

В.В. Подбельский, О.В. Максименкова

Программирование на C#

Семинар 8

Класс `string` и его Члены

Задача 1

Определите методы:

1. **CreateString(int len, char min, char max)** - для создания строки заданной длины из символов, случайно выбираемых из указанного диапазона (верхняя граница включительно).
2. **MoveOff(string source, string charsToDelete)** - для удаления из строки source всех символов строки charsToDelete.

Самостоятельно в основной программе получите от пользователя целое число **N**, создайте строку из символов **N** десятичных цифр. Удалите из неё чётные цифры.

Задача 1

```
// Создать строку заданных размеров (len) из заданных СИМВОЛОВ
public static string CreateString(int len, char min, char max)
{
    if (len < 0)
        return null; // throw new Exception("Аргумент метода должен быть положительным!");
    // min, max- Границы диапазона СИМВОЛОВ.
    if (maxChar < minChar)
    {
        char c = min;
        min = max;
        max = c;
    }
    // пустая строка, останется пустой, если СИМВОЛОВ 0
    string line = string.Empty;
    for (int i = 0; i < len; i++)
        line += (char)gen.Next(min, max + 1);
    return line;
}
```

Задача 1

```
static Random gen = new Random();

public static int GetIntValue(string prompt) {
    int intVal;
    do
        Console.Write(prompt);
    while (!int.TryParse(Console.ReadLine(), out intVal));
    return intVal;
}

public static string MoveOff(string source, string charsToDelete) {
    string res = source;
    int index;
    for (int i = 0; i < charsToDelete.Length; i++)
        while ((index = res.IndexOf(charsToDelete[i])) >= 0)
            res = res.Remove(index, 1);
    return res;
}
```

Задача 2

Программа получает на вход строку из латинских символов, пробелов и точек с запятой.

Пример:

Let it be; All you need is love; Dizzy Miss Lizzy

Каждую подстроку, стоящую между точками с запятой, преобразуйте в аббревиатуру, сокращая каждое отдельное слово подстроки «до первой гласной» (гласная входит в сокращённую запись). В полученной аббревиатуре каждую первую букву отдельного слова запишите в верхнем регистре. Результат выведите на экран, размещая аббревиатуры в столбик, например, для приведённого выше предложения получим:

LeIBe

AYNeLlo

DiMiLi

Задача 2

- Определим действия, которые потребуются при решении задачи:
 - Проверка, что строка состоит только из латинских символов и пробелов:
`bool Validate(string str)`
 - Получение массива строк из строки, в которой подстроки связаны (или разделены) точками с запятой:
`string[] ValidatedSplit(string str, char ch)`
 - Удаление из слова букв, размещённых после первой гласной:
`string Shorten(string str)`
 - Приведение первой буквы слова к верхнему регистру, а остальных - к нижнему:
`void FirstUppcase(ref string str)`

Задача 2

- Методы разместите в отдельном от основной программы классе в отдельном файле. Для решения задачи попробуйте модифицировать готовые методы:
 - `public static bool Validate(string str)`
 - `public static string[] ValidatedSplit(string str, char ch)`
 - `public static string Shorten(string str)`
 - `public static string Abbreviation(string str)`
 - `public static void FirstUppcase(ref string str)`
- Код основной программы реализуйте самостоятельно. Получайте от пользователя строки, разделённые точками с запятой, преобразуйте их к аббревиатурам, а для некорректных строк выводите сообщения об ошибке.

Задача 2

```
// проверка, что строки состоят только из символов латинского алфавита
// и пробелов
public static bool Validate(string str) {
    // TODO: требуется валидировать строки с заглавные латинскими
    // символами также как верные
    char[] english = new char[27];
    english[0] = ' ';
    for (int i = 1; i < english.Length; i++) {
        english[i] = (char)('a' + i);
    }
    if (str.IndexOfAny(english) < 0) return false;
    return true;
} // end of Validate(string)
// получение массива строк
// каждый элемент проверен на соответствие формату
public static string[] ValidatedSplit(string str, char ch) {
    string[] output = null;
    output = str.Split(ch);
    foreach (string s in output) {
        if (!Validate(s)) return null;
    }
    return output;
} // end of ValidatdSplit(string, char)
```


Задача 2

```
// Обрезка строки по первому гласному
public static string Shorten(string str) {
    // TODO: учесть заглавные гласные
    char[] alph = { 'a', 'e', 'i', 'o', 'u', 'y' };
    int ind = str.IndexOfAny(alph);
    return str.Substring(0, ind + 1);
} // end of Shorten(string)
// Метод создания аббревиатуры для ПОДстроки (в ней много слов)
public static string Abbreviation(string str) {
    string output = String.Empty;
    if (str != String.Empty) {
        string[] tmp = str.Split(' ');
        foreach (string s in tmp) {
            string shortenS = Shorten(s);
            FirstUppcase(ref shortenS);
            output += shortenS;
        }
    }
    return output;
} // end of Abbreviation(string)
```

Задача 2

```
// Метод преобразования первого символа к заглавному
public static void FirstUppcase(ref string str) {
// TODO: буквы после первой могут быть не приведены к нижнему
// регистру
str = str[0].ToString().ToUpper() + str.Substring(1);
} // end of FirstUppcase(ref string)
```

Реализуйте отмеченную в **TODO** функциональность.

Внимание!

В коде методов есть ошибки – исправьте их и отладьте приложение!

Подумайте об оптимизации работы приложения...

Задача 3: Смена Входного Поточка Данных

1. Свяжите входной поток с файлом Program.cs. Иными словами, чтение входных данных должно осуществляться не с консоли, а из данного файла;
2. Прочитайте файл Program.cs;
3. Проанализируйте количество операторных скобок;
4. Вычислите количество вхождений каждой строчной буквы латинского алфавита.

Задача 3

```
static int[] stat = new int[26]; // статистика по лат. буквам
static void Main(string[] args) {
    string tmp;
    int openBrackets = 0; // количество {
    int closedBrackets = 0; // количество }
    int total = 0; // общее количество символов файла

    var In = Console.In; // Запоминаем стандартный входной поток
    // Создаем файл и текстовый входной поток:
    StreamReader stream_in = new StreamReader(@"..\..\..\Program.cs");
    // Настраиваем стандартный входной поток на чтение из файла:
    Console.SetIn(stream_in);

    // чтение из файла
    // восстановление потока
} // end of Main()
```

Задача 3

```
while (true) { // цикл бесконечен
    tmp = stream_in.ReadLine();
    if (tmp == null) break; // условие прерывание цикла
    total += tmp.Length;
    // подсчёт количества фигурных скобок
    BracketsCount(tmp, ref openBrackets, ref closedBrackets);
    Console.WriteLine(tmp.Trim());
    Console.WriteLine(tmp);
}
// восстанавливаем состояние потока
stream_in.Close();
Console.SetIn(In);
// обрабатываем данные по скобкам
tmp = "Баланс скобок не соблюден";
if (openBrackets == closedBrackets)
    tmp = "Баланс скобок соблюден, количество блоков " + closedBrackets;
Console.WriteLine(StatToString());
Console.WriteLine(tmp);
Console.WriteLine("Для завершения работы нажмите любую клавишу.");
Console.ReadKey();
```

Задача 3

```
/// <summary>
/// Вычисляет количество открывающихся и закрывающихся скобок в строке
/// </summary>
/// <param name="tmp">строка СИМВОЛОВ</param>
/// <param name="openBrackets">количество открывающихся скобок</param>
/// <param name="closedBrackets">количество закрывающихся скобок</param>
private static void BracketsCount(string tmp, ref int openBrackets, ref int closedBrackets)
{
    for (int i = 0; i < tmp.Length; i++) {
        // статистика по строчным латинским символам
        if (tmp[i] >= 'a' && tmp[i] <= 'z')
            stat[tmp[i] - 'a']++;
        if (tmp[i] == '{') openBrackets++;
        if (tmp[i] == '}') closedBrackets++;
    }
}
```

Задача 3

```
/// <summary>
/// метод формирует строку со статистикой по строчным латинским,
/// символам, содержащимся в тексте файла
/// </summary>
/// <returns>возвращает строку с представлением статистики</returns>
public static string StatToString() {
    string output = string.Empty;
    for (int i = 0; i < stat.Length; i++) {
        output += (char)('a' + i) + " - " + stat[i] + " ";
    }
    return output;
}
```

Задача 3: Самостоятельно

1. Не все скобки в коде программы означают блоки, некоторые могут быть включены в строки. Модифицируйте код так, чтобы считались только валидные операторные скобки.
2. Переназначьте поток вывода `Console.Out` в произвольный текстовый файл, путь до которого указывается при запуске программы. Весь вывод программы должен быть перенаправлен в файл. Восстановите поток перед запросом нажатия клавиши.

Задача 4

Определите вспомогательный метод для выделения из символьного массива подмассива, не содержащего конкретных символов, определяемых параметром типа `string`.

Заголовок метода:

```
static char [] Row(char [] line, string model)
```

`line` – исходный массив;

`model` – строка символов, которые не должны войти в подмассив.

При отсутствии результата возвращайте пустой массив.

Задача 5

Напишите метод **ConvertHex2Bin()**, выполняющий перевод шестнадцатеричного числа в двоичное. Метод имеет следующий заголовок:

string ConvertHex2Bin(string HexNumber)

HexNumber – строка, представляющая шестнадцатеричное число, например *5A1*. Функция должна возвращать строку с двоичным представлением числа. Например, для шестнадцатеричного числа, представленного строкой *5A1*, функция должна вернуть строку *10110100001*.

Задача 6

В методе `Main()` введите размер символьного массива и процент букв в нём. Сформируйте массив из букв и цифр и выведите соответствующую ему строку, заменив символы цифр их названиями.

Задача 6

Определите метод, формирующий случайно заполненный символьный массив, элементы которого представляют латинские буквы и десятичные цифры. Первый параметр – длина массива, второй параметр задаёт в процентах отношение количества букв к общему количеству символов в массиве.

Псевдокод метода:

```
static char [] Series(int k, int ratio) // Заголовок метода
```

```
{
```

1. Определить ссылку типа `char []` и символьный массив из `k` элементов;
2. Присвоить случайно выбираемым элементам массива значения случайных латинских букв. (Их количество равно $k \cdot \text{ratio} / 100$.);
3. Присвоить всем “незаполненным” элементам массива значения случайно выбираемых десятичных цифр;
4. Вернуть ссылку на заполненный массив.

```
}
```

Задача 6

Определите метод, преобразующий символьный массив в строку. Элементы массива, представляющие десятичные цифры в строке, нужно заменить их названиями на русском языке. Например, символ '7' заменить на «семь».

Псевдокод метода:

```
static string Line(char []series) // Заголовок метода
{
    1. Определить ссылку result типа string, связав с ней пустую строку;
    2. Определить массив строк с русскими названиями цифр;
    3. Цикл до конца массива символов;
    4. Выделить из массива буквы от текущей позиции до ближайшей цифры, создать из них строку и присоединить ее к result;
    5. Присоединить к result русское название цифры;
    6. Конец цикла;
    7. Вернуть ссылку result.
}
```

В основной программе:

- Введите параметры генерации строки и сгенерировать саму строку;
- Выведите результат;
- Заменить в строке все цифры их названиями;
- Выведите результат.

Решите Самостоятельно

Для решения заданий из данной группы используйте "однопроходные" алгоритмы, позволяющие получить требуемый результат после однократного просмотра набора исходных данных.

Задание 1

Дана строка (вводится пользователем), состоящая из русских слов, разделённых пробелами (одним или несколькими). Преобразуйте её так, чтобы между словами был ровно один пробел и выведите результат.

Задание 2

Дана строка (вводится пользователем), состоящая из русских слов, разделённых пробелами (одним или несколькими). Выведите количество слов, состоящих более чем из четырёх букв.

Задание 3

Дана строка (вводится пользователем), состоящая из русских слов, разделённых пробелами (одним или несколькими). Выведите количество слов, начинающихся с гласной буквы.

Дополнительный Пункт к Задаче 1

Внимание, данное дополнительное задание предназначено только для знакомых с ООП (ООП в рамках курса будет разбираться позже, поэтому если возникают трудности, не переживайте)

На основе предложенного на слайдах задачи 1 кода разработайте класс **UserString**:

- Поле класса – строка;
- Преобразуйте метод **CreateString()** в конструктор с параметрами;
- Метод **GetIntValue()** вынесите в отдельный класс;
- Остальные методы из статических преобразуйте в экземплярные методы класса **UserString**.