b1 = 4;
    static double b2 = 2;
    static double bn = 0;

    static double F(int i)
    {
        bn = (b1/i) + (Math.Pow(i, 2)/b2);
        b1 = b2;
        b2 = bn;
        return bn;
    }
    static void Main(string[] args)
    {
        Console.WriteLine("Введите, сколько нужно вычислить n-элементов: ");
        int n = Convert.ToInt32(Console.ReadLine());

        for (int i = 3; i <= n; ++i)
        {
            bn = F(i);
            Console.WriteLine($"b{i} = {bn}");
        }
    }
}
```



```
Консоль отладки Microsoft Visual Studio
Введите, сколько нужно вычислить n-элементов:
15
b3 = 5,833333333333333
b4 = 3,242857142857143
b5 = 8,875917767988252
b6 = 4,596394794165495
b7 = 11,928517252863395
b8 = 5,939843177775403
b9 = 14,962114466032062
b10 = 7,277531635038501
b11 = 17,98670875039697
b12 = 8,612372556764594
b13 = 21,006524860036556
b14 = 9,945603764632539
b15 = 24,023495925402923
```

$$b_1 = 0, b_{2n} = b_{2n-1} + 3, b_{2n+1} = 2b_{2n}$$

```

class Program
{
    static double b1 = 0;
    static double b2n = 2;
    static double b2n1 = 0;

    static double b_2n(int i)
    {
        b2n = b1 + 3;
        b1 = b2n;
        return b2n;
    }
    static double b_2n1(int i, double b2n)
    {
        b2n1 = 2 * b2n;
        b2n = b2n1;
        return b2n1;
    }
    static void Main(string[] args)
    {

        Console.WriteLine("Введите, сколько нужно вычислить n-элементов: ");
        int n = Convert.ToInt32(Console.ReadLine());

        Console.WriteLine($"{b1}");
        for (int i = 2; i <= n; ++i)
        {
            b2n = b_2n(i);
            b2n1 = b_2n1(i, b2n);
            Console.WriteLine($"{b2n1}");
        }
    }
}

```

Задание 2. Вычислить бесконечную сумму ряда с заданной точностью $\varepsilon > 0$.

$$\sum_{i=1}^{\infty} \frac{(-3)^{2i}}{3i!}$$

```

class Program
{
    static double s = 0;
    static int factorial = 1;
    static int a = -1;

    static int Factorial(int i)
    {
        factorial *= (3*i);
        return factorial;
    }

    static double Summ(int i, int factorial)
    {
        s = Math.Pow(-3, 2*i)/factorial;
        return s;
    }
    static void Main(string[] args)
    {

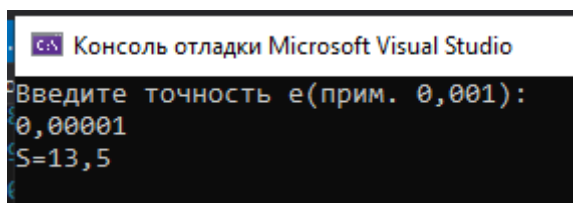
```

```

Console.WriteLine("Введите точность e(прим. 0,001): ");
double e = Convert.ToDouble(Console.ReadLine());

for (int i = 2; Math.Abs(a) >= e; ++i)
{
    factorial = Factorial(i);
    s += Summ(i, factorial);
    a /= -i;
}
Console.WriteLine($"S={s}");
}
}

```



$$13. S = \sum_{i=1}^{\infty} \frac{(-1)^{2i}}{i(i+1)(i+2)}$$

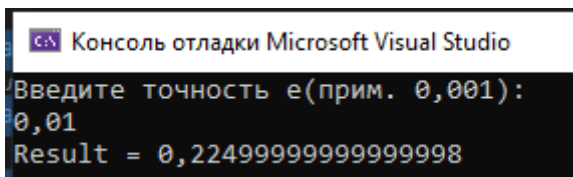
```

class Program
{
    static double s = 1;
    static double Summ(double i)
    {
        return 1 / ((i * (i + 1)) * (i + 2));
    }
    static void Main(string[] args)
    {
        Console.WriteLine("Введите точность e(прим. 0,001): ");
        double e = Convert.ToDouble(Console.ReadLine());

        double i = 1;
        while (Math.Abs(Summ(i)) >= e)
        {
            if (i == 1)
            {
                s -= 1;
            }

            s += Summ(i);
            i++;
        }
        Console.WriteLine($"Result = {s}");
    }
}

```



Задание 3. Вычислить и вывести на экран значение функции $F(x)$ на отрезке $[a,b]$ с шагом $h=0.1$ и точностью ε . Результат работы программы представить в виде таблицы:

$$F(x) = 2 \left(\frac{x-1}{x+1} + \frac{(x-1)^3}{3(x+1)^3} + \frac{(x-1)^5}{5(x+1)^5} + \dots \right), \quad x \in [1; 2].$$

$$11. F(x) = 2 \left(\frac{x-1}{x+1} + \frac{(x-1)^3}{3(x+1)^3} + \frac{(x-1)^5}{5(x+1)^5} \dots \right), \quad x \in [1; 2]$$

```
class Program
{
    static double F = 1;
    static double a = -1;
    static double h = 1.0;
    static int count = 0;

    static double Func(double x, double n)
    {
        F = Math.Pow((x - 1), n) / (n * Math.Pow((x + 1), n));
        return F;
    }

    static void Main(string[] args)
    {
        string sx = "Значение x";
        string sfx = "Значение F(x)";
        string scout = "Количество слагаемых n";

        Console.WriteLine("Введите точность e (прим. 0.001): ");
        double e = Convert.ToDouble(Console.ReadLine());

        Console.WriteLine($"{sx,30} | {sfx,30} | {scout,30}");
        for (int i = 1; Math.Abs(a) >= e; ++i)
        {
            F += Func(h, i);
            count += 1;
            a /= -i;
            Console.WriteLine($"{h,30} | {F,30} | {count,30}");

            h += 0.1;
            if (h > 2)
            {
                break;
            }
        }
    }
}
```

Консоль отладки Microsoft Visual Studio

Введите точность e (прим. 0.001):
0,0005

Значение x	Значение F(x)	Количество слагаемых n
1	1	1
1,1	1,0000000979407708	2
1,2000000000000002	1,0000001126024314	3
1,3000000000000003	1,0000001150308764	4
1,4000000000000004	1,000000115532033	5
1,5000000000000004	1,0000001156580638	6
1,6000000000000005	1,000000115695441	7

					ККОО.ОАXXXX.000	Лист
						42
Изм.	Лист	№ докум.	Подпись	Дата		

Лабораторная работа №13.2

Тема. Работа с массивами

Цель: научиться использовать массив в виде параметра метода.

Задание: разработать консольное приложение по варианту.

Дополнение к заданию.

При вводе исходного массива вначале следует ввести его размер (одно число для одномерных массивов, два числа — количество строк и столбцов — для двумерных массивов-матриц), а затем — все его элементы.

Если в задании явно не указывается размер одномерного массива, являющегося параметром метода -процедуры или метода - функции, то предполагается, что этот размер может изменяться в пределах от 1 до 10. Для двумерных массивов-матриц предполагается, что число их строк и столбцов может меняться от 1 до 10.

При разработке методов - процедур, выполняющих преобразование массива, не следует использовать вспомогательный массив того же размера.

Для заполнения и вывода массива на экран разработать дополнительные методы. Для выполнения преобразования массива тоже разработать отдельный метод.

Критерии оценки:

Разработаны все необходимые методы и с их помощью выполнено одно задание из раздела одномерные массивы. На дополнительные вопросы даны положительные ответы – оценка удовлетворительно

Разработаны все необходимые методы и с их помощью выполнено одно задание из раздела одномерные массивы и частично выполнено задание из раздела двумерные массивы. На дополнительные вопросы даны положительные ответы – оценка хорошо

					ККОО.ОАXXXX.000	Лист
Изм.	Лист	№ докум.	Подпись	Дата		43

Разработаны все необходимые методы и с их помощью выполнено по одному заданию из обоих разделов, ответы на дополнительные вопросы правильные – оценка отлично

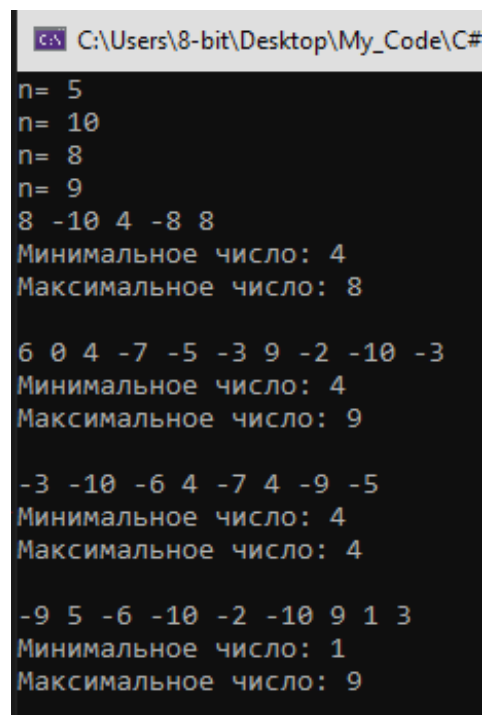
1. Заданы массивы A1(6), A2(7), A3(4), A4(6), состоящие из положительных и отрицательных чисел. Найти минимальное значение положительных чисел массива и определить максимальное среди них. Нахождение минимального значения положительных чисел массива, заполнение массивов и вывод на печать оформить в виде методов.

```
class Program
{
    static void Print(int[] a)
    {
        foreach (int elem in a)
        {
            Console.Write($"{elem} ");
        }
        Console.WriteLine();
    }
    static void Input(out int[] a)
    {
        Console.Write("n= ");
        int n = int.Parse(Console.ReadLine());
        a = new int[n];
        Random rnd = new Random();
        for (int i = 0; i < n; i++)
        {
            a[i] = rnd.Next(-10, 10);
        }
    }
    static void Min(int[] a)
    {
        int min = 10;
        int max = 0;
        foreach (int elem in a)
        {
            if (elem > 0)
            {
                if (elem < min)
                {
                    min = elem;
                }
                if (elem > max)
                {
                    max = elem;
                }
            }
        }
        Console.WriteLine($"Минимальное число: {min}");
        Console.WriteLine($"Максимальное число: {max}");
    }
    static void Main(string[] args)
    {
        int[] A1;
        int[] A2;
```

```

int[] A3;
int[] A4;
Input(out A1);
Input(out A2);
Input(out A3);
Input(out A4);
Print(A1);
Min(A1);
Console.WriteLine();
Print(A2);
Min(A2);
Console.WriteLine();
Print(A3);
Min(A3);
Console.WriteLine();
Print(A4);
Min(A4);
Console.ReadKey();
}
}

```



```

C:\Users\8-bit\Desktop\My_Code\C#
n= 5
n= 10
n= 8
n= 9
8 -10 4 -8 8
Минимальное число: 4
Максимальное число: 8

6 0 4 -7 -5 -3 9 -2 -10 -3
Минимальное число: 4
Максимальное число: 9

-3 -10 -6 4 -7 4 -9 -5
Минимальное число: 4
Максимальное число: 4

-9 5 -6 -10 -2 -10 9 1 3
Минимальное число: 1
Максимальное число: 9

```

2. Заданы массивы A(7),B(4),C(12), состоящие из положительных и отрицательных чисел. Определить, в каком месте больше среднее арифметическое положительных чисел. Подсчет среднего арифметического положительных чисел массива оформить в виде метода. Заполнение массивов и вывод на печать оформить в виде методов.

```

using System;

namespace ConsoleApp6
{
    class Program
    {

```

					KKOO.OAXXXX.000	Лист
						45
Изм.	Лист	№ докум.	Подпись	Дата		

```

static void Print(int[] a)
{
    foreach (int elem in a)
    {
        Console.Write($"{elem} ");
    }
    Console.WriteLine();
}
static void Sred(out int[] a)
{
    Console.Write("n= ");
    int n = int.Parse(Console.ReadLine());
    a = new int[n];
    Random rnd = new Random();
    for (int i = 0; i < n; i++)
    {
        a[i] = rnd.Next(-10, 10);
    }
}
static int Min(int[] a)
{
    int sr = 0;
    int v = 0;
    foreach (int elem in a)
    {
        if (elem > 0)
        {
            sr += elem;
            v++;
        }
    }
    sr = sr / v;
    return sr;
}
static void Main(string[] args)
{
    int[] A1;
    int[] A2;
    int[] A3;
    Sred(out A1);
    Sred(out A2);
    Sred(out A3);
    Print(A1);
    Console.WriteLine($"Среднее арифметическое массива: {Min(A1)}");
    Console.WriteLine();
    Print(A2);
    Console.WriteLine($"Среднее арифметическое массива: {Min(A2)}");
    Console.WriteLine();
    Print(A3);
    Console.WriteLine($"Среднее арифметическое массива: {Min(A3)}");
    Console.WriteLine();
    if (Min(A1) > Min(A2) && Min(A1) > Min(A3))
    {
        Console.WriteLine($"Среднее арифметическое первого массива самое большое");
    }
    if (Min(A1) < Min(A2) && Min(A2) > Min(A3))
    {
        Console.WriteLine($"Среднее арифметическое второго массива самое большое");
    }
    else

```

```

    {
        Console.WriteLine($"Среднее арифметическое третьего массива самое большое");
    }
    Console.ReadKey();
}
}
}

```

```

n= 7
n= 4
n= 12
7 -6 0 1 -8 -6 0
Среднее арифметическое массива: 4

-10 -10 5 4
Среднее арифметическое массива: 4

-10 -5 -10 3 3 -5 7 -9 8 -1 -8 -4
Среднее арифметическое массива: 5

Среднее арифметическое третьего массива самое большое

```

3. Заданы массивы $X_1(5)$, $X_2(7)$, $X_3(7)$, состоящие из положительных и отрицательных чисел. Вычислить среднеарифметическое сумм положительных элементов. Вычисление суммы положительных элементов оформить в виде метода. Заполнение массивов и вывод на печать оформить в виде методов.

```

class Program
{
    static void Print(int[] a)
    {
        foreach (int elem in a)
        {
            Console.Write($"{elem} ");
        }
        Console.WriteLine();
    }
    static void Sred(out int[] a)
    {
        Console.Write("n= ");
        int n = int.Parse(Console.ReadLine());
        a = new int[n];
        Random rnd = new Random();
        for (int i = 0; i < n; i++)
        {
            a[i] = rnd.Next(-10, 10);
        }
    }
    static int Min(int[] a)
    {
        int sum = 0;
        int x = 0;
        foreach (int elem in a)
        {
            if (elem > 0)

```

```

        {
            sum += elem;
            x++;
        }
    }
    sum = sum / x;
    return sum;
}
static void Main(string[] args)
{
    int[] A1;
    int[] A2;
    int[] A3;
    Sred(out A1);
    Sred(out A2);
    Sred(out A3);
    Print(A1);
    Console.WriteLine($"Среднеарифметическое сумм положительных элементов: {Min(A1)}");
    Console.WriteLine();
    Print(A2);
    Console.WriteLine($"Среднеарифметическое сумм положительных элементов: {Min(A2)}");
    Console.WriteLine();
    Print(A3);
    Console.WriteLine($"Среднеарифметическое сумм положительных элементов: {Min(A3)}");
    Console.WriteLine();
    Console.ReadKey();
}
}

```

```

n= 5
n= 7
n= 7
-10 -8 -4 -7 7
Среднеарифметическое сумм положительных элементов: 7
7 7 2 3 -2 -2 -8
Среднеарифметическое сумм положительных элементов: 4
9 7 -7 2 -3 -7 -6
Среднеарифметическое сумм положительных элементов: 6

```

4. Заданы массивы A(7), B(4), C(9), состоящие из положительных и отрицательных чисел. Определить, в каком массиве больше отрицательных элементов. Подсчет количества отрицательных элементов оформить в виде метода. Заполнение массивов и вывод на печать оформить в виде методов.

```

class Program
{
    static void Print(int[] a)
    {
        foreach (int elem in a)
        {
            Console.Write($"{elem} ");
        }
        Console.WriteLine();
    }
    static void Sred(out int[] a)
    {
        Console.Write("n= ");
        int n = int.Parse(Console.ReadLine());
    }
}

```

					KKOO.OAXXXX.000	Лист
						48
Изм.	Лист	№ докум.	Подпись	Дата		


```

a = new int[n];
Random rnd = new Random();
for (int i = 0; i < n; i++)
{
    a[i] = rnd.Next(-10, 10);
}
}
static int Min(int[] a)
{
    int v = 0;
    foreach (int elem in a)
    {
        if (elem < 0)
        {
            v++;
        }
    }
    return v;
}
static void Main(string[] args)
{
    int[] A1;
    int[] A2;
    int[] A3;
    Sred(out A1);
    Sred(out A2);
    Sred(out A3);
    Print(A1);
    Console.WriteLine($"Количество отрицательных элементов: {Min(A1)}");
    Console.WriteLine();
    Print(A2);
    Console.WriteLine($"Количество отрицательных элементов: {Min(A2)}");
    Console.WriteLine();
    Print(A3);
    Console.WriteLine($"Количество отрицательных элементов: {Min(A3)}");
    Console.WriteLine();
    if (Min(A1) > Min(A2) && Min(A1) > Min(A3))
    {
        Console.WriteLine($"В первом массиве самое большое количество отрицательных элементов");
    }
    if (Min(A1) < Min(A2) && Min(A2) > Min(A3))
    {
        Console.WriteLine($"Во втором массиве самое большое количество отрицательных элементов");
    }
    else
    {
        Console.WriteLine($"В третьем массиве самое большое количество отрицательных элементов");
    }
    Console.ReadKey();
}
}

```

```

n= 7
Print(int[] a)
n= 4
n= 9
4 2 -9 3 -2 -4 -3
Количество отрицательных элементов: 4
Console.Write($"{elem} ");
-2 -10 7 -9
Количество отрицательных элементов: 3
2 -3 7 9 -3 8 9 -3 -5
Количество отрицательных элементов: 4
В третьем массиве самое большое количество отрицательных элементов

```

5. Заданы массивы X(8),Y(7),Z(9), состоящие из положительных и отрицательных чисел. Найти максимальное значение отрицательных чисел в каждом массиве и определить минимальное среди них. Определение максимального значения отрицательных чисел в массиве оформить в виде метода. Заполнение массивов и вывод на печать оформить в виде методов.

```

class Program
{
    static void Print(int[] a)
    {
        foreach (int elem in a)
        {
            Console.Write($"{elem} ");
        }
        Console.WriteLine();
    }
    static void Sred(out int[] a)
    {
        Console.Write("n= ");
        int n = int.Parse(Console.ReadLine());
        a = new int[n];
        Random rnd = new Random();
        for (int i = 0; i < n; i++)
        {
            a[i] = rnd.Next(-10, 10);
        }
    }
    static void Min(int[] a)
    {
        int min = 0;
        int max = -10;
        foreach (int elem in a)
        {
            if (elem < 0)
            {
                if (elem < min)
                {
                    min = elem;
                }
            }
            if (elem > max)
            {
                max = elem;
            }
        }
    }
}

```

```

    }
}
Console.WriteLine($"Минимальное значение из отрицательных чисел: {min}");
Console.WriteLine($"Максимальное значение из отрицательных чисел: {max}");
}
static void Main(string[] args)
{
    int[] A1;
    int[] A2;
    int[] A3;
    Sred(out A1);
    Sred(out A2);
    Sred(out A3);
    Print(A1);
    Min(A1);
    Console.WriteLine();
    Print(A2);
    Min(A2);
    Console.WriteLine();
    Print(A3);
    Min(A3);
    Console.WriteLine();
    Console.ReadKey();
}
}

```

```

n= 8
id Print(int[] a)
n= 7
n= 9
2 -1 -7 -1 7 4 4 -4
Минимальное значение из отрицательных чисел: -7
Максимальное значение из отрицательных чисел: -1

-5 1 -6 8 9 -8 -9
Минимальное значение из отрицательных чисел: -9
Максимальное значение из отрицательных чисел: -5
id Sred(out int[] a)
6 0 -10 5 -9 8 6 6 -6
Минимальное значение из отрицательных чисел: -10
Максимальное значение из отрицательных чисел: -6

```

6. Заданы массивы A(5), B(7), C(6), состоящие из положительных и отрицательных чисел. Определить в каком массиве, больше чисел кратных 3. Подсчет количества кратных чисел оформить в виде метода. Заполнение массивов и вывод на печать оформить в виде методов.

```

class Program
{
    static void Print(int[] a)
    {
        foreach (int elem in a)
        {
            Console.Write($"{elem} ");
        }
        Console.WriteLine();
    }
    static void Sred(out int[] a)
    {
        Console.Write("n= ");
        int n = int.Parse(Console.ReadLine());
    }
}

```

					KKOO.OAXXXX.000	Лист
Изм.	Лист	№ докум.	Подпись	Дата		51

```

a = new int[n];
Random rnd = new Random();
for (int i = 0; i < n; i++)
{
    a[i] = rnd.Next(-10, 10);
}
}
static int Min(int[] a)
{
    int v = 0;
    foreach (int elem in a)
    {
        if (elem % 3 == 0)
        {
            v++;
        }
    }
    return v;
}
static void Main(string[] args)
{
    int[] A1;
    int[] A2;
    int[] A3;
    Sred(out A1);
    Sred(out A2);
    Sred(out A3);
    Print(A1);
    Console.WriteLine($"Количество чисел кратных 3: {Min(A1)}");
    Console.WriteLine();
    Print(A2);
    Console.WriteLine($"Количество чисел кратных 3: {Min(A2)}");
    Console.WriteLine();
    Print(A3);
    Console.WriteLine($"Количество чисел кратных 3: {Min(A3)}");
    Console.WriteLine();
    if (Min(A1) > Min(A2) && Min(A1) > Min(A3))
    {
        Console.WriteLine($"В первом массиве самое большое количество чисел кратных 3");
    }
    if (Min(A1) < Min(A2) && Min(A2) > Min(A3))
    {
        Console.WriteLine($"Во втором массиве самое большое количество чисел кратных 3");
    }
    else
    {
        Console.WriteLine($"В третьем массиве самое большое количество чисел кратных 3");
    }
    Console.ReadKey();
}
}

```

```

n= 5
n= 7
n= 6
5 -6 -5 6 5
Количество чисел кратных 3: 2
-10 6 -10 8 1 -7 -9
Количество чисел кратных 3: 2
5 1 5 -2 3 -9
Количество чисел кратных 3: 2
В третьем массиве самое большое количество чисел кратных 3

```

7. Заданы массивы A(10) и B(7). Определить, в каком массиве и на каком месте стоит наибольшее число. Поиск максимального элемента и его индекса в массиве оформить в виде метода. Заполнение массивов и вывод на печать оформить в виде методов.

```

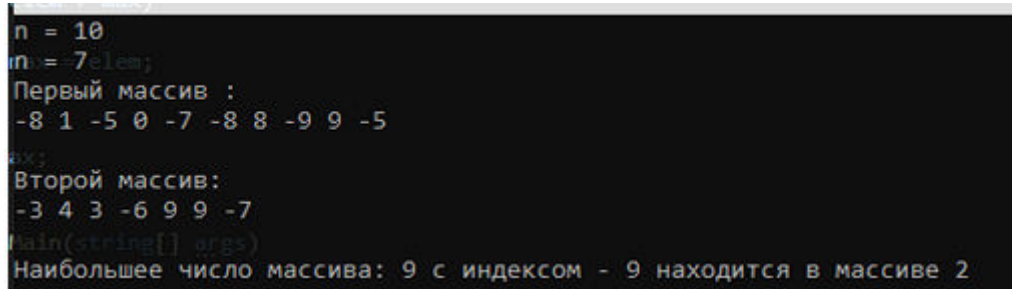
class Program
{
    static void Print(int[] a)
    {
        foreach (int elem in a)
        {
            Console.Write($"{elem} ");
        }
        Console.WriteLine();
    }
    static void Sred(out int[] a)
    {
        Console.Write("n = ");
        int n = int.Parse(Console.ReadLine());
        a = new int[n];
        Random rnd = new Random();
        for (int i = 0; i < n; i++)
        {
            a[i] = rnd.Next(-10, 10);
        }
    }
    static int Max(int[] a)
    {
        int max = 0;
        foreach (int elem in a)
        {
            if (elem > max)
            {
                max = elem;
            }
        }
        return max;
    }
    static void Main(string[] args)
    {
        int[] A;
        int[] B;
        Sred(out A);
        Sred(out B);
    }
}

```

```

Console.WriteLine("Первый массив :");
Print(A);
Console.WriteLine();
Console.WriteLine("Второй массив:");
Print(B);
Console.WriteLine();
if (Max(A) > Max(B))
{
    int index = Array.IndexOf(A, (Max(A)));
    Console.WriteLine($"Наибольшее число массива: {Max(A)} с индексом - {index + 1} находится
в массиве 1");
}
else
{
    int index = Array.IndexOf(A, Max(B));
    Console.WriteLine($"Наибольшее число массива: {Max(B)} с индексом - {index + 1} находится в
массиве 2");
}
Console.ReadKey();
}
}

```



```

n = 10
m = 7 elem;
Первый массив :
-8 1 -5 0 -7 -8 8 -9 9 -5
Второй массив:
-3 4 3 -6 9 9 -7
Наибольшее число массива: 9 с индексом - 9 находится в массиве 2

```

8. Заданы массивы A(5), B(4), C(7). Определить, в каком массиве больше чисел, попадающих в интервал от -1 до 1. Подсчет количества чисел, попадающих в интервал от -1 до 1 оформить в виде метода. Заполнение массивов и вывод на печать оформить в виде методов.

```

class Program
{
    static void Print(int[] a)
    {
        foreach (int elem in a)
        {
            Console.Write($"{elem} ");
        }
        Console.WriteLine();
    }
    static void Sred(out int[] a)
    {
        Console.Write("n= ");
        int n = int.Parse(Console.ReadLine());
        a = new int[n];
        Random rnd = new Random();
        for (int i = 0; i < n; i++)
        {
            a[i] = rnd.Next(-10, 10);
        }
    }
    static int Min(int[] a)
    {

```

```

int v = 0;
foreach (int elem in a)
{
    if ((elem > -0.9) && (elem <= 1))
    {
        v++;
    }
}
return v;
}
static void Main(string[] args)
{
    int[] A1;
    int[] A2;
    int[] A3;
    Sred(out A1);
    Sred(out A2);
    Sred(out A3);
    Print(A1);
    Console.WriteLine($"Количество чисел, попадающих в интервал от -1 до 1: {Min(A1)}");
    Console.WriteLine();
    Print(A2);
    Console.WriteLine($"Количество чисел, попадающих в интервал от -1 до 1: {Min(A2)}");
    Console.WriteLine();
    Print(A3);
    Console.WriteLine($"Количество чисел, попадающих в интервал от -1 до 1: {Min(A3)}");
    Console.WriteLine();
    if (Min(A1) > Min(A2) && Min(A1) > Min(A3))
    {
        Console.WriteLine($"В первом массиве самое большое количество чисел, попадающих в
интервал от -1 до 1");
    }
    if (Min(A1) < Min(A2) && Min(A2) > Min(A3))
    {
        Console.WriteLine($"Во втором массиве самое большое количество чисел, попадающих в
интервал от -1 до 1");
    }
    else
    {
        Console.WriteLine($"В третьем массиве самое большое количество чисел, попадающих в
интервал от -1 до 1");
    }
    Console.ReadKey();
}
}

```

```

n= 5
4 2 5 -4
n= 4
9 -5 3 -7
n= 7
7 8 -2 9 1 5
Количество чисел, попадающих в интервал от -1 до 1: 0
4 2 5 -4
Количество чисел, попадающих в интервал от -1 до 1: 0
9 -5 3 -7 1 -6 8
Количество чисел, попадающих в интервал от -1 до 1: 1
В третьем массиве самое большое количество чисел, попадающих в интервал от -1 до 1

```

9. Заданы массивы А (10) и В (7). Определить, в каком массиве и на каком месте стоит наименьшее число. Поиск минимального элемента и его

индекса в массиве оформить в виде метода. Заполнение массивов и вывод на печать оформить в виде методов.

```
class Program
{
    static void Print(int[] a)
        // печать массива
    {
        foreach (int elem in a)
        {
            Console.Write($"{elem} ");
        }
        Console.WriteLine();
    }
    static void Input(out int[] a)
        // ввод размерности массива и его заполнение
    {
        Console.Write("n= ");
        int n = int.Parse(Console.ReadLine());
        a = new int[n];
        Random rnd = new Random();
        for (int i = 0; i < n; i++)
        {
            a[i] = rnd.Next(-10, 10);
        }
    }
    static int Min(int[] a)
        //нахождение минимального элемента
    {
        int min = 0;
        foreach (int elem in a)
        {
            if (elem < min)
            {
                min = elem;
            }
        }
        return min;
    }

    static int IndexMin(int[] a)
    {
        int indexMin = Array.IndexOf(a, Min(a));
        return indexMin;
    }
    static void Main(string[] args)
    {
        int[] A1;
        int[] B1;

        Input(out A1);
        Input(out B1);

        Print(A1);
        Print(B1);

        int IndexMinA = IndexMin(A1);
        int IndexMinB = IndexMin(B1);
    }
}
```



```

        Console.WriteLine($"В массиве A1 найдено минимальное число {Min(A1)} и его индекс {IndexMinA}");
        Console.WriteLine($"В массиве B1 найдено минимальное число {Min(B1)} и его индекс {IndexMinB}");
    }
}

```

The screenshot shows the Microsoft Visual Studio Debug Console with the following output:

```

n= 10
n= 7
6 7 9 2 0 6 1 2 -5 -1
3 7 0 -8 0 -5 10
В массиве A1 найдено минимальное число -5 и его индекс 8
В массиве B1 найдено минимальное число -8 и его индекс 2

```

10. Заданы массивы A (15), B (7), C (10), состоящие из положительных и отрицательных чисел. Определить в каком массиве, больше четных чисел. Подсчет количества четных чисел оформить в виде метода. Заполнение массивов и вывод на печать оформить в виде методов.

```

class Program
{
    static void Print(int[] a)
    {
        // печать массива
        foreach (int elem in a)
        {
            Console.Write($"{elem} ");
        }
        Console.WriteLine();
    }

    static void Input(out int[] a)
    {
        // ввод размерности массива и его заполнение
        Console.Write("n= ");
        int n = int.Parse(Console.ReadLine());
        a = new int[n];
        Random rnd = new Random();
        for (int i = 0; i < n; i++)
        {
            a[i] = rnd.Next(-10, 10);
        }
    }

    static int CountChetElem(int[] a)
    {
        int count = 0;
        for (int i = 0; i < a.Length; i++)
        {
            if (a[i] % 2 == 0)
            {
                count++;
            }
        }
        return count;
    }
}

```

```

    }
    static void Main(string[] args)
    {
        int[] A1;
        int[] B1;
        int[] C1;

        Input(out A1);
        Input(out B1);
        Input(out C1);

        Print(A1);
        Print(B1);
        Print(C1);

        int a_chetcount = CountChetElem(A1);
        int b_chetcount = CountChetElem(B1);
        int c_chetcount = CountChetElem(C1);

        int[] CountArray = new int[3] { a_chetcount, b_chetcount, c_chetcount };

        int max = 0;
        if (a_chetcount < b_chetcount)
        {
            max = b_chetcount;
            if (max < c_chetcount)
            {
                max = c_chetcount;
            }
        }
        else
        {
            max = a_chetcount;
            if (max < c_chetcount)
            {
                max = c_chetcount;
            }
        }

        for (int i = 0; i < CountArray.Length; i++)
        {
            if (max == a_chetcount)
            {
                Console.WriteLine($"В массиве А найдено наибольшее количество четных чисел: {max}");
                break;
            }
            if (max == b_chetcount)
            {
                Console.WriteLine($"В массиве В найдено наибольшее количество четных чисел: {max}");
                break;
            }
            if (max == c_chetcount)
            {
                Console.WriteLine($"В массиве С найдено наибольшее количество четных чисел: {max}");
                break;
            }
        }
    }
}

```

```

Microsoft Visual Studio Debug Console
n= 15
n= 7
n= 10
-7 -9 3 -9 4 -1 -7 -8 -8 -6 4 -1 -2 4 9
7 -2 5 -10 4 5 -6
-4 -9 -5 -7 7 -4 -3 6 6 -3
В массиве A найдено наибольшее количество четных чисел: 7

```

11. Заданы массивы А (5), В (7), С (6), состоящие из положительных чисел. Определить в каком массиве, больше чисел меньших 30. Подсчет количества чисел меньших 30 оформить в виде метода. Заполнение массивов и вывод на печать оформить в виде методов.

```

class Program
{
    static void Print(int[] a)
    // печать массива
    {
        foreach (int elem in a)
        {
            Console.Write($"{elem} ");
        }
        Console.WriteLine();
    }
    static void Input(out int[] a)
    // ввод размерности массива и его заполнение
    {
        Console.Write("n= ");
        int n = int.Parse(Console.ReadLine());
        a = new int[n];
        Random rnd = new Random();
        for (int i = 0; i < n; i++)
        {
            a[i] = rnd.Next(0, 100);
        }
    }

    static int CountElem(int[] a)
    {
        int count = 0;
        for (int i = 0; i < a.Length; i++)
        {
            if (a[i] < 30)
            {
                count++;
            }
        }
        return count;
    }
    static void Main(string[] args)
    {
        int[] A1;
        int[] B1;
        int[] C1;

        Input(out A1);

```

```

Input(out B1);
Input(out C1);

Print(A1);
Print(B1);
Print(C1);

int a_chetcount = CountElem(A1);
int b_chetcount = CountElem(B1);
int c_chetcount = CountElem(C1);

int[] CountArray = new int[3] { a_chetcount, b_chetcount, c_chetcount };

int max = 0;
if (a_chetcount < b_chetcount)
{
    max = b_chetcount;
    if (max < c_chetcount)
    {
        max = c_chetcount;
    }
}
else
{
    max = a_chetcount;
    if (max < c_chetcount)
    {
        max = c_chetcount;
    }
}

for (int i = 0; i < CountArray.Length; i++)
{
    if (max == a_chetcount)
    {
        Console.WriteLine($"В массиве А найдено наибольшее количество чисел, меньших 30:
{max}");
        break;
    }
    if (max == b_chetcount)
    {
        Console.WriteLine($"В массиве В найдено наибольшее количество чисел, меньших 30:
{max}");
        break;
    }
    if (max == c_chetcount)
    {
        Console.WriteLine($"В массиве С найдено наибольшее количество чисел, меньших 30:
{max}");
        break;
    }
}
}

```

```

Microsoft Visual Studio Debug Console
n= 5
n= 7
n= 6
71 48 64 18 2
27 7 34 38 7 22 89
17 86 10 96 67 70
B массиве B найдено наибольшее количество чисел, меньших 30: 4

```

12. Заданы массивы A1 (6), A2 (7), A3 (14). Найти среднее арифметическое значение элементов массива и определить максимальное среди них. Нахождение среднего арифметического элементов массива оформить в виде метода. Заполнение массивов и вывод на печать оформить в виде методов.

13. Заданы массивы A (6), B (7), C (14). Найти какой из массивов имеет наибольшее число отрицательных элементов. Нахождение количества отрицательных элементов массива оформить в виде метода. Заполнение массивов и вывод на печать оформить в виде методов.

```

class Program
{
    static void Print(int[] a)
    // печать массива
    {
        foreach (int elem in a)
        {
            Console.Write($"{elem} ");
        }
        Console.WriteLine();
    }
    static void Input(out int[] a)
    // ввод размерности массива и его заполнение
    {
        Console.Write("n= ");
        int n = int.Parse(Console.ReadLine());
        a = new int[n];
        Random rnd = new Random();
        for (int i = 0; i < n; i++)
        {
            a[i] = rnd.Next(-100, 100);
        }
    }

    static int CountElem(int[] a)
    {
        int count = 0;
        for (int i = 0; i < a.Length; i++)
        {
            if (a[i] < 0)
            {
                count++;
            }
        }
    }
}

```

```

    }
    }
    return count;
}
static void Main(string[] args)
{
    int[] A1;
    int[] B1;
    int[] C1;

    Input(out A1);
    Input(out B1);
    Input(out C1);

    Print(A1);
    Print(B1);
    Print(C1);

    int a_chetcount = CountElem(A1);
    int b_chetcount = CountElem(B1);
    int c_chetcount = CountElem(C1);

    int[] CountArray = new int[3] { a_chetcount, b_chetcount, c_chetcount };

    int max = 0;
    if (a_chetcount < b_chetcount)
    {
        max = b_chetcount;
        if (max < c_chetcount)
        {
            max = c_chetcount;
        }
    }
    else
    {
        max = a_chetcount;
        if (max < c_chetcount)
        {
            max = c_chetcount;
        }
    }

    for (int i = 0; i < CountArray.Length; i++)
    {
        if (max == a_chetcount)
        {
            Console.WriteLine($"В массиве А найдено наибольшее количество отрицательных элементов:
{max}");
            break;
        }
        if (max == b_chetcount)
        {
            Console.WriteLine($"В массиве В найдено наибольшее количество отрицательных элементов:
{max}");
            break;
        }
        if (max == c_chetcount)
        {
            Console.WriteLine($"В массиве С найдено наибольшее количество отрицательных элементов:
{max}");
            break;
        }
    }
}

```

```

    }
}
}

Microsoft Visual Studio Debug Console
n= 6
n= 7
n= 14
-24 -52 -76 17 30 -45
-93 97 -39 99 -4 -64 -20
55 16 11 26 98 23 89 24 -86 44 23 -8 44 -16
В массиве В найдено наибольшее количество отрицательных элементов: 5

```

14. Заданы массивы А (5), В (4), С (7). Определить, в каком массиве меньше чисел, попадающих в интервал от 10 до 20. Подсчет количества чисел попадающих в интервал от 10 до 20 оформить в виде метода. Заполнение массивов и вывод на печать оформить в виде методов.

15. Заданы массивы А (15), В (7), С (10), состоящие из положительных и отрицательных чисел. Определить в каком массиве, больше нечетных чисел. Подсчет количества нечетных чисел оформить в виде метода. Заполнение массивов и вывод на печать оформить в виде методов.

```

class Program
{
    static void Print(int[] a)
    // печать массива
    {
        foreach (int elem in a)
        {
            Console.Write($"{elem} ");
        }
        Console.WriteLine();
    }
    static void Input(out int[] a)
    // ввод размерности массива и его заполнение
    {
        Console.Write("n= ");
        int n = int.Parse(Console.ReadLine());
        a = new int[n];
        Random rnd = new Random();
        for (int i = 0; i < n; i++)
        {
            a[i] = rnd.Next(-10, 10);
        }
    }

    static int CountChetElem(int[] a)
    {
        int count = 0;
        for (int i = 0; i < a.Length; i++)

```

```

    {
        if (a[i] % 2 != 0)
        {
            count++;
        }
    }
    return count;
}
static void Main(string[] args)
{
    int[] A1;
    int[] B1;
    int[] C1;
    Input(out A1);
    Input(out B1);
    Input(out C1);
    Print(A1);
    Print(B1);
    Print(C1);
    int a_chetcount = CountChetElem(A1);
    int b_chetcount = CountChetElem(B1);
    int c_chetcount = CountChetElem(C1);

    int[] CountArray = new int[3] { a_chetcount, b_chetcount, c_chetcount };

    int max = 0;
    if (a_chetcount < b_chetcount)
    {
        max = b_chetcount;
        if (max < c_chetcount)
        {
            max = c_chetcount;
        }
    }
    else
    {
        max = a_chetcount;
        if (max < c_chetcount)
        {
            max = c_chetcount;
        }
    }
    for (int i = 0; i < CountArray.Length; i++)
    {
        if (max == a_chetcount)
        {
            Console.WriteLine($"В массиве А найдено наибольшее количество нечетных чисел: {max}");
            break;
        }
        if (max == b_chetcount)
        {
            Console.WriteLine($"В массиве В найдено наибольшее количество нечетных чисел: {max}");
            break;
        }
        if (max == c_chetcount)
        {
            Console.WriteLine($"В массиве С найдено наибольшее количество нечетных чисел: {max}");
            break;
        }
    }
}
}

```



```
System;
Microsoft Visual Studio Debug Console
n= 15
n= 7
n= 10
-5 -8 -2 4 6 9 5 3 -1 -7 -8 -7 0 -3 -8
8 -10 4 1 -9 1 4
8 -6 8 -2 -4 3 2 6 2 -9
В массиве A найдено наибольшее количество нечетных чисел: 8
```

16. Описать метод - функцию MinElem(X, n) целого типа, находящую минимальный элемент целочисленного массива X размера n. С помощью этого метода найти минимальные элементы массивов A, B, C размера na, nb, nc соответственно.

```
class Program
{
    static void Print(int[] a)
    // печать массива
    {
        foreach (int elem in a)
        {
            Console.Write($"{elem} ");
        }
        Console.WriteLine();
    }
    static void Input(out int[] a)
    // ввод размерности массива и его заполнение
    {
        Console.Write("n = ");
        int n = int.Parse(Console.ReadLine());
        a = new int[n];
        Random rnd = new Random();
        for (int i = 0; i < n; i++)
        {
            a[i] = rnd.Next(-30, 30);
        }
    }
    static int MinElem(int[] X)
    //нахождение минимального элемента
    {
        int min = X[1];

        foreach (int elem in X)
        {
            if (elem < min)
            {
                min = elem;
            }
        }
        return min;
    }

    static void Main(string[] args)
    {
        int[] A1;
```

```

int[] B1;
int[] C1;

Input(out A1);
Input(out B1);
Input(out C1);

Print(A1);
Print(B1);
Print(C1);

Console.WriteLine($"В массиве A1 найдено минимальное число {MinElem(A1)}");
Console.WriteLine($"В массиве B1 найдено минимальное число {MinElem(B1)}");
Console.WriteLine($"В массиве C1 найдено минимальное число {MinElem(C1)}");
}
}

```

```

Консоль отладки Microsoft Visual Studio
n = 5
n = 10
n = 15
-20 4 15 -17 -17
-8 -7 20 25 20 -6 -30 -13 -30 0
-14 -9 26 11 23 -25 21 -4 3 2 -1 0 -28 14 7
В массиве A1 найдено минимальное число -20
В массиве B1 найдено минимальное число -30
В массиве C1 найдено минимальное число -28

```

17. Описать метод - функцию MaxNum(X,n) целого типа, находящую номер максимального элемента вещественного массива X размера N. С помощью этого метода найти номера максимальных элементов массивов A, B, C размера na, nb, nc соответственно.

```

class Program
{
    static void Print(int[] a)
    // печать массива
    {
        foreach (int elem in a)
        {
            Console.Write($"{elem} ");
        }
        Console.WriteLine();
    }
    static void Input(out int[] a)
    // ввод размерности массива и его заполнение
    {
        Console.Write("n= ");
        int n = int.Parse(Console.ReadLine());
        a = new int[n];
        Random rnd = new Random();
        for (int i = 0; i < n; i++)
        {
            a[i] = rnd.Next(-10, 10);
        }
    }
    static int Max(int[] a)
    //нахождение максимального элемента

```

```

{
    int max = a[0];
    foreach (int elem in a)
    {
        if (elem > max)
        {
            max = elem;
        }
    }
    return max;
}

static int IndexMax(int[] a)
    //нахождение индекса максимального элемента в массиве
{
    int indexMax = Array.IndexOf(a, Max(a));
    return indexMax;
}
static void Main(string[] args)
{
    int[] A1;
    int[] B1;
    int[] C1;

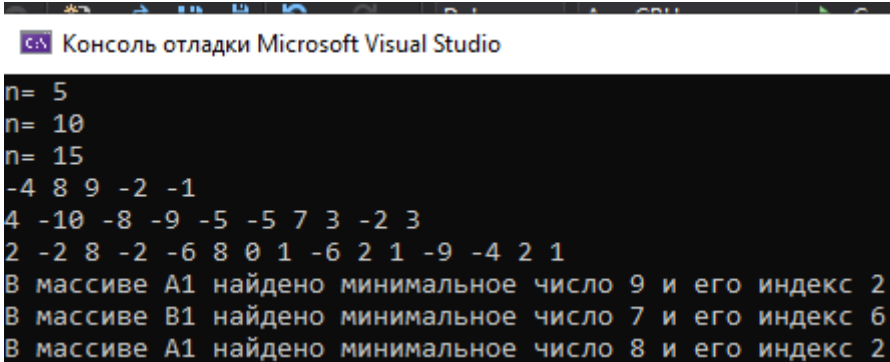
    Input(out A1);
    Input(out B1);
    Input(out C1);

    Print(A1);
    Print(B1);
    Print(C1);

    int IndexMaxA = IndexMax(A1);
    int IndexMaxB = IndexMax(B1);
    int IndexMaxC = IndexMax(C1);

    Console.WriteLine($"В массиве A1 найдено минимальное число {Max(A1)} и его индекс
{IndexMaxA}");
    Console.WriteLine($"В массиве B1 найдено минимальное число {Max(B1)} и его индекс
{IndexMaxB}");
    Console.WriteLine($"В массиве A1 найдено минимальное число {Max(C1)} и его индекс
{IndexMaxC}");
}
}

```



```

n= 5
n= 10
n= 15
-4 8 9 -2 -1
4 -10 -8 -9 -5 -5 7 3 -2 3
2 -2 8 -2 -6 8 0 1 -6 2 1 -9 -4 2 1
В массиве A1 найдено минимальное число 9 и его индекс 2
В массиве B1 найдено минимальное число 7 и его индекс 6
В массиве A1 найдено минимальное число 8 и его индекс 2

```

18. Описать метод - процедуру MinmaxNum(X, n, nmin, nmax), находящую номера минимального и максимального элемента вещественного массива X размера n. Выходные параметры целого типа: nmin (номер

минимального элемента) и pmax(номер максимального элемента). С помощью этого метода найти номера минимальных и максимальных элементов массивов A, B, C размера na, nb, nc соответственно.

```
class Program
{
    static void Print(int[] a)
    // печать массива
    {
        foreach (int elem in a)
        {
            Console.Write($"{elem} ");
        }
        Console.WriteLine();
    }
    static void Input(out int[] a)
    // ввод размерности массива и его заполнение
    {
        Console.Write("n = ");
        int n = int.Parse(Console.ReadLine());
        a = new int[n];
        Random rnd = new Random();
        for (int i = 0; i < n; i++)
        {
            a[i] = rnd.Next(-30, 30);
        }
    }
    static void MinMaxElem(int[] X, ref int min, ref int max, ref int IndexMin, ref int IndexMax)
    //нахождение минимального и максимального элемента и их индекса
    {
        min = X[0];
        max = X[0];

        foreach (int elem in X)
        {
            if (elem < min)
            {
                min = elem;
            }
        }
        foreach (int elem in X)
        {
            if (elem > max)
            {
                max = elem;
            }
        }
        IndexMax = Array.IndexOf(X, max);
        IndexMin = Array.IndexOf(X, min);
    }
    static void Main(string[] args)
    {
        int[] A1;
        int[] B1;
        int[] C1;
        int min = 0;
        int max = 0;
        int IndexMax = 0;
        int IndexMin = 0;
    }
}
```

```

Input(out A1);
Input(out B1);
Input(out C1);

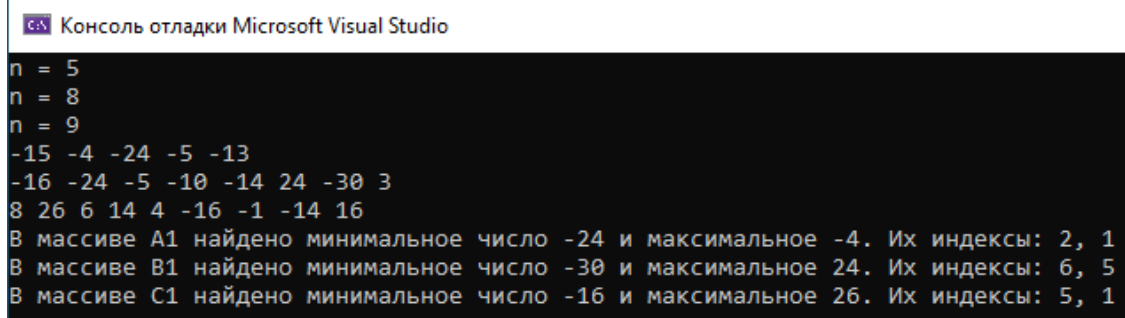
Print(A1);
Print(B1);
Print(C1);

MinMaxElem(A1, ref min, ref max, ref IndexMin, ref IndexMax);
Console.WriteLine($"В массиве A1 найдено минимальное число {min} и максимальное {max}. Их
индексы: {IndexMin}, {IndexMax}");

MinMaxElem(B1, ref min, ref max, ref IndexMin, ref IndexMax);
Console.WriteLine($"В массиве B1 найдено минимальное число {min} и максимальное {max}. Их
индексы: {IndexMin}, {IndexMax}");

MinMaxElem(C1, ref min, ref max, ref IndexMin, ref IndexMax);
Console.WriteLine($"В массиве C1 найдено минимальное число {min} и максимальное {max}. Их
индексы: {IndexMin}, {IndexMax}");
}
}

```



```

cs Консоль отладки Microsoft Visual Studio
n = 5
n = 8
n = 9
-15 -4 -24 -5 -13
-16 -24 -5 -10 -14 24 -30 3
8 26 6 14 4 -16 -1 -14 16
В массиве A1 найдено минимальное число -24 и максимальное -4. Их индексы: 2, 1
В массиве B1 найдено минимальное число -30 и максимальное 24. Их индексы: 6, 5
В массиве C1 найдено минимальное число -16 и максимальное 26. Их индексы: 5, 1

```

Двумерные массивы

1. Описать метод - процедуру ArrayToMatrRow(A, K, M, N, B), формирующую по вещественному массиву A размера K матрицу B размера $M \times N$ (матрица заполняется элементами массива A по строкам). «Лишние» элементы массива игнорируются; если элементов массива недостаточно, то оставшиеся элементы матрицы полагаются равными 0. Двумерный массив B является выходным параметром. С помощью этого метода на основе данного массива A размера K и целых чисел M и N сформировать матрицу B размера $M \times N$.

```

class Program
{
    static void PrintA(double[] A)
    {
        for(int i = 0; i < A.Length; i++)
        {
            Console.Write($"{A[i]:f2} ");
        }
        Console.WriteLine();
    }
}

```

					KKOO.OAXXXX.000	Лист
Изм.	Лист	№ докум.	Подпись	Дата		69

```

    }
    static void Input(out double[] A, int K)
    {
        Random rnd = new Random();
        A = new double[K];
        for (int i = 0; i < K; i++)
        {
            A[i] = rnd.Next(-10, 10);
        }
    }
    static void PrintB(double[,] B, int M, int N)
    {
        for (int j = 0; j < M; j++)
        {
            for (int i = 0; i < N; i++)
            {
                Console.Write($"{B[j, i]:f2} ");
            }
            Console.WriteLine();
        }
    }
    static void ArrayToMatrRow(double[] A, int K, int M, int N, out double[,] B)
    {
        int elem = 0;
        B = new double[M, N];
        for (int j = 0; j < M; j++)
        {
            for (int i = 0; i < N; i++)
            {
                if (elem < K)
                {
                    B[j, i] = A[elem];
                }
                if (elem >= K)
                {
                    B[j, i] = 0;
                }
                elem++;
            }
        }
    }
    static void Main(string[] args)
    {
        Console.Write("Введите размерность массива K: ");
        int K = int.Parse(Console.ReadLine());
        double[] A = new double [K];
        Input(out A, K);
        PrintA(A);
        Console.Write("Введите количество строк M: ");
        int M = int.Parse(Console.ReadLine());
        Console.Write("Введите количество столбцов N: ");
        int N = int.Parse(Console.ReadLine());
        double[,] B = new double[M,N];
        ArrayToMatrRow(A, K, M, N, out B);
        PrintB(B, M, N);
    }
}

```

```

C:\> Консоль отладки Microsoft Visual Studio
Введите размерность массива K: 8
9,00 7,00 4,00 -8,00 -9,00 -1,00 -10,00 9,00
Введите количество строк M: 3
Введите количество столбцов N: 3
9,00 7,00 4,00
-8,00 -9,00 -1,00
-10,00 9,00 0,00
  
```

2. Описать метод - процедуру `ArrayToMatrCol(A, K, M, N, B)`, формирующую по вещественному массиву `A` размера `K` матрицу `B` размера $M \times N$ (матрица заполняется элементами массива `A` по столбцам). «Лишние» элементы массива игнорируются; если элементов массива недостаточно, то оставшиеся элементы матрицы полагаются равными 0. Двумерный массив `B` является выходным параметром. С помощью этого метода на основе данного массива `A` размера `K` и целых чисел `M` и `N` сформировать матрицу `B` размера $M \times N$.

```

class Program
{
    static void PrintA(double[] A)
    {
        for(int i = 0; i < A.Length; i++)
        {
            Console.Write($"{A[i]:f2} ");
        }
        Console.WriteLine();
    }
    static void Input(out double[] A, int K)
    {
        Random rnd = new Random();
        A = new double[K];
        for (int i = 0; i < K; i++)
        {
            A[i] = rnd.Next(-10, 10);
        }
    }
    static void PrintB(double[,] B, int M, int N)
    {
        for (int j = 0; j < M; j++)
        {
            for (int i = 0; i < N; i++)
            {
                Console.Write($"{B[j, i],5:f2} ");
            }
            Console.WriteLine();
        }
    }
    static void ArrayToMatrRow(double[] A, int K, int M, int N, out double[,] B)
    {
        int elem = 0;
        B = new double[M, N];
        for (int i = 0; i < M; i++)
  
```

```

    {
        for (int j = 0; j < N; j++)
        {
            if (elem < K)
            {
                B[j, i] = A[elem];
            }
            if (elem >= K)
            {
                B[j, i] = 0;
            }
            elem++;
        }
    }
}
static void Main(string[] args)
{
    Console.WriteLine("Введите размерность массива K: ");
    int K = int.Parse(Console.ReadLine());
    double[] A = new double [K];
    Input(out A, K);
    PrintA(A);
    Console.WriteLine("Введите количество строк M: ");
    int M = int.Parse(Console.ReadLine());
    Console.WriteLine("Введите количество столбцов N: ");
    int N = int.Parse(Console.ReadLine());
    double[,] B = new double[M,N];
    ArrayToMatrRow(A, K, M, N, out B);
    PrintB(B, M, N);
}
}

```

```

cs Консоль отладки Microsoft Visual Studio
Введите размерность массива K: 8
8,00 -7,00 5,00 -5,00 6,00 2,00 -5,00 -1,00
Введите количество строк M: 4
Введите количество столбцов N: 4
8,00 6,00 0,00 0,00
-7,00 2,00 0,00 0,00
5,00 -5,00 0,00 0,00
-5,00 -1,00 0,00 0,00

```

3. Описать метод - процедуру Chessboard(M, N, A), формирующую по целым положительным числам M и N матрицу A размера $M \times N$, которая содержит числа 0 и 1, расположенные в «шахматном» порядке, причем $A_{1,1} = 0$. Двумерный целочисленный массив A является выходным параметром. С помощью этого метода по данным целым числам M и N сформировать матрицу A размера $M \times N$.

```

class Program
{
    static void PrintA(int[,] A, int M, int N)
    {
        for (int j = 0; j < M; j++)
        {
            for (int i = 0; i < N; i++)

```



```

    {
        Console.Write($"{A[i, j]} ");
    }
    Console.WriteLine();
}
}
static void Chessboard(out int[,] A, int M, int N)
{
    A = new int[M, N];
    for (int j = 0; j < M; j++)
    {
        for (int i = 0; i < N; i++)
        {
            if (j % 2 == 0)
            {
                if (i % 2 == 0)
                {
                    A[j, i] = 0;
                }
                else
                {
                    A[j, i] = 1;
                }
            }
            if (j % 2 != 0)
            {
                if (i % 2 == 0)
                {
                    A[j, i] = 1;
                }
                else
                {
                    A[j, i] = 0;
                }
            }
        }
    }
}
static void Main(string[] args)
{
    Console.Write("Введите количество строк M: ");
    int M = int.Parse(Console.ReadLine());
    Console.Write("Введите количество столбцов N: ");
    int N = int.Parse(Console.ReadLine());
    int[,] A = new int[M, N];
    Chessboard(out A, N, M);
    PrintA(A, M, N);
}
}

```

4. Описать метод - функцию SumRow(A, M, N, K) вещественного типа, вычисляющую сумму элементов вещественной матрицы A размера $M \times N$, расположенных в K-й строке (если $K > M$, то функция возвращает 0). Для данной матрицы A размера $M \times N$ и трех данных K найти SumRow(A, M, N, K).

```
class Program
{
    static void PrintA(double[,] A, int M, int N)
    {
        for (int i = 0; i < M; i++)
        {
            for (int j = 0; j < N; j++)
            {
                Console.Write($"{A[i, j], 5:f2} ");
            }
            Console.WriteLine();
        }
    }
    static void Input(out double[,] A, int M, int N)
    {
        Random rnd = new Random();
        A = new double[M, N];
        for (int i = 0; i < M; i++)
        {
            for (int j = 0; j < N; j++)
            {
                A[i, j] = rnd.Next(-10, 10);
            }
        }
    }
    static double SumRow(double[,] A, int M, int N, int K)
    {
        double sum = 0;
        if (K <= M)
        {
            for (int i = 0; i < N; i++)
            {
                sum = sum + A[K - 1, i];
            }
        }
        if (K > M)
        {
            sum = 0;
        }
        return sum;
    }
    static void Main(string[] args)
    {
        Console.Write("Введите количество строк M: ");
        int M = int.Parse(Console.ReadLine());
        Console.Write("Введите количество столбцов N: ");
        int N = int.Parse(Console.ReadLine());
        double[,] A = new double[M, N];
        Input(out A, M, N);
        PrintA(A, M, N);
        int K = 0;
        for(int i = 0; i < 3; i++)
```

```

{
    Console.Write("Введите номер строки K: ");
    K = int.Parse(Console.ReadLine());
    Console.WriteLine($"Сумма строки {K} равна {SumRow(A, M, N, K)} ");
}
}
}

```

```

Консоль отладки Microsoft Visual Studio
Введите количество строк M: 5
Введите количество столбцов N: 4
0,00 -5,00 3,00 4,00
5,00 4,00 -1,00 5,00
6,00 8,00 5,00 -4,00
8,00 -8,00 5,00 7,00
2,00 9,00 4,00 1,00
Введите номер строки K: 3
Сумма строки 3 равна 15
Введите номер строки K: 8
Сумма строки 8 равна 0
Введите номер строки K: 1
Сумма строки 1 равна 2

```

5. Описать метод - функцию SumCol(A, M, N, K) вещественного типа, вычисляющую сумму элементов вещественной матрицы A размера $M \times N$, расположенных в K-м столбце (если $K > N$, то функция возвращает 0). Для данной матрицы A размера $M \times N$ и трех данных K найти SumCol(A, M, N, K).

```

class Program
{
    static void PrintA(double[,] A, int M, int N)
    {
        for (int i = 0; i < M; i++)
        {
            for (int j = 0; j < N; j++)
            {
                Console.Write($"A[{i}, {j}], 5:f2} ");
            }
            Console.WriteLine();
        }
    }
    static void Input(out double[,] A, int M, int N)
    {
        Random rnd = new Random();
        A = new double[M, N];
        for (int i = 0; i < M; i++)
        {
            for (int j = 0; j < N; j++)
            {
                A[i, j] = rnd.Next(-10, 10);
            }
        }
    }
    static double SumCol(double[,] A, int M, int N, int K)
    {
        double sum = 0;
        if (K <= N)

```

```

    {
        for (int i = 0; i < N; i++)
        {
            sum = sum + A[i, K - 1];
        }
    }
    if (K > N)
    {
        sum = 0;
    }
    return sum;
}
static void Main(string[] args)
{
    Console.Write("Введите количество строк M: ");
    int M = int.Parse(Console.ReadLine());
    Console.Write("Введите количество столбцов N: ");
    int N = int.Parse(Console.ReadLine());
    double[,] A = new double[M, N];
    Input(out A, M, N);
    PrintA(A, M, N);
    int K = 0;
    for(int i = 0; i < 3; i++)
    {
        Console.Write("Введите номер столбца K: ");
        K = int.Parse(Console.ReadLine());
        Console.WriteLine($"Сумма строки {K} равна {SumCol(A, M, N, K)} ");
    }
}
}

```

```

Консоль отладки Microsoft Visual Studio
Введите количество строк M: 3
Введите количество столбцов N: 3
7,00 -7,00 9,00
8,00 -10,00 -1,00
-2,00 5,00 -7,00
Введите номер столбца K: 1
Сумма строки 1 равна 13
Введите номер столбца K: 2
Сумма строки 2 равна -12
Введите номер столбца K: 12
Сумма строки 12 равна 0

```

6. Описать метод - процедуру SwapRow(A, M, N, K1, K2), осуществляющую перемену местами строк вещественной матрицы A размера $M \times N$ с номерами K1 и K2. Матрица A является входным и выходным параметром; если K1 или K2 больше M, то матрица не изменяется. Используя эту метод - метод - процедуру, поменять для данной матрицы A размера $M \times N$ строки с данными номерами K1 и K2.

```

class Program
{
    static void PrintA(double[,] A, int M, int N)
    {
        for (int i = 0; i < M; i++)

```

```

    {
        for (int j = 0; j < N; j++)
        {
            Console.WriteLine($"{A[i, j], 5:f2} ");
        }
        Console.WriteLine();
    }
}
static void Input(out double[,] A, int M, int N)
{
    Random rnd = new Random();
    A = new double[M, N];
    for (int i = 0; i < M; i++)
    {
        for (int j = 0; j < N; j++)
        {
            A[i, j] = rnd.Next(-10, 10);
        }
    }
}
static void SwapRow(double[,] A, int M, int N, int K1, int K2)
{
    double[] B = new double[N];
    for (int i = 0; i < N; i++)
    {
        B[i] = A[K1-1, i];
    }
    for (int i = 0; i < M; i++)
    {
        A[K1 - 1, i] = A[K2 - 1, i];
    }
    for (int i = 0; i < M; i++)
    {
        A[K2 - 1, i] = B[i];
    }
}
static void Main(string[] args)
{
    Console.WriteLine("Введите количество строк M: ");
    int M = int.Parse(Console.ReadLine());
    Console.WriteLine("Введите количество столбцов N: ");
    int N = int.Parse(Console.ReadLine());
    double[,] A = new double[M, N];
    Input(out A, M, N);
    PrintA(A, M, N);
    Console.WriteLine("Введите номер первой строки K1: ");
    int K1 = int.Parse(Console.ReadLine());
    Console.WriteLine("Введите номер второй строки K2: ");
    int K2 = int.Parse(Console.ReadLine());
    SwapRow(A, M, N, K1, K2);
    PrintA(A, M, N);
    Console.ReadKey();
}
}

```

```

Консоль отладки Microsoft Visual Studio
Введите количество строк M: 4
Введите количество столбцов N: 5
-1,00  8,00  5,00  3,00 -10,00
-5,00 -3,00  4,00 -9,00  1,00
-2,00  1,00  8,00 -4,00 -1,00
 5,00  4,00  6,00  5,00  1,00
Введите номер первой строки K1: 1
Введите номер второй строки K2: 3
-2,00  1,00  8,00 -4,00 -10,00
-5,00 -3,00  4,00 -9,00  1,00
-1,00  8,00  5,00  3,00 -1,00
 5,00  4,00  6,00  5,00  1,00

```

7. Описать метод - процедуру SwapCol(A, M, N, K1, K2), осуществляющую перемену местами столбцов вещественной матрицы A размера $M \times N$ с номерами K1 и K2. Матрица A является входным и выходным параметром; если K1 или K2 больше N, то матрица не изменяется. Используя эту метод - метод - процедуру, поменять для данной матрицы A размера $M \times N$ столбцы с данными номерами K1 и K2.

```

class Program
{
    static void PrintA(double[,] A, int M, int N)
    {
        for (int i = 0; i < M; i++)
        {
            for (int j = 0; j < N; j++)
            {
                Console.Write($"{A[i, j], 5:f2} ");
            }

            Console.WriteLine();
        }
    }

    static void Input(out double[,] A, int M, int N)
    {
        Random rnd = new Random();
        A = new double[M, N];
        for (int i = 0; i < M; i++)
        {
            for (int j = 0; j < N; j++)
            {
                A[i, j] = rnd.Next(-10, 10) + rnd.NextDouble(); ;
            }
        }
    }

    static void SwapRow(double[,] A, int M, int N, int K1, int K2)
    {
        double[] B = new double[M];
        for (int i = 0; i < M; i++)
        {
            B[i] = A[i, K1-1 ];
        }
        for (int i = 0; i < M; i++)
    }

```

```

    {
        A[i, K1 - 1] = A[i, K2 - 1];
    }
    for (int i = 0; i < M; i++)
    {
        A[i, K2 - 1] = B[i];
    }
}
static void Main(string[] args)
{
    Console.WriteLine("Введите количество строк M: ");
    int M = int.Parse(Console.ReadLine());
    Console.WriteLine("Введите количество столбцов N: ");
    int N = int.Parse(Console.ReadLine());
    double[,] A = new double[M, N];
    Input(out A, M, N);
    PrintA(A, M, N);
    Console.WriteLine("Введите номер первого столбца K1: ");
    int K1 = int.Parse(Console.ReadLine());
    Console.WriteLine("Введите номер второго столбца K2: ");
    int K2 = int.Parse(Console.ReadLine());
    SwapRow(A, M, N, K1, K2);
    PrintA(A, M, N);
    Console.ReadKey();
}
}

```

```

Консоль отладки Microsoft Visual Studio
Введите количество строк M: 5
Введите количество столбцов N: 5
7,66 -1,34 -6,15 -5,05 8,33
-3,24 9,42 7,68 5,75 7,19
7,75 2,72 -4,55 7,39 -4,07
-3,21 1,69 -6,51 -3,84 0,66
7,05 -3,00 -5,30 0,91 4,90
Введите номер первого столбца K1: 1
Введите номер второго столбца K2: 2
-1,34 7,66 -6,15 -5,05 8,33
9,42 -3,24 7,68 5,75 7,19
2,72 7,75 -4,55 7,39 -4,07
1,69 -3,21 -6,51 -3,84 0,66
-3,00 7,05 -5,30 0,91 4,90

```

8. Описать метод - метод - процедуру Transp(A, M), выполняющую транспонирование (т. е. зеркальное отражение относительно главной диагонали) квадратной вещественной матрицы A порядка M. Матрица A является входным и выходным параметром. Используя эту метод - метод - процедуру, транспонировать данную матрицу A порядка M.

```

class Program
{
    static void Print(double[,] A, int M)
    {
        for (int i = 0; i < M; i++)
        {
            for (int j = 0; j < M; j++)
            {
                Console.WriteLine($"{A[i, j], 5:f2} ");
            }
        }
    }
}

```

```

    }
    Console.WriteLine();
}
}
static void Input(out double[,] A, int M)
{
    Random rnd = new Random();
    A = new double[M, M];
    for (int i = 0; i < M; i++)
    {
        for (int j = 0; j < M; j++)
        {
            A[i, j] = rnd.Next(-10, 10) + rnd.NextDouble();
        }
    }
}
static void PrintNew(double[,] A, int M)
{
    for (int i = 0; i < M; i++)
    {
        for (int j = 0; j < M; j++)
        {
            Console.Write($"{A[i, j], 5:f2} ");
        }
        Console.WriteLine();
    }
}
static void Transp(double[,] A, int M)
{
    for (int i = 0; i < M; i++)
    {
        for (int j = 0; j < M; j++)
        {
            if (i > j)
            {
                A[j, i] = A[i, j];
            }
        }
    }
    for (int i = 0; i < M; i++)
    {
        for (int j = 0; j < M; j++)
        {
            if (i < j)
            {
                A[i, j] = A[j, i];
            }
        }
    }
}
static void Main(string[] args)
{
    Console.Write("Введите размер матрицы: ");
    int M = int.Parse(Console.ReadLine());
    double[,] A;
    Input(out A, M);
    Print(A, M);
    Console.WriteLine("Новая матрица:");
    Transp(A, M);
    PrintNew(A, M);
    Console.ReadKey();
}
}

```



```

C# Консоль отладки Microsoft Visual Studio
Введите размер матрицы: 5
4,90 -3,19 -8,99 3,93 1,97
5,27 -0,12 1,85 -3,61 -7,42
-4,99 2,51 4,01 -0,96 -4,20
4,58 6,05 -6,99 -2,47 -4,05
6,11 -8,37 1,11 9,32 9,64
Новая матрица
4,90 5,27 -4,99 4,58 6,11
-3,19 -0,12 2,51 6,05 -8,37
-8,99 1,85 4,01 -6,99 1,11
3,93 -3,61 -0,96 -2,47 9,32
1,97 -7,42 -4,20 -4,05 9,64

```

Контрольная работа

Тема. Массивы

Решить по одной задаче из каждого задания. По варианту. При выполнении задания максимально использовать методы. Способ заполнения исходного массива произвольный, если не указан в задании.

Задание1.

Заданы массивы A(5), B(7),C(6), состоящие из положительных чисел. Определить в каком массиве, больше чисел меньших 30. Подсчет количества чисел меньших 30 оформить в виде метода. Заполнение массивов и вывод на печать оформить в виде метода.

```
class Program
{
    static void Print(int[] a)
    {
        foreach (int elem in a)
        {
            Console.Write($"{elem} ");
        }
        Console.WriteLine();
    }
    static void Input(out int[] a)
    {
        Console.Write("N = ");
        int n = int.Parse(Console.ReadLine());
        a = new int[n];
        Random rnd = new Random();
        for (int i = 0; i < n; i++)
        {
            a[i] = rnd.Next(0, 100);
        }
    }
    static int CountElem(int[] a)
    {
        int count = 0;
        for (int i = 0; i < a.Length; i++)
        {
            if (a[i] < 30)
            {
                count++;
            }
        }
        return count;
    }
    static void Main(string[] args)
    {
        int[] A;
        int[] B;
        int[] C;
        Input(out A);
        Input(out B);
        Input(out C);
    }
}
```

```


Print(A);
Print(B);
Print(C);
int a_chetcount = CountElem(A);
int b_chetcount = CountElem(B);
int c_chetcount = CountElem(C);

int[] CountArray = new int[3] { a_chetcount, b_chetcount, c_chetcount };

int max = 0;
if (a_chetcount < b_chetcount)
{
    max = b_chetcount;
    if (max < c_chetcount)
    {
        max = c_chetcount;
    }
}
else
{
    max = a_chetcount;
    if (max < c_chetcount)
    {
        max = c_chetcount;
    }
}

for (int i = 0; i < CountArray.Length; i++)
{
    if (max == a_chetcount)
    {
        Console.WriteLine($"В массиве А найдено наибольшее количество чисел, меньших 30:
{max}");
        break;
    }
    if (max == b_chetcount)
    {
        Console.WriteLine($"В массиве В найдено наибольшее количество чисел, меньших 30:
{max}");
        break;
    }
    if (max == c_chetcount)
    {
        Console.WriteLine($"В массиве С найдено наибольшее количество чисел, меньших 30:
{max}");
        break;
    }
}
}

```

 Выбрать Консоль отладки Microsoft Visual Studio

```

N = 5
N = 7
N = 6
87 32 85 0 22
11 18 70 95 37 32 75
37 21 84 59 31 64
В массиве А найдено наибольшее количество чисел, меньших 30: 2

```

Задание 2.

Дана матрица размера $M \times N$ и целое число K ($1 \leq K \leq M$). Найти сумму и произведен

ие элементов K -й строки данной матрицы. M , N и K вводятся с экрана.

```
class Program
{
    static void Print(double[,] A, int M, int N)
    {
        // печать массива на экран
        for (int i = 0; i < M; i++)
        {
            for (int j = 0; j < N; j++)
            {
                if (j == 0)
                {
                    Console.Write($"{A[i, j],10:f2} |");
                }
                else
                {
                    Console.Write($" {A[i, j],10:f2} |");
                }
            }
            Console.WriteLine();
        }
    }
    static void Input(out double[,] A, int M, int N)
    {
        // заполнение массива случайными числами
        Random rnd = new Random();
        A = new double[M, N];
        for (int i = 0; i < M; i++)
        {
            for (int j = 0; j < N; j++)
            {
                A[i, j] = rnd.Next(-30, 30);
            }
        }
    }
    static double SumRow(double[,] A, int M, int N, int K)
    {
        // сложение строки
        double sum = 0;
        if (K <= M)
        {
            for (int i = 0; i < N; i++)
            {
                sum = sum + A[K - 1, i];
            }
        }
        if (K > M)
        {
            sum = 0;
        }
        return sum;
    }

    static double MultiplyRow(double[,] A, int M, int N, int K)
    {

```

```

// умножение строки
double result = 1;
if (K <= M)
{
    for (int i = 0; i < N; i++)
    {
        result = result * A[K - 1, i];
    }
}
if (K > M)
{
    result = 0;
}
return result;
}

static void Main(string[] args)
{
    // ввод размерности
    Console.Write("Введите количество строк M: ");
    int M = int.Parse(Console.ReadLine());
    Console.Write("Введите количество столбцов N: ");
    int N = int.Parse(Console.ReadLine());

    double[,] A = new double[M, N];

    Input(out A, M, N);
    Print(A, M, N);

    Console.Write("Введите номер строки K: "); // ввод номера строки
    int K = int.Parse(Console.ReadLine());

    Console.Clear();
    Console.WriteLine($"Сумма строки {K} равна {SumRow(A, M, N, K)} ");

    Console.WriteLine($"Произведение строки {K} равно {MultiplyRow(A, M, N, K)} ");

    Console.WriteLine();
    Print(A, M, N);
    Console.ReadLine();
}
}

```

Задание 3.

Дан массив 15 целых чисел. Элементы массива заполняются случайно с помощью генератора случайных чисел из диапазона [-5,15]. Найти сумму первых десяти элементов массива.

```
class Program
{
    static void Print(int[] a)
    {
        // печать массива на экран
        foreach (int elem in a)
        {
            Console.Write($"{elem} ");
        }
        Console.WriteLine();
    }
    static void Input(out int[] a)
    {
        // заполнение массива
        Console.Write("n = ");
        int n = int.Parse(Console.ReadLine());
        a = new int[n];
        Random rnd = new Random();
        for (int i = 0; i < n; i++)
        {
            a[i] = rnd.Next(-5, 15);
        }
    }
    static int SumElem(int[] a)
    {
        // подсчет суммы 10 элементов
        int sum = 0;
        for (int i = 0; i <= a.Length; i++)
        {
            if (i < 10)
            {
                sum += a[i];
            }
        }
        return sum;
    }
    static void Main(string[] args)
    {
        int[] A;

        Input(out A);

        Print(A);

        Console.WriteLine($"Сумма первых 10 элементов массива: {SumElem(A)}");
    }
}
```

Выбрать Консоль отладки Microsoft Visual Studio

n = 15

10 3 -4 -5 12 8 1 -3 8 8 10 4 1 14 5

Сумма первых 10 элементов массива: 38

Лабораторная работа №13.3

Тема: работа с массивами.

Цель: научиться использовать массив в виде параметра метода.

Задание 2. Для закрепления теоретического материала по теме сортировка необходимо изучить теоретический материал и выполнить предложенные задания.

Критерии оценки.

Выполнены все задания для одного метода – оценка удовлетворительно.

Выполнены все задания для двух методов – оценка удовлетворительно.

Выполнены все задания для трех методов – оценка удовлетворительно.

Для получения дополнительной оценки предлагается решить задачи из дополнительного раздела.

Постановка задачи. Для решения многих задач необходимо упорядочить данные по определенному признаку. Процесс упорядочения заданного множества объектов по заданному признаку называется сортировкой. Для простоты изложения рассматривается одномерный массив из целых чисел. Суть большинства алгоритмов сортировки от такого упрощения не изменяется. Алгоритмы сортировки отличаются друг от друга степенью эффективности, под которой понимается количество сравнений и количество обменов, произведенных в процессе сортировки. В основном мы будем оценивать эффективность количеством операций сравнения (порядком этого значения). Заметим, что элементы массива можно сортировать:

- по возрастанию — каждый следующий элемент больше предыдущего;
- по неубыванию — каждый следующий элемент не меньше

					ККОО.ОАXXXX.000	Лист
Изм.	Лист	№ докум.	Подпись	Дата		87

предыдущего, то есть больше или равен;

- по убыванию — каждый следующий элемент меньше предыдущего;
- по невозрастанию — каждый следующий элемент не больше

предыдущего, то есть меньше или равен.

Научившись выполнять одну сортировку, изменить ее, чтобы получить другую, не составляет особого труда.

Задание. Ознакомьтесь с алгоритмами сортировки простого выбора, простого обмена, простой вставки.

Запрограммируйте эти методы, стараясь не использовать подсказки. Ниже приведены программные коды, реализующие эти методы.

Комментарии.

Класс предоставляет набор методов и свойств, которые можно использовать для точного измерения затраченного времени.

Метод Start() запускает таймер, который измеряет время выполнения приложения.

Свойство ElapsedMilliseconds это длинное целое число, представляющее общее число миллисекунд, измеренное текущим экземпляром.

Свойство ElapsedTicks это длинное целое число, представляющее общее число тактов таймера, измеренное текущим экземпляром.

Для получения более полной информации о классе Stopwatch и его методах можно воспользоваться ссылкой <https://docs.microsoft.com/ru-ru/dotnet/api/system.diagnostics.stopwatch?view=netcore-2.0>

Сортировка простым выбором

```
class Program
{
    static void Main(string[] args)
    {
        int n = 100;
        int[] a = new int[n];
        Random r = new Random();
        for (int i = 0; i < n; i++) {
            a[i] = r.Next();
            Console.Write(a[i] + " ");
        }

        Stopwatch watch = new Stopwatch();
        watch.Start();

        //System.Threading.Thread.Sleep(10000);

        //метод простого выбора
        for (int i = 0; i < n - i; i++)
        {
            int max = a[0];
            int maxi = 0;
            for (int j = 0; j < n - i; j++)
            {
                if (a[j] > max)
                {
                    max = a[j];
                    maxi = j;
                }
            }
            a[maxi] = a[n - i - 1];
            a[n - i - 1] = max;
        }

        watch.Stop();
        Console.WriteLine();
        for (int i = 0; i < n; i++)
        {
            Console.Write(a[i] + " ");
        }

        double mseconds = watch.ElapsedMilliseconds;
        double seconds = watch.Elapsed.Seconds;
        double tickrate = watch.ElapsedTicks;

        Console.WriteLine("Метод простого выбора");
        Console.WriteLine($"Сортировка массива из {n} элементов");
        Console.WriteLine($"Время выполнения программы в миллисекундах: {mseconds} мс");
        Console.WriteLine($"Время выполнения программы в секундах: {seconds} с");
        Console.WriteLine($"Время выполнения программы в Тиках: {tickrate}");
    }
}
```

Сортировка простым обменом

					ККОО.ОАXXXX.000	Лист
Изм.	Лист	№ докум.	Подпись	Дата		89

```

class Program
{
    static void Main(string[] args)
    {
        int n = 100000;
        int[] a = new int[n];
        Random r = new Random();
        for (int i = 0; i < n; i++) {
            a[i] = r.Next();
            Console.Write(a[i] + " ");
        }

        Stopwatch watch = new Stopwatch();
        watch.Start();

        //System.Threading.Thread.Sleep(10000);

        //метод простого обмена
        for (int i = 0; i < n - i; i++)
        {
            int w;
            for (int j = 0; j < n - i - 1; j++)
            {
                if (a[j] > a[j + 1])
                {
                    w = a[j];
                    a[j] = a[j + 1];
                    a[j + 1] = w;
                }
            }
        }

        watch.Stop();
        Console.WriteLine();
        for (int i = 0; i < n; i++)
        {
            Console.Write(a[i] + " ");
        }

        double mseconds = watch.ElapsedMilliseconds;
        double seconds = watch.Elapsed.Seconds;
        double tickrate = watch.ElapsedTicks;

        Console.WriteLine("Метод простого обмена");
        Console.WriteLine($"Сортировка массива из {n} элементов");
        Console.WriteLine($"Время выполнения программы в миллисекундах: {mseconds} мс");
        Console.WriteLine($"Время выполнения программы в секундах: {seconds} с");
        Console.WriteLine($"Время выполнения программы в Тиках: {tickrate}");

    }
}

```

Сортировка простыми вставками

```
class Program
{
    static void Main(string[] args)
    {
        int n = 100;
        int[] a = new int[n];
        Random r = new Random();
        for (int i = 0; i < n; i++) {
            a[i] = r.Next();
            Console.Write(a[i] + " ");
        }

        Stopwatch watch = new Stopwatch();
        watch.Start();

        //System.Threading.Thread.Sleep(10000);

        //метод простых вставок
        for (int i = 1; i < n; i++)
        {
            int w = a[i];
            int j = i - 1;
            while (j >= 0 && w < a[j])
            {
                a[j + 1] = a[j];
                j--;
            }
            a[j + 1] = w;
        }

        watch.Stop();
        Console.WriteLine();
        for (int i = 0; i < n; i++)
        {
            Console.Write(a[i] + " ");
        }

        double mseconds = watch.ElapsedMilliseconds;
        double seconds = watch.Elapsed.Seconds;
        double tickrate = watch.ElapsedTicks;

        Console.WriteLine("Метод простой вставки");
        Console.WriteLine($"Сортировка массива из {n} элементов");
        Console.WriteLine($"Время выполнения программы в миллисекундах: {mseconds} мс");
        Console.WriteLine($"Время выполнения программы в секундах: {seconds} с");
        Console.WriteLine($"Время выполнения программы в Тиках: {tickrate}");
    }
}
```

Задание 3. Проведите испытание. Задайте количество элементов в массиве 100,1000,10000,100000 и сравните время выполнения алгоритма. Какой алгоритм эффективнее? Для ответа на этот вопрос заполните таблицу (* заменить на полученные данные)

					ККОО.ОАXXXX.000	Лист
Изм.	Лист	№ докум.	Подпись	Дата		91

метод \ n	100	1000	10000	100000
Метод простого выбора	0 мс 0 сек 132 тик	0 мс 0 сек 8253 тик	86 мс 0 сек 860227 тик	7987 мс 7 сек 79879953 тик
Метод простых обменов	0 мс 0 сек 228 тик	1 мс 0 сек 19114 тик	254 мс 0 сек 2543233 тик	25164 мс 25 сек 251641984 тик
Метод простых вставок	0 мс 0 сек 121 тик	0 мс 0 сек 7899 тик	75 мс 0 сек 754580 тик	7565 мс 7 сек 75654789 тик

Задание 4. Как изменятся алгоритмы, если сортировать массив надо по убыванию.

Метод простого выбора – не изменится.

Метод простых обменов – не изменится.

Метод простых вставок – не изменится.

Задание 5. Изменить решения в трех рассмотренных методах так, чтобы осуществлялась сортировка:

- четных элементов массива;
- элементов, записанных на нечетных местах;
- отрицательных элементов массива и т.д.

Лабораторная работа №14

Тема: Основы ООП

Часть 1

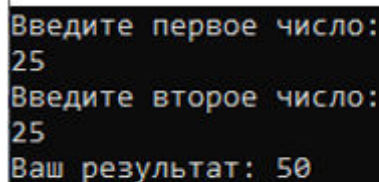
1. Запустите Visual C#.
2. Создайте новый проект, выбрав: шаблон: Консольное приложение, сохраните проект в своей папке.
3. Напишите мини программку, которая складывает два числа введенных с клавиатуры без использования ООП:

```
class Program
{
    static void Main(string[] args)
    {
        int a, b, c;
        Console.WriteLine("Введите первое число: \n>>");
        a = Convert.ToInt16(Console.ReadLine());

        Console.WriteLine("Введите второе число: \n>>");
        b = Convert.ToInt16(Console.ReadLine());

        c = a + b;
        Console.WriteLine($"Ваш результат: {c}");
    }
}
```

4. Запустите программу на выполнение (F5).



```
Введите первое число:
25
Введите второе число:
25
Ваш результат: 50
```

5. Дополните программу комментариями.

```
static void Main(string[] args)
{
    int a, b, c; // объявление переменных
    Console.WriteLine("Введите первое число:"); // вывод сообщения на экран
    // запрос ввода у пользователя с клавиатуры
    // перевод типа данных из string в int16
    a = Convert.ToInt16(Console.ReadLine());

    Console.WriteLine("Введите второе число:"); // вывод сообщения на экран
    // запрос ввода у пользователя с клавиатуры
    // перевод типа данных из string в int16
    b = Convert.ToInt16(Console.ReadLine());

    // сложение двух введенных чисел
    // и занесение полученного результата в переменную
    c = a + b;

    // вывод полученного результата
    Console.WriteLine($"Ваш результат: {c}");
}
```

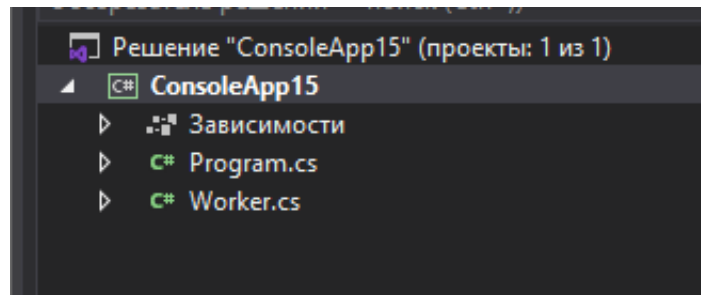
					ККОО.ОАXXXX.000	Лист
Изм.	Лист	№ докум.	Подпись	Дата		93

```

        Console.ReadLine();
    }

```

6. Создайте новый проект, в котором будет отдельный класс, а в основной программе этот класс будет использован.



7. Добавьте в проект новый класс (Проект – Добавить класс) и назовите этот класс Worker. В класс добавьте два общедоступных поля: имя и возраст и одно скрытое поле: вес (обратите внимание, что пространство имен не надо переименовывать):

```

class Worker
{
    public int age = 0;
    public string name;
    private float weight;
}

```

В теле функции Main создайте объект класса Worker:

```

class Program
{
    static void Main(string[] args)
    {
        Worker worker= new Worker();
        worker.age = 31;
        worker.name = "Ахмед";
        Console.WriteLine($"Работник имеет имя {worker.name} и его возраст {worker.age}");
    }
}

```

8. Запустите программу на выполнение. Дополните ее комментариями.

```

class Program
{
    static void Main(string[] args)
    {
        // объявление экземпляра класса и выделение под него места в памяти
        Worker worker= new Worker();

        // запись значения в переменную age
        worker.age = 31;

        // запись значения в переменную name
        worker.name = "Ахмед";
    }
}

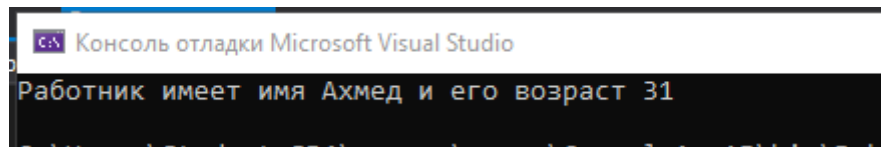
```

```

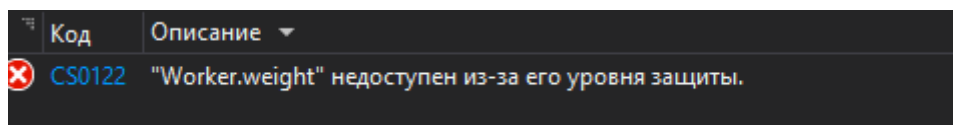
        Console.WriteLine($"Работник имеет имя {worker.name} и его возраст {worker.age}");

        // такая запись не сработает, так как переменная weight
        // имеет уровень защиты private
        //worker.weight = 31;
    }
}

```



9. Попробуйте записать значение в поле weight. Почему данные не записались?



10. Для записи и чтения данных из скрытых полей используют методы. Добавим вовнутрь класса Worker новый метод (действие) который будет отвечать за еду, если человек чего-то там съест, то его вес должен будет увеличиться на количество съеденного.

```

public void eat(float howMuchEat)
{
    weight = weight + howMuchEat;
}

```

11. Если поле вес скрытое, то мы в него не только писать не можем, но и читать тоже не можем. Для чтения данных из скрытого поля необходимо использовать еще один метод:

```

public float getWeight()
{
    return weight;
}

```

12. Почему в последних двух функциях после слова public идут различные слова? Что они обозначают и на что влияют?

Void – метод не возвращает никакого значения

Float – тип данных чисел с плавающей точкой

13. Теперь эти два метода надо использовать в нашей программе. Заставьте рабочего съесть 2, а затем 3 кг пищи, а потом проверьте его вес.

```

worker.Eat(5);
worker.Eat(10);

```

```

float weight_pub;

```

					ККОО.ОАXXXX.000		Лист
							95
Изм.	Лист	№ докум.	Подпись	Дата			

```
weight_pub = worker.GetWeight();
Console.WriteLine($"Его нынешний вес: {weight_pub}");
```

14. Запустите программу на выполнение. Проверьте работоспособность. Добавьте комментарии.

```
class Program
{
    static void Main(string[] args)
    {
        // объявление экземпляра класса и выделение под него места в памяти
        Worker worker= new Worker();

        // запись значения в переменную age
        worker.age = 31;


        // запись значения в переменную name
        worker.name = "Ахмед";
        Console.WriteLine($"Работник имеет имя {worker.name} и его возраст
{worker.age}");

        // вызов методов класса и
        // передача данным методам аргументов
        worker.Eat(5);
        worker.Eat(10);

        // объявление переменной
        float weight_pub;

        // вызов метода получения веса
        weight_pub = worker.GetWeight();

        // вывод веса на экран
        Console.WriteLine($"Его нынешний вес: {weight_pub}");
    }
}
```

 Консоль отладки Microsoft Visual Studio

```
Работник имеет имя Ахмед и его возраст 31
Его нынешний вес: 15
```

15. Усовершенствуйте метод eat таким образом, что если рабочий за раз съедает более чем 10 кг, то его возраст увеличивается на год, а вес увеличивается только на половину съеденного.

```
class Worker
{
    public int age = 0;
    public string name;
    private float weight;

    public void Eat(float howMuchEat)
    {
        if (howMuchEat > 10)
        {
            age++;
            weight = weight + (howMuchEat / 2);
        }
    }
}
```

					ККОО.ОАXXXX.000	Лист
						96
Изм.	Лист	№ докум.	Подпись	Дата		


```

    }
    else
    {
        weight = weight + howMuchEat;
    }
}

public float GetWeight()
{
    return weight;
}
}

```

16. Попросите рабочего съесть 15 кг и посмотрите на результат работы программы.

```

class Program
{
    static void Main(string[] args)
    {
        // объявление экземпляра класса и выделение под него места в памяти
        Worker worker= new Worker();

        // запись значения в переменную age
        worker.age = 31;

        // запись значения в переменную name
        worker.name = "Ахмед";
        Console.WriteLine($"Работник имеет имя {worker.name} и его возраст
{worker.age}");

        // вызов методов класса и
        // передача данным методам аргументов
        worker.Eat(5);
        worker.Eat(10);

        // объявление переменной
        float weight_pub;

        // вызов метода получения веса
        weight_pub = worker.GetWeight();

        // вывод веса на экран
        Console.WriteLine($"Его нынешний вес: {weight_pub}");

        Console.WriteLine($"Мы попросили рабочего съесть 15кг");

        worker.Eat(15);
        Console.WriteLine($"Его нынешний вес: {weight_pub} \nЕго нынешний возраст:
{worker.age}");
    }
}

```

 Консоль отладки Microsoft Visual Studio

```

Работник имеет имя Ахмед и его возраст 31
Его нынешний вес: 15
Мы попросили рабочего съесть 15кг
Его нынешний вес: 22,5
Его нынешний возраст: 32

```

17. Измените программу так, чтобы имя рабочего и его первоначальный возраст вводились с клавиатуры и вносились в соответствующие переменные.

```
class Program
{
    static void Main(string[] args)
    {
        // объявление экземпляра класса и выделение под него места в памяти
        Worker worker= new Worker();

        Console.WriteLine("Введите возраст рабочего:");
        worker.age = Convert.ToInt16(Console.ReadLine()); // запись значения в
        переменную age

        Console.WriteLine("Введите имя рабочего:");
        worker.name = Console.ReadLine(); // запись значения в переменную name
        Console.WriteLine($"Работник имеет имя {worker.name} и его возраст
        {worker.age}");

        // вызов методов класса и
        // передача данным методам аргументов
        worker.Eat(5);
        worker.Eat(10);

        // объявление переменной
        float weight_pub;

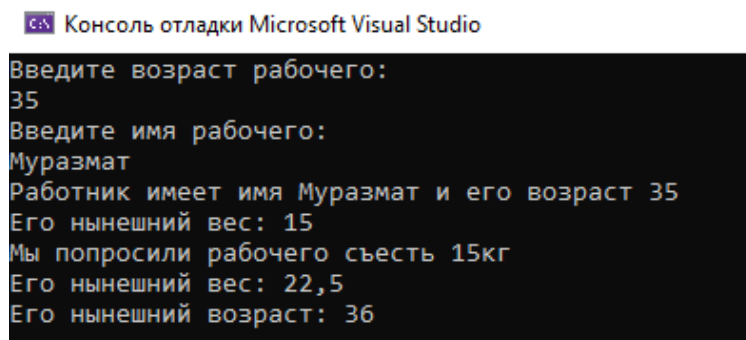
        // вызов метода получения веса
        weight_pub = worker.GetWeight();

        // вывод веса на экран
        Console.WriteLine($"Его нынешний вес: {weight_pub}");

        Console.WriteLine($"Мы попросили рабочего съесть 15кг");

        worker.Eat(15);
        weight_pub = worker.GetWeight();
        Console.WriteLine($"Его нынешний вес: {weight_pub} \nЕго нынешний возраст:
        {worker.age}");
    }
}
```

18. Запустите программу. Проверьте ее работоспособность.



19. Добавьте рабочему еще одно скрытое поле, которое будет отвечать за настройку и будет иметь первоначальное значение равное 10.

					ККОО.ОАXXXX.000	Лист
Изм.	Лист	№ докум.	Подпись	Дата		98

```
private int emotion = 10;
```

20. Добавьте три метода: гулять (метод должен увеличивать настроение на 1), танцевать (метод должен увеличивать настроение на 2) и работать (метод должен уменьшать настроение на 2).

```
public void Walk()
{
    emotion++;
}

public void Dance()
{
    emotion += 2;
}

public void Work()
{
    emotion -= 2;
}
```

21. Дополните основную программу так, чтобы рабочий после еды два раза погулял и три раза потанцевал.

```
Console.WriteLine($"Мы отправили рабочего погулять и потанцевать");
worker.Walk();
worker.Walk();
worker.Dance();
worker.Dance();
worker.Dance();
```

22. Добавьте в класс функцию, которая будет возвращать текущее настроение пользователя.

```
public int GetEmotion()
{
    return emotion;
}
```

23. Добавьте в основную программу метод Работать 9 раз (можно в цикле) и выведите настроение пользователя на экран.

```
class Program
{
    static void Work()
    {
        Worker.Work();
    }
    static void Main(string[] args)
    {
        // объявление экземпляра класса и выделение под него места в памяти
        Worker worker= new Worker();

        Console.WriteLine("Введите возраст рабочего:");
        worker.age = Convert.ToInt16(Console.ReadLine()); // запись значения в
        переменную age
    }
}
```

```

        Console.WriteLine("Введите имя рабочего:");
        worker.name = Console.ReadLine(); // запись значения в переменную name
        Console.WriteLine($"Работник имеет имя {worker.name} и его возраст
{worker.age}");

        // вызов методов класса и
        // передача данным методам аргументов
        worker.Eat(5);
        worker.Eat(10);

        // объявление переменной
        float weight_pub;

        // вызов метода получения веса
        weight_pub = worker.GetWeight();

        // вывод веса на экран
        Console.WriteLine($"Его нынешний вес: {weight_pub}");

        Console.WriteLine($"Мы попросили рабочего съесть 15кг");

        worker.Eat(15);
        weight_pub = worker.GetWeight();
        Console.WriteLine($"Его нынешний вес: {weight_pub} \nЕго нынешний возраст:
{worker.age}");

        Console.WriteLine($"Мы отправили рабочего погулять и потанцевать");
        worker.Walk();
        worker.Walk();
        worker.Dance();
        worker.Dance();
        worker.Dance();
        worker.GetEmotion();

        for (int i = 0; i < 10; i++)
        {
            Work();
        }
        Console.WriteLine($"Настроение работника: {worker.GetEmotion()}");
    }
}

class Worker
{
    public int age = 0;
    public string name;
    private float weight;
    private static int emotion = 10;

    public void Eat(float howMuchEat)
    {
        if (howMuchEat > 10)
        {
            age++;
            weight = weight + (howMuchEat / 2);
        }
        else
        {
            weight = weight + howMuchEat;
        }
    }

    public void Walk()
    {

```

```

        emotion++;
    }

    public void Dance()
    {
        emotion += 2;
    }

    public static void Work()
    {
        emotion -= 2;
    }

    public float GetWeight()
    {
        return weight;
    }

    public int GetEmotion()
    {
        return emotion;
    }
}

```

Консоль отладки Microsoft Visual Studio

```

Введите возраст рабочего:
20
Введите имя рабочего:
Джамшут
Работник имеет имя Джамшут и его возраст 20
Его нынешний вес: 15
Мы попросили рабочего съесть 15кг
Его нынешний вес: 22,5
Его нынешний возраст: 21
Мы отправили рабочего погулять и потанцевать
Настроение работника: -2

```

24. Настроение получилось отрицательным? – ужасно. Измените, метод Работать таким образом, что бы настроение никогда не было меньше нуля (т.е. если настроение было 1 и человек поработал, то оно должно стать не меньше 0).

```

    public static void Work()
    {
        emotion -= 2;
        if (emotion < 0)
        {
            emotion = 0;
        }
    }
}

```

25. Проверьте заново работоспособность программы.

```

Введите возраст рабочего:
25
Введите имя рабочего:
равшан
Работник имеет имя равшан и его возраст 25
Его нынешний вес: 15
Мы попросили рабочего съесть 15кг
Его нынешний вес: 22,5
Его нынешний возраст: 26
Мы отправили рабочего погулять и потанцевать
Настроение работника: 18
Мы отправили рабочего поработать
Настроение работника: 0
    
```

Задание на разработку простейшего класса

Создать класс Point, содержащий следующие члены класса:

1. Поля: int x, y; // координаты точки
2. Методы, позволяющие: вывести координаты точки на экран;

рассчитать расстояние от начала координат до точки;

переместить точку на плоскости на вектор (a, b)

В классе Program продемонстрировать работу класса. Используя методы вывести на экран координаты трех произвольных точек, рассчитать расстояние от начала координат до этих точек, переместить все точки на вектор (a, b). Координаты точек и значения вектора (a, b) вводить с экрана.

```

class Program
{
    static void Main(string[] args)
    {
        //вызов класса
        Point point_core = new Point();
        //присваивание значения переменных из класса
        for(int i = 0; i < 3; i++)
        {
            Console.WriteLine("Введите значение x:");
            point_core.x = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Введите значение y:");
            point_core.y = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Введите значение a:");
            point_core.a = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Введите значение b:");
            point_core.b = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine($"Координаты точки: ({point_core.PrintDots(point_core.x,
point_core.y)})");
            Console.WriteLine($"Расстояние от начала координат:
{point_core.S(point_core.x, point_core.y)}");
        }
    }
}
    
```

```

        Console.WriteLine($"Координаты точек векторов:
({point_core.PrintDots(point_core.a, point_core.b)})");
        Console.WriteLine($"Перемещаем точки x,y на вектор a,b...");
        point_core.Vector_Move(point_core.a, point_core.b);
        Console.WriteLine($"Координаты точек стали:
({point_core.PrintDots(point_core.x, point_core.y)})");
    }
    Console.ReadLine();
}
}
class Point
{
    public int x, y;
    public int a, b;

    public string PrintDots(int x, int y)
    {
        return $"{x}; {y}";
    }
    public double S(int x, int y)
    {
        double rast;
        rast = Math.Sqrt(Math.Pow(x, 2) + Math.Pow(y, 2));
        return rast;
    }

    public void Vector_Move(int a, int b)
    {
        this.x += a;
        this.y += b;
    }
}

```

```

Введите значение x: 25
Введите значение y: 25
Введите значение a: 9
Введите значение b: 7
Координаты точки: (25; 25)
Расстояние от начала координат: 35.35533905932738
Координаты точек векторов: (9; 7)
Перемещаем точки x,y на вектор a,b...
Координаты точек стали: (34; 32)

```

Лабораторная работа № 15

Динамические структуры данных

Цель работы: освоить и закрепить приемы работы с основными динамическими структурами данных.

Списки.

Теоретические сведения.

Списком называется упорядоченное множество, состоящее из переменного числа элементов, к которым применимы операции включения и исключения. Список, отражающий отношения соседства между элементами, называется линейным. Если ограничения на длину списка не установлены, то список представляется в памяти в виде связной структуры. Линейные связные списки являются простейшими динамическими структурами данных.

1.1 Машинное представление связных линейных списков

На рисунке 1 приведена структура однонаправленного списка, где поле INFO - информационное поле, данные, NEXT - указатель на следующий элемент списка. Каждый список должен иметь особый элемент, называемый началом списка или головой списка, который обычно по формату отличен от остальных элементов. В поле указателя последнего элемента списка находится специальный признак NULL, свидетельствующий о конце списка. Обработка однонаправленного списка не всегда удобна, так как отсутствует возможность продвижения по списку в сторону противоположной направлению имеющимся связям. Такую возможность обеспечивает двунаправленный список, каждый элемент которого содержит два указателя: на следующий и предыдущий элементы списка. Структура линейного двунаправленного списка приведена на рисунке 2, где поле NEXT - указатель на следующий элемент, поле PREV - указатель на предыдущий элемент. В крайних элементах соответствующие указатели должны содержать NULL. Для удобства обработки списка добавляют еще один особый элемент - указатель конца списка. Наличие двух указателей в каждом элементе усложняет список и приводит к дополнительным затратам памяти, но в то же

					KKOO.OAXXXX.000	Лист
						104
Изм.	Лист	№ докум.	Подпись	Дата		

время обеспечивает более эффективное выполнение некоторых операций над списком (перемещение, добавление и т. п.).

Ниже приведен проект ListL1 программной реализации однонаправленного списка на языке C#.

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

namespace Spiski1
{
    class Program
    {
        class ElementL1List //Класс ElementL1List - отдельный элемент (узел) Л1-списка
        {
            public int info; //информационная часть
            public ElementL1List next; //указательная часть
        };
        class L1
        {
            ElementL1List head; //начало (голова) списка
            //Добавить элемент в начало списка
            public void ins_begin(int value)
            {
                //создать узел списка и задать ему значения
                ElementL1List elem = new ElementL1List();
                elem.info = value;
                elem.next = head; //следующий элемент после созданного - текущая "голова" списка
                head = elem; //сделать новый элемент "головой"
            }
            //Вставить элемент в заданную позицию
            public void insert_node(int value, int pos)
            { //проверить значение pos на соответствие размеру списка
                if (pos < 2)
                    ins_begin(value);
                else
                {
                    if (pos > number_node())
                        pos = number_node() + 1;
                    ElementL1List elem = head; //elem-временная переменная (то же самое что и head)
                    //перемещаем elem до вставляемой позиции
                    for (int i = 0; i < pos - 2; i++)
                        elem = elem.next;
                    //после цикла elem = элемент списка, после которого надо добавить новый элемент
                    ElementL1List v = new ElementL1List(); //v-новый элемент, вставляемый в список
                    v.info = value;
                    v.next = elem.next;
                    elem.next = v; //следующим элементом за elem будет v
                }
            }
            //Подсчитать количество элементов списка
            public int number_node()
            {
                int k = 0; //счетчик элементов списка
                ElementL1List elem = head;
                //пока не достигнут конец списка
                while (elem != null)
```

					ККОО.ОАXXXX.000	Лист
						105
Изм.	Лист	№ докум.	Подпись	Дата		

```

    {
        k++;
        elem = elem.next; //переходить к следующему элементу
    }
    return k;
}
//Получить значение элемента в value из заданной позиции pos.
//Если элемент по позиции pos существует, функция вернет true, не существует - false
public bool out_node(int pos, out int value)
{
    //проверка на допустимость позиции
    if (pos > number_node() || pos < 1 || number_node() == 0)
    {
        value = 0;
        return false;
    }
    else
    {
        ElementL1List elem = head;
        //перемещение от "головы" списка до элемента по позиции pos
        for (int i = 0; i < pos - 1; i++)
            elem = elem.next;
        value = elem.info; //записать запрошенное значение
        return true;
    }
}
//Удалить элемент заданной позицией pos из списка.
//Если элемент по позиции pos существует, функция вернет true, не существует - false
public bool delete_node(int pos)
{
    //проверка на допустимость позиции
    if (pos > number_node() || pos < 1 || number_node() == 0)
        return false;
    if (pos == 1)
    {
        //переместить "голову" списка на следующий элемент
        //((прежняя "голова" будет потеряна, т. е. удалена)
        head = head.next;
    }
    else
    {
        ElementL1List elem = head;
        //перемещение от "головы" списка до элемента по позиции pos-1
        for (int i = 0; i < pos - 2; i++)
            elem = elem.next;
        //связать элемент предшествующий позиции pos с элементом - следующим за pos
        elem.next = elem.next.next;
    }
    return true;
}
}
static void show(L1 list)
{
    int value;
    for (int i = 1; i <= list.number_node(); i++)
    {
        list.out_node(i, out value);
        Console.WriteLine(value);
    }
}
static void Main(string[] args)
{
    int value, position, code;
    L1 list = new L1(); //Л1-список

```

```

do
{
    //Вывод меню работы с Л1-списком
    Console.WriteLine(" Работа с Л1-списком:");
    Console.WriteLine("1. Добавить элемент в начало списка");
    Console.WriteLine("2. Добавить элемент в заданную позицию списка (нумерация с 1)");
    Console.WriteLine("3. Удалить элемент из заданной позиции списка (нумерация с 1)");
    Console.WriteLine("4. Вывести значение элемента списка по его номеру");
    Console.WriteLine("5. Вывести весь список на экран");
    Console.WriteLine("0. Завершить работу");
    Console.WriteLine("\tКоличество элементов в Л1-списке = {0}", list.number_node());
    Console.Write(" Ваш выбор: ");
    try { code = Convert.ToInt32(Console.ReadLine()); }
    catch { code = 9; }
    switch (code) //разбор вариантов действий
    {
        case 1:
            Console.Write("Введите значение добавляемого элемента: ");
            value = Convert.ToInt32(Console.ReadLine());
            list.ins_begin(value);
            break;
        case 2:
            Console.Write("Введите значение позиции: ");
            positon = Convert.ToInt32(Console.ReadLine());
            Console.Write("Введите значение нового элемента: ");
            value = Convert.ToInt32(Console.ReadLine());
            list.insert_node(value, positon);
            break;
        case 3:
            Console.Write("Введите значение позиции: ");
            positon = Convert.ToInt32(Console.ReadLine());
            if (list.delete_node(positon))
                Console.WriteLine("Элемент по позиции {0} удален.", positon);
            break;
        case 4:
            Console.Write("Введите значение позиции: ");
            positon = Convert.ToInt32(Console.ReadLine());
            if (list.out_node(positon, out value))
                Console.WriteLine("Элемент по позиции {0} равен {1}", positon, value);
            else
                Console.WriteLine("Элемента по позиции {0} не существует", positon);
            break;
        case 5:
            show(list);
            break;
        case 0: break;
        default:
            Console.WriteLine("Неверный выбор пункта меню");
            break;
    }
    Console.WriteLine();
} while (code != 0);
}
}

```

1.2 Реализация линейных списков в библиотеке System.Collection

В связи с тем, что линейные списки являются одними из самых распространенных структур данных, используемыми программистами,

					ККОО.ОАXXXX.000	Лист
Изм.	Лист	№ докум.	Подпись	Дата		107

большинство программных библиотек содержат их реализацию. В среде Microsoft Visual Studio, в частности, доступны следующие классы-списки:

- классы ArrayList, SortedList в пространстве имен System. Collections библиотеки. NET

- классы-шаблоны List, LinkedList в пространстве имен System. Collections. Generic библиотеки .NET

Ниже приведен проект DemoArrayList на языке C# использующий класс ArrayList из библиотеки. NET.

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

namespace Spisok2
{
    class Program
    {
        //Вспомогательная функция - вывод элементов списка на экран
        static void show(ArrayList list)
        {
            for (int i = 0; i < list.Count; i++)
                Console.WriteLine(list[i]);
        }

        static void Main(string[] args)
        {
            int value, positon, code;
            ArrayList arlist = new ArrayList(); //Л1-список
            do
            {
                //вывод меню работы с Л1-списком
                Console.WriteLine(" Работа с Л1-списком, используя класс ArrayList:");
                Console.WriteLine("1. Добавить элемент в начало списка"); Console.WriteLine("2. Добавить элемент в заданную позицию списка (нумерация с 1)");
                Console.WriteLine("3. Удалить элемент из заданной позиции списка (нумерация с 1)");
                Console.WriteLine("4. Вывести значение элемента списка по его номеру");
                Console.WriteLine("5. Вывести весь список на экран");
                Console.WriteLine("6. Вставить элемент в конец списка"); Console.WriteLine("7. Проверить список на пустоту "); Console.WriteLine("8. Сортировать элементы списка по возрастанию");
                Console.WriteLine("0. Завершить работу");
                Console.WriteLine("\tКоличество элементов в Л1-списке = {0}", arlist.Count);
                Console.WriteLine("\tКоличество мест для элементов в Л1-списке = {0}", arlist.Capacity);
                Console.WriteLine(" Ваш выбор: ");
                try { code = Convert.ToInt32(Console.ReadLine()); }
                catch { code = 9; }
                switch (code) //разбор вариантов действий
                {
                    case 1:
                        Console.WriteLine("Введите значение добавляемого элемента: ");
                        value = Convert.ToInt32(Console.ReadLine());
                        arlist.Insert(0, value);
```

```

        break;
    case 2:
        Console.WriteLine("Введите значение позиции: ");
        positon = Convert.ToInt32(Console.ReadLine());
        Console.WriteLine("Введите значение добавляемого элемента: ");
        value = Convert.ToInt32(Console.ReadLine());
        arlist.Insert(positon - 1, value);
        break;
    case 3:
        Console.WriteLine("Введите значение позиции: ");
        positon = Convert.ToInt32(Console.ReadLine());
        try
        {
            arlist.RemoveAt(positon - 1);
            Console.WriteLine("Элемент по позиции {0} удален.", positon);
        }
        catch { }
        break;
    case 4:
        Console.WriteLine("Введите значение позиции: ");
        positon = Convert.ToInt32(Console.ReadLine());
        try
        {
            Console.WriteLine("Элемент по позиции {0} равен {1}", positon, arlist[positon - 1]);
        }
        catch (ArgumentOutOfRangeException)
        {
            Console.WriteLine("Элемента по позиции {0} не существует", positon);
        }
        break;
    case 5:
        show(arlist);
        break;
    case 6:
        Console.WriteLine("Введите значение добавляемого элемента: ");
        value = Convert.ToInt32(Console.ReadLine());
        arlist.Insert(arlist.Count, value);
        break;
    case 7:
        if (arlist.Count == 0)
            Console.WriteLine("Список пуст");
        else
            Console.WriteLine("Список не пуст");
        break;
    case 8:
        arlist.Sort();
        show(arlist);
        break;
    case 0: break;
    default:
        Console.WriteLine("Неверный выбор меню");
        break;
    }
    Console.WriteLine();
}
while (code != 0); }

```

Задание 1. Разобраться с примерами использования линейного однонаправленного списка, приведенными в проектах: ListL1 (пример

реализации однонаправленного списка) и DemoArratList (использование класса ArrayList из библиотеки. NET).

Стек

Стек - такой последовательный список с переменной длиной, где включение и исключение элементов выполняются только с одной стороны списка, называемого вершиной стека. Применяются и другие названия стека - магазин и очередь, функционирующая по принципу LIFO (Last In – First Out – «последним пришел - первым исключается»). Примеры стека в повседневной жизни: пистолетный патронный магазин, тупиковый железнодорожный разъезд для сортировки вагонов.

1.2 Программная реализация стека

Программно стек можно реализовать двумя основными способами:

1) на базе массива. При этом способе количество элементов ограничено размером массива. Дно стека соответствует элементу массива с индексом 0, а вершина стека – значению массива с индексом равному <количество элементов в стеке – 1>.

2) на базе связного списка. Количество элементов ограничено размером свободной оперативной памяти. Каждый элемент связного списка содержит два поля – информационное (содержит значение элемента) и адресное (содержит адрес следующего элемента стека). Поле адреса последнего элемента стека содержит значение NULL.

```
/** Реализация стека на базе одномерного массива */
class Stack
{
    const int SIZE=10; //максимальное количество элементов в стеке (массиве)
    int[] mas=new int [SIZE]; // массив для элементов стека
    int count; //количество элементов хранящихся в массиве
    //Конструктор класса
    public Stack()
    {
        mas = new int[SIZE]; //массив для хранения элементов стека
        InitStack();
    }
    //Конструктор класса с параметром – размером стека
    public Stack(int size)
    {
        mas = new int[size]; //массив для хранения элементов стека
        InitStack();
    }
}
```

```

//Инициализация стека
public void InitStack()
{ count=0; } //обнуление количества элементов стека
//Добавление элемента Elem в стек
public bool PushStack(int elem)
{
if(count < SIZE) //если есть свободное место в массиве
{
mas[count]=elem; //добавление в массив нового элемента
count++; //увеличить количество элементов стека
return true;
}
return false;
}
//Извлечение элемента из стека
public int PopStack()
{
if (count >0) //если есть элементы в стеке
{
count--; //уменьшить количество элементов стека
return mas[count]; //вернуть вершину стека
}
return -1;
}
//Проверка на пустоту
public bool IsEmptyStack()
{
if(count==0) //если количество элементов = 0
return true; //то, вернуть - истина (стек пуст)
else
return false; //иначе, вернуть - ложь (стек не пуст)
}
Демонстрация работы с методами класса Stack
//Главная функция консольного приложения
static void Main(string[] args)
{
int value, code;
Stack st = new Stack(); //объект-стек
do {
//вывод меню работы со стеком
Console.WriteLine(" Работа со стеком:");
Console.WriteLine("1. Добавить элемент в стек");
Console.WriteLine("2. Удалить элемент из стека");
Console.WriteLine("3. Проверка на пустоту");
Console.WriteLine("0. Завершить работу");
Console.WriteLine(" Ваш выбор: ");
try { code = Convert.ToInt32(Console.ReadLine()); }
catch { code = 9; }
switch (code) //разбор вариантов действий
{
case 1: Console.WriteLine("Введите значение добавляемого элемента: ");
value = Convert.ToInt32(Console.ReadLine());
st.PushStack(value);
break;
case 2:
if (!st.IsEmptyStack())
Console.WriteLine("Элемент {0} из вершины стека удален.", st.PopStack ());
else
Console.WriteLine("Стек пуст.");
break;
case 3:
if (st.IsEmptyStack())

```

					ККОО.ОАXXXX.000	Лист
Изм.	Лист	№ докум.	Подпись	Дата		111

```

Console. WriteLine("Стек пуст.");
else
Console. WriteLine("Стек НЕ пуст.");
break;
case 0: break;
default: Console. WriteLine("Неверный выбор меню");
break;
}
Console. WriteLine();
} while (code!= 0);
}

```

Реализация стека использующий классы-шаблоны Stack<T> из библиотеки. NET.

```

//Подключение пространств имен
using System;
using System. Collections. Generic; //пространство для использования классов Stack и Queue
namespace DemoStackAndQueue
{
class Program
{
//Работа со стеком целых чисел
static void DemoStack()
{
int value, code;
Stack<int> st = new Stack<int>(); //стек целых чисел
do {
//вывод меню работы со стеком
Console. WriteLine(" Работа со стеком целых чисел:");
Console. WriteLine("1. Добавить элемент в стек");
Console. WriteLine("2. Удалить элемент из стека");

```

```

Console. WriteLine("3. Прочитать элемент из вершины стека (без удаления)");
Console. WriteLine("4. Очистить весь стек");
Console. WriteLine("5. Проверка на пустоту");
Console. WriteLine("0. Завершить работу");
Console. WriteLine("\tВ стеке находится {0} элементов", st. Count);
Console. Write(" Ваш выбор: ");
try { code = Convert. ToInt32(Console. ReadLine()); }
catch { code = 9; }
switch (code) //разбор вариантов действий
{
case 1: Console. Write("Введите значение добавляемого элемента: ");
value = Convert. ToInt32(Console. ReadLine());
st. Push(value);

```

					ККОО.ОАXXXX.000	Лист
Изм.	Лист	№ докум.	Подпись	Дата		112


```

break;
case 2:
if (st. Count > 0)
Console. WriteLine("Элемент {0} из вершины стека удален", st. Pop());
else
Console. WriteLine("Стек пуст");
break;
case 3:
if (st. Count > 0)
Console. WriteLine("Элемент в вершине стека равен = {0}", st. Peek());
else
Console. WriteLine("Стек пуст");
break;
case 4: st. Clear();
Console. WriteLine("Стек очищен");
break;
case 5:
if (st. Count == 0)
Console. WriteLine("Стек пуст");
else
Console. WriteLine("Стек НЕ пуст");
break;
case 0: break;
default: Console. WriteLine("Неверный выбор меню");
break;
}
Console. WriteLine();
} while (code!= 0);
}

```

Задание 1. Разобраться с примерами использования стека, приведенными в примерах.

Очередь

Очередью FIFO (First In - First Out – «первым пришел - первым исключается») называется такой последовательный список с переменной длиной, в котором включение элементов выполняется только с одной стороны списка (называемой концом или хвостом очереди), а исключение - с

					ККОО.ОАXXXX.000	Лист
Изм.	Лист	№ докум.	Подпись	Дата		113

другой стороны (начало или голова очереди). Очереди к прилавкам и к кассам являются типичным бытовым примером очереди FIFO.

Программная реализация очереди

Программно очередь можно реализовать тремя основными способами:

1) на базе массива. При этом способе количество элементов ограничено размером массива. Начало очереди соответствует элементу массива с индексом 0, а конец очереди – значению массива с индексом равному $\langle \text{количество элементов в очереди} - 1 \rangle$.

2) на базе циклического массива. Этот способ похож на предыдущий. Отличие в том, что, если при добавлении элемента в конец очереди достигается верхняя граница массива, то производится попытка добавить элемент в начало массива (если оно не занято). При этом массив как бы сворачивается в кольцо (считают что за последним элементом массива следует первый элемент). Аналогично происходит и с извлечением элемента из очереди.

3) на базе связного списка. Способ аналогичен представлению стека на базе связного списка. Начало очереди соответствует первому элементу связного списка, конец очереди – последнему элементу списка (возможно и представление наоборот: конец очереди соответствует началу списка, а начало очереди – концу списка).

Рекомендуемый состав методов класса «Очередь»

```
// тип элементов, хранимых в очереди – MyType
void InitQueue(); // инициализировать очередь
void PushQueue (MyType Elem); // добавить в конец очереди
MyType PopQueue (); // прочитать с удалением из начала очереди
MyType TopQueue (); // прочитать без удаления из начала очереди
bool IsEmptyQueue (); // проверить на пустоту
bool IsFullQueue (); // проверить на переполнение
```

Тип `MyType` описывает тип значений элементов очереди. Для обеспечения возможности хранения в очереди значений выбранного типов достаточно изменить `MyType` на выбранный тип.

Метод `InitQueue` выполняет инициализацию очереди; метод должен быть выполнен перед началом работы с очередью.

Метод PushQueue используется для записи в конец очереди нового значения.

Метод PopQueue обеспечивает получение значения, записанного в очередь первым (из начала очереди); выполнение операции выполняется с удалением возвращаемого значения из очереди.

Метод TopQueue также используется для получения значения из начала очереди; в отличие от предыдущей функции, возвращаемое значение из очереди не удаляется.

Метод IsEmptyQueue проверяет очередь на пустоту, возвращает значение true в случае, когда в очереди элементов нет.

Метод IsFullQueue проверяет очередь на полное заполнение; возвращает значение true когда в очереди нет свободного места для нового элемента.

Ниже приведен проект QueueOnList программной реализации очереди на базе линейного списка на языке C#.

```
//Тип элементов хранимых в очереди
class MyClass
{
    public string str; //строковое поле
    public int value; //целочисленное поле
}
/** Реализация очереди на базе линейного списка
// (начало списка = начало очереди) */
class QueueOnList
{
    List<MyClass> list; //линейный список элементов типа MyClass
    public int Count //свойство - количество элементов в очереди (только чтение)
    {
        get { return list.Count; }
    }
    //Конструктор класса-очередь
    public QueueOnList()
    {
        list= new List<MyClass>(); //список для хранения элементов очереди
    }
    //Добавление элемента Elem в очередь
    public bool PushQueue(MyClass Elem)
    {
        list.Add(Elem); //вставить элемент в конец списка
        return true;
    }
    //Извлечение элемента из стека
    public MyClass PopQueue()
    {
        MyClass temp=new MyClass();
        if (list.Count>0) //если есть элементы в очереди
```

					ККОО.ОАXXXX.000	Лист
						115
Изм.	Лист	№ докум.	Подпись	Дата		

```

{
temp=list[0]; //сохранить элемент в начале списка (начале очереди)
list. RemoveAt(0); // удалить начало очереди
}

return temp; //вернуть элемент из начала очереди
}

```

Реализация очереди в различных библиотеках

В Microsoft Visual Studio 2005 доступны следующие классы, реализующие очередь:

- класс Queue в пространстве имен System. Collections библиотеки. NET
- класс-шаблон Queue<T> в пространстве имен System. Collections.

Generic библиотеки. NET

Ниже приведен метод DemoQueue на языке C#, использующий класс Queue <T> из библиотеки. NET.

```

//Работа с очередью строк
static void DemoQueue()
{
int code;
string value;
Queue<string> q = new Queue<string>(); // очередь строк
do {
//вывод меню работы с очередью
Console. WriteLine(" Работа с очередью строк:");
Console. WriteLine("1. Добавить элемент в конец очереди");
Console. WriteLine("2. Удалить элемент из начала очереди");
Console. WriteLine("3. Прочитать элемент из начала очереди(без удаления)");
Console. WriteLine("4. Очистить всю очередь");
Console. WriteLine("5. Проверка на пустоту");
Console. WriteLine("0. Завершить работу");
Console. WriteLine("\tВ очереди находится {0} элементов", q. Count);
Console. Write(" Ваш выбор: ");
try { code = Convert. ToInt32(Console. ReadLine()); }
catch { code = 9; }
switch (code) //разбор вариантов действий
{
case 1: Console. Write("Введите значение добавляемого элемента: ");
value = Console. ReadLine();
q. Enqueue (value);
break;
case 2:
if (q. Count > 0)
Console. WriteLine("Элемент {0} из начала очереди удален", q. Dequeue());
else
Console. WriteLine("Очередь пуста");
break;
case 3:
if (q. Count > 0)
Console. WriteLine("Элемент в начале очереди равен = {0}", q. Peek());
else

```

					ККОО.ОАXXXX.000	Лист
						116
Изм.	Лист	№ докум.	Подпись	Дата		

```

Console. WriteLine("Очередь пуста");
break;
case 4: q. Clear();
Console. WriteLine("Очередь очищена");

break;
case 5:
if (q. Count == 0)
Console. WriteLine("Очередь пуста");
else
Console. WriteLine("Очередь НЕ пуста");
break;
case 0: break;
default: Console. WriteLine("Неверный выбор меню");
break;
}
Console. WriteLine();
} while (code!= 0);
}
//Главная функция консольного приложения
static void Main(string[] args)
{.....}
}
}

```

					ККОО.ОАXXXX.000	Лист
						117
Изм.	Лист	№ докум.	Подпись	Дата		

Итоговая контрольная работа

Работа с двумя и тремя массивами

10. Даны два массива из 20 однозначных чисел. В первом из них записано количество мячей, забитых футбольной командой в игре, во втором — количество пропущенных мячей в этой же игре.

а) Для каждой проведенной игры напечатать словесный результат: "выигрыш", "ничья" или "проигрыш".

б) Определить количество выигрышей данной команды.

в) Определить количество выигрышей и количество проигрышей данной команды.

г) Определить количество выигрышей, количество ничьих и количество проигрышей данной команды.

д) Определить, в скольких играх разность забитых и пропущенных мячей была большей или равной трем.

е) Общее число очков, набранных командой (за выигрыш дается 3 очка, за ничью — 1, за проигрыш — 0).

```
class Program
{
    static void PrintResultOfGame(int[] a)
    // Задание А
    {
        // перебор массива с результатами подсчитанных забитых и пропущенных
        foreach (int games in a)
        {
            // если разница больше 0 - выигрыш
            if (games > 0)
            {
                Console.WriteLine("Выигрыш");
            }
            else
            {
                // если разница меньше 0 - проигрыш
                if (games < 0)
                {
                    Console.WriteLine("Проигрыш");
                }
                // если разница равна 0 - ничья
                else
                {
                    Console.WriteLine("Ничья");
                }
            }
        }
    }
}
```

```

static void InputArr(out int[] a, out int[] b, out int[] c)
// заполнение массива
{
    int n = 20;
    a = new int[n];
    b = new int[n];
    c = new int[n];
    Random rnd = new Random();
    for (int i = 0; i < n; i++)
    {
        a[i] = rnd.Next(0, 12);
        b[i] = rnd.Next(0, 12);
        c[i] = a[i] - b[i];
    }
}

static int CountWin(int [] a)
// Задание Б
{
    int count = 0;
    // перебор массива с результатами подсчитанных забитых и пропущенных
    foreach (int games in a)
    {
        // если разница больше 0 - выигрыш
        if (games > 0)
        {
            count++;
        }
    }
    return count;
}

static int CountLose(int[] a)
// Задание В
{
    int count = 0;
    // перебор массива с результатами подсчитанных забитых и пропущенных
    foreach (int games in a)
    {
        // если разница меньше 0 - проигрыш
        if (games < 0)
        {
            count++;
        }
    }
    return count;
}

static int CountDraw(int[] a)
// Задание Г
{
    int count = 0;
    // перебор массива с результатами подсчитанных забитых и пропущенных
    foreach (int games in a)
    {
        // если разница меньше 0 - проигрыш
        if (games == 0)
        {
            count++;
        }
    }
}

```

```

        return count;
    }

    static int CountGame3(int[] a)
    // Задание Г
    {
        int count = 0;
        // перебор массива с результатами подсчитанных забитых и пропущенных
        foreach (int games in a)
        {
            // если разница меньше 0 - проигрыш
            if (Math.Abs(games) >= 3)
            {
                count++;
            }
        }
        return count;
    }

    static void Main(string[] args)
    {
        int[] goals; // забитые мячи
        int[] fails; // пропущенные
        int[] resultOfGames; // разница между забитыми и пропущенными
        int count_win = 0; // выигрыши
        int count_lose = 0; // проигрыши
        int count_draw = 0; // ничьи
        int count_game3 = 0; // разность больше или равно 3
        int score = 0; // количество очков

        // заполнение массива
        InputArr(out goals, out fails, out resultOfGames);

        for (int i = 0; i < 20; i++)
            // вывод всех игр на экран
            {
                Console.WriteLine($"Игра {i + 1} прошла со счетом {goals[i]} :
{fails[i]}");
            }
        Console.WriteLine();

        // Задание А
        // вывод результатов игр
        PrintResultOfGame(resultOfGames);

        // Задание Б
        // подсчет количества выигрышей команды
        count_win = CountWin(resultOfGames);
        Console.WriteLine($"Количество выигрышей данной команды: {count_win}");

        // Задание В
        // подсчет количества выигрышей и проигрышей команды
        count_lose = CountLose(resultOfGames);
        Console.WriteLine($"Количество выигрышей данной команды: {count_win}");
        Console.WriteLine($"Количество проигрышей данной команды: {count_lose}");

        // Задание Г
        // подсчет количества выигрышей, проигрышей и ничьих команды
        count_draw = CountDraw(resultOfGames);
    }

```



```

Console.WriteLine($"\\nКоличество выигрышей данной команды: {count_win}");
Console.WriteLine($"Количество проигрышей данной команды: {count_lose}");

Console.WriteLine($"Количество ничьих данной команды: {count_draw}");

// Задание Д
// подсчет количества игр, в которых разность забитых и пропущенных
больше или равна 3
count_game3 = CountGame3(resultOfGames);
Console.WriteLine($"\\nКоличество игр, в которых разность забитых и
пропущенных мячей больше или равна 3: {count_game3}");

// Задание Е
// подсчет количества очков команды
score = (count_win * 3) + (count_draw * 1);
Console.WriteLine($"\\nКоличество набранных очков данной команды:
{score}");
}
}

```

The screenshot shows the Microsoft Visual Studio Debug Console with the following content:

```

Игра 1 прошла со счетом 4 : 7
Игра 2 прошла со счетом 4 : 1
Игра 3 прошла со счетом 6 : 6
Игра 4 прошла со счетом 11 : 8
Игра 5 прошла со счетом 11 : 9
Игра 6 прошла со счетом 1 : 9
Игра 7 прошла со счетом 7 : 5
Игра 8 прошла со счетом 2 : 0
Игра 9 прошла со счетом 10 : 9
Игра 10 прошла со счетом 1 : 2
Игра 11 прошла со счетом 9 : 7
Игра 12 прошла со счетом 3 : 4
Игра 13 прошла со счетом 9 : 1
Игра 14 прошла со счетом 9 : 10
Игра 15 прошла со счетом 7 : 4
Игра 16 прошла со счетом 10 : 7
Игра 17 прошла со счетом 1 : 4
Игра 18 прошла со счетом 4 : 7
Игра 19 прошла со счетом 4 : 7
Игра 20 прошла со счетом 7 : 4

else
{
    // если разность меньше
    if (games < 0)
    {
        Console.WriteLine("
    }
    // если разность равна 0
    else

```

```

Microsoft Visual Studio Debug Console
Проигрыш .....if(Math.Abs(games)>=3){
Выигрыш .....{
Проигрыш .....count++;
Выигрыш .....}
Выигрыш .....}
Выигрыш .....return count;
Выигрыш .....}
Проигрыш }
Проигрыш }
Проигрыш .....static void Main(string[] args)
Выигрыш .....{
.....int[] goals; //забитые мячи
Количество выигршей данной команды: 11 //пропущенные
.....int[] resultOfGames; //разница между забитыми и пропущенными
Количество выигршей данной команды: 11 == 0; //выигрыши
Количество проигрышей данной команды: 8 == 0; //проигрыши
.....int count_draw = 0; //ничьи
Количество выигршей данной команды: 11 != 0; //разность больше или равно 3
Количество проигрышей данной команды: 8 //количество очков
Количество ничьих данной команды: 1
.....//заполнение массива
Количество игр, в которых разность забитых и пропущенных мячей больше или равна 3: 11
Количество набранных очков данной команды: 34 * 0; i++)
.....//вывод всех игр на экран
C:\Users\8-bit\source\repos\ConsoleApp1\bin\Debug\net5.0\ConsoleApp1.exe (process 7832) exited with code 0.
Press any key to close this window.
.....Console.WriteLine($"Игра {i+1} прошла со счетом {goals[i]}-{goals[i+1]}");
.....Console.WriteLine();
}

```

10. Описать процедуру Swar(x, y), меняющую содержимое переменных X и Y. С ее помощью для данных переменных A, B, C, D последовательно поменять содержимое следующих пар: A и B, C и D, B и C.

```

class Program
{
    static void Swap(ref double X, ref double Y)
    {
        double temp;
        temp = X;
        X = Y;
        Y = temp;
    }
    static void Main(string[] args)
    {
        double a = 15;
        double b = 5;
        double c = 99;
        double d = 150;

        Console.WriteLine($"Пары ({a},{b}) и ({c},{d})");

        Swap(ref a, ref b);
        Swap(ref c, ref d);
        Swap(ref b, ref c);

        Console.WriteLine($"Новые пары ({a},{b}) и ({c},{d})");
    }
}

```

```

Microsoft Visual Studio Debug Console
Пары (15,5) и (99,150)
Новые пары (5,150) и (15,99)

```