Программирование на C#

Семинар №13

Модуль №1

Тема:

Строки.

Класс String

Класс StringBuilder

Задания преподавателя к семинару

Класс StringBuilder

https://docs.microsoft.com/ru-ru/dotnet/api/system.text.stringbuilder?view=net-5.0

Класс String. https://learn.microsoft.com/ru-ru/dotnet/api/system.string?view=net-7.0

Строки и стандарт Юникода https://learn.microsoft.com/ru-ru/dotnet/api/system.string?view=net-7.0#Unicode

Полезные материалы к семинару

1. Выполните задания к семинару на слайдах 6 и 13.

Класс String

Операции над элементами строки – это все операции, определенные для типа **char.** Исключение: значение элемента строки изменить непосредственно невозможно – строки неизменяемы.

```
int K; string s=”ABBA”; // объявление стоки с инициализацией
K=s[0]; // Код => код символа ‘A’=65 по таблице кодировки
s[0]=’Z’;  // ошибка!
```

В отличие от массива копируется не ссылка, а ***значение*** строки.

```
string s1=”ВАСЯ”, s2;
s2=s1;
```

Операции: == != сравнение строк

больше(>), больше или равно(>=), меньше(<), меньше или равно(<=) – сравнение ссылок

Ошибка компиляции

```
string s1 = "Vasia";
string s2 = "Anna";
Console.WriteLine(s1 > s2 ? s1 : s2);
```

Некоторые Методы Класса String

```
string s1=”око , за”;
string[]  word;
char[] sep; // массив разделителей
sep = new char[]{‘ ’, ‘,’};
// с 8-й позиции массива s1 дописывается массив “_око”
s1=s1.Insert(8, “_око”);
// с 8-й позиции массива s1 дописывается массив “_око”
// заменили око на зуб
s1=s1.Replace(“око”, “зуб”); // зуб,_за_ зуб

word=s1.Split(sep); //делим строку на массив строк-слов
// из строки получили массив строк
s1=string.Join (“>”, word);
```

зуб>>>за>зуб

Self Задания

Self 01. Пользователь вводит предложение (строку). Напечатать все его различные слова (уникальные). Слово – последовательность символов в предложении, не содержащая пробелов.

Self 02. Пользователь вводит строку. Отсортировать все слова строки по длине и вывести на экран.

Self 03. Пользователь вводит строку. Вывеси на экран только те слова, которые начинаются с латинских букв.

Self 04. Пользователь вводит строку цифр через пробел:

- Удалить из каждого слова цифру 7
- Добавить в начало каждого слова три цифры 2
- Дописать в конец каждого слова цифру 5

После каждого действия выводить массив слов на экран

Self Задания

Self 05. Пользователь вводит с клавиатуры строку, в которой несколько раз идут подряд одинаковые цифры. Найти наибольшее число, образованное этими цифрами:

1111 222в 999 с8795ак 45т**6789**

Self 06. Баланс скобок. Дан текст с открывающимися и закрывающимися круглыми скобками. Определить, что все скобки имеют пару с учетом их вложенности. Если есть ошибки, то написать о них на экране.

Self 07*. Поиск в тексте подстроки «ABCD». Текст получать из текстового файла input.txt, размещённого в папке с приложением. Текст может состоять из нескольких строк. При отсутствии подстрок сообщить об этом.

Класс StringBuilder

Пространства имён - System.Text

Класс StringBuilder предназначен для динамического редактирования строк.

Создание динамической строки:

1. StringBuilder ИмяСтроки;
2. ИмяСтроки=new StringBuilder();

Отличия от обычной строки класса String:

1. Элементы динамической строки можно изменять напрямую, путем присваивания.

2. В операциях ==, !=, = участвуют не элементы строки, а адреса (аналогично массиву).

Длину хранит поле объекта: Length

Методы класса StringBuilder

```
//Добавление символа в конец строки.
    ИмяСтроки.Append (символ)
//Добавление заданного количества символов в конец строки
    ИмяСтроки.Append (символ, кол-во)
//Удаление заданного кол-ва символов с заданной позиции
ИмяСтроки.Remove (номер позиции, кол-во символов)
```

Трансформации полезны

Из массива символов ⟶ в строку

```
char[] a => string s
            s=new string(a);
```

Из строки ⟶ в массив символов

```
string s => char[] a
            s.ToCharArray;
```

Из строки ⟶ в динамическую строку

```
string s => StringBuilder b
            b=new StringBuilder(s);
```

Из динамической строки ⟶ в строку

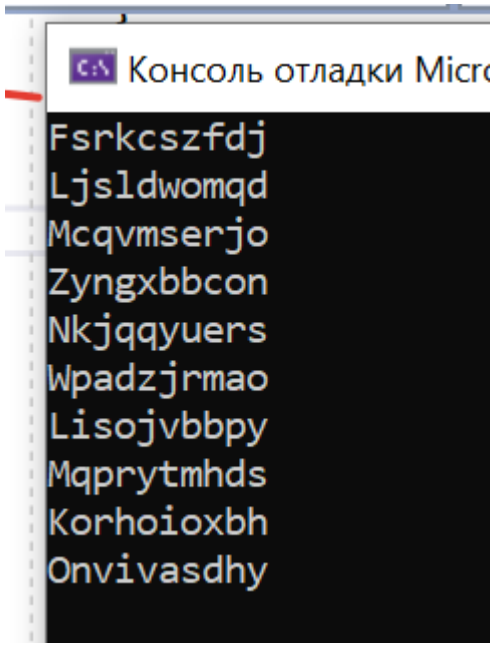
```
   StringBuilder b => string s
            s=b.ToString();
```

Demo. Генерация строк

```
using System.Text;
using System;
namespace BuilderDemo
{
    internal class Program
    {
        static void Main()
        {
         // С StringBuilder генерация работает быстро, т.к.
         //на каждой итерации цикла строки не создаются заново.
            for (int i = 0; i < 10; ++i)
            { Console.WriteLine(GetRandomString(10)); }
        }
```

Demo. Генерация строк

```
public static string GetRandomString(int len)
        {
            Random rnd = new Random();
            StringBuilder sBuilder = new(len);
            sBuilder.Append((char)rnd.Next('A', 'Z' + 1));
            for (int i = 0; i < len - 1; ++i)
            { sBuilder.Append((char)rnd.Next('a', 'z' + 1)); }
            string str = sBuilder.ToString();
            return str;
        }
    }
}
```



Self. Задания

Self 08. Сохранить исходный текст (несколько строк) в файл в разных кодовых страницах. Потом считать его тоже в разных кодировках и вывести на экран результат.

Self 09. Проверить, как работает конкатенация строк через объекты String и StringBuilder

Self 10. Пользователь вводит строку. Напечатать все различные слова, встреченные в ней. Слово – последовательность символов, не содержащая пробелов.

Self 11. Выявить во введенном с клавиатуры тексте слова (текст может вводиться как одной, так и несколькими строками), содержащие гласные латинские буквы, и записать в результирующую строку. Вывести на экран.