**Практическая работа №4**

**Строки**

***Цель работы:*** получение практических навыков по работе со строковыми данными и объектами.

***Задачи:***

1. Изучить теоретический материал

2. Написать программы для решения практических заданий

3. Ответить на контрольные вопросы.

4. Оформить отчет.

***Теоретический материал***

Для хранения строк в языке C# используется тип string. Чтобы объявить (и, как правило, сразу инициализировать) строковую переменную, можно написать следующий код:

string a = "Текст"; string b = "строки";

Над строками можно выполнять операцию сложения – в этом случае текст одной строки будет добавлен к тексту другой:

string c = a + " " + b; // Результат: Текст строки

Тип string на самом деле является псевдонимом для классаString, с помощью которого над строками можно выполнять ряд более сложных операций. Например, методIndexOf может осуществлять поиск подстроки в строке, а методSubstring возвращает часть строки указанной длины, начиная с указанной позиции:

string a = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

int index = a.IndexOf("OP");

// Результат: 14 (счёт с 0) string b = a.Substring(3, 5);// Результат: DEFGH

Если требуется добавить в строку специальные символы, это можно сделать с помощью escape-последовательностей, начинающихся с обратного слэша:

|  |  |
| --- | --- |
| \" | Кавычка |
| \\ | Обратная косая черта |
| \n | Новая строка |
| \r | Возврат каретки |
| \t | Горизонтальная табуляция |
|  | Более эффективная работа со строками |

Строки типа string представляют собой неизменяемые объекты: после того, как строка инициализирована, изменить её уже нельзя. Рассмотрим для примера следующий код:

string s ="Hello, "; s +="world!";

Здесь компилятор создаёт в памяти строковый объект и инициализирует его строкой «*Hello,*», а затем создаёт другой строковый объект и инициализирует его значением первого объекта и новой строкой «*world!*», а затем заменяет значение переменнойs на новый объект. В результате строкаs содержит именно то, что хотел программист, однако в памяти остаётся и изначальный объект со строкой «*Hello,*». Конечно, со временем сборщик мусора уничтожит этот бесхозный объект, однако если в программе идёт интенсивная работа со строками, то таких бесхозных объектов может оказаться очень много. Как правило, это негативно сказывается на производительности программы и объеме потребляемой ею памяти.

Для строк определены следующие операции:

• присваивание (=);

• проверка на равенство (==);

• проверка на неравенство ( ! = ) ;

• обращение по индексу ( [ ] ) ;

• сцепление (конкатенация) строк ( + ) .

**Основные методы строк**

Основная функциональность класса String раскрывается через его методы, среди которых можно выделить следующие:

* **Compare**: сравнивает две строки с учетом текущей культуры (локали) пользователя
* **CompareOrdinal**: сравнивает две строки без учета локали
* **Contains**: определяет, содержится ли подстрока в строке
* **Concat**: соединяет строки
* **CopyTo**: копирует часть строки или всю строку в другую строку
* **EndsWith**: определяет, совпадает ли конец строки с подстрокой
* **Format**: форматирует строку
* **IndexOf**: находит индекс первого вхождения символа или подстроки в строке
* **Insert**: вставляет в строку подстроку
* **Join**: соединяет элементы массива строк
* **LastIndexOf**: находит индекс последнего вхождения символа или подстроки в строке
* **Replace**: замещает в строке символ или подстроку другим символом или подстрокой
* **Remove:** удаляет подстроку из заданной позиции
* **StartsWith, EndsWith:** возвращает true или false в зависимости от того, начинается ли или заканчивается строка заданной подстрокой.
* **Split**: разделяет одну строку на массив строк
* **Substring**: извлекает из строки подстроку, начиная с указанной позиции
* **ToLower**: переводит все символы строки в нижний регистр
* **ToUpper**: переводит все символы строки в верхний регистр
* **Trim**: удаляет начальные и конечные пробелы из строки

**Конкатенация**

Конкатенация строк или объединение может производиться как с помощью операции +, так и с помощью метода Concat:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | string s1 = "hello";  string s2 = "world";  string s3 = s1 + " " + s2; // результат: строка "hello world"  string s4 = String.Concat(s3, "!!!"); // результат: строка "hello world!!!"    Console.WriteLine(s4); |

Метод Concat является статическим методом класса String, принимающим в качестве параметров две строки. Также имеются другие версии метода, принимающие другое количество параметров.

Для объединения строк также может использоваться метод Join:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | string s5 = "apple";  string s6 = "a day";  string s7 = "keeps";  string s8 = "a doctor";  string s9 = "away";  string[] values = new string[] { s5, s6, s7, s8, s9 };    String s10 = String.Join(" ", values);  // результат: строка "apple a day keeps a doctor away" |

Метод Join также является статическим. Использованная выше версия метода получает два параметра: строку-разделитель (в данном случае пробел) и массив строк, которые будут соединяться и разделяться разделителем.

**Сравнение строк**

Для сравнения строк применяется статический метод Compare:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | string s1 = "hello";  string s2 = "world";    int result = String.Compare(s1, s2);  if (result<0)  {      Console.WriteLine("Строка s1 перед строкой s2");  }  else if (result > 0)  {      Console.WriteLine("Строка s1 стоит после строки s2");  }  else  {      Console.WriteLine("Строки s1 и s2 идентичны");  }  // результатом будет "Строка s1 перед строкой s2" |

Данная версия метода Compare принимает две строки и возвращает число. Если первая строка по алфавиту стоит выше второй, то возвращается число меньше нуля. В противном случае возвращается число больше нуля. И третий случай - если строки равны, то возвращается число 0.

В данном случае так как символ h по алфавиту стоит выше символа w, то и первая строка будет стоять выше.

**Поиск в строке**

С помощью метода IndexOf мы можем определить индекс первого вхождения отдельного символа или подстроки в строке:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | string s1 = "hello world";  char ch = 'o';  int indexOfChar = s1.IndexOf(ch); // равно 4  Console.WriteLine(indexOfChar);    string subString = "wor";  int indexOfSubstring = s1.IndexOf(subString); // равно 6  Console.WriteLine(indexOfSubstring); |

Подобным образом действует метод LastIndexOf, только находит индекс последнего вхождения символа или подстроки в строку.

Еще одна группа методов позволяет узнать начинается или заканчивается ли строка на определенную подстроку. Для этого предназначены методы StartsWith и EndsWith. Например, у нас есть задача удалить из папки все файлы с расширением exe:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | string path = @"C:\SomeDir";    string[] files = Directory.GetFiles(path);    for (int i = 0; i < files.Length; i++)  {      if(files[i].EndsWith(".exe"))          File.Delete(files[i]);  } |

**Разделение строк**

С помощью функции Split мы можем разделить строку на массив подстрок. В качестве параметра функция Split принимает массив символов или строк, которые и будут служить разделителями. Например, подсчитаем количество слов в сроке, разделив ее по пробельным символам:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | string text = "И поэтому все так произошло";    string[] words = text.Split(new char[] { ' ' });    foreach (string s in words)  {      Console.WriteLine(s);  } |

Это не лучший способ разделения по пробелам, так как во входной строке у нас могло бы быть несколько подряд идущих пробелов и в итоговый массив также бы попадали пробелы, поэтому лучше использовать другую версию метода:

|  |  |
| --- | --- |
| 1 | string[] words = text.Split(new char[] { ' ' }, StringSplitOptions.RemoveEmptyEntries); |

Второй параметр StringSplitOptions.RemoveEmptyEntries говорит, что надо удалить все пустые подстроки.

**Обрезка строки**

Для обрезки начальных или концевых символов используется функция Trim:

|  |  |
| --- | --- |
| 1  2  3  4 | string text = " hello world ";    text = text.Trim(); // результат "hello world"  text = text.Trim(new char[] { 'd', 'h' }); // результат "ello worl" |

Функция Trim без параметров обрезает начальные и конечные пробелы и возвращает обрезанную строку. Чтобы явным образом указать, какие начальные и конечные символы следует обрезать, мы можем передать в функцию массив этих символов.

Эта функция имеет частичные аналоги: функция TrimStart обрезает начальные символы, а функция TrimEnd обрезает конечные символы.

Обрезать определенную часть строки позволяет функция **Substring**:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | string text = "Хороший день";  // обрезаем начиная с третьего символа  text = text.Substring(2);  // результат "роший день"  Console.WriteLine(text);  // обрезаем сначала до последних двух символов  text = text.Substring(0, text.Length - 2);  // результат "роший де"   Console.WriteLine(text); |

Функция Substring также возвращает обрезанную строку. В качестве параметра первая использованная версия применяет индекс, начиная с которого надо обрезать строку. Вторая версия применяет два параметра - индекс начала обрезки и длину вырезаемой части строки.

**Вставка**

Для вставки одной строки в другую применяется функция Insert:

|  |  |
| --- | --- |
| 1  2  3  4  5 | string text = "Хороший день";  string subString = "замечательный ";    text = text.Insert(8, subString);  Console.WriteLine(text); |

Первым параметром в функции Insert является индекс, по которому надо вставлять подстроку, а второй параметр - собственно подстрока.

**Удаление строк**

Удалить часть строки помогает метод Remove:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | string text = "Хороший день";  // индекс последнего символа  int ind = text.Length - 1;  // вырезаем последний символ  text = text.Remove(ind);  Console.WriteLine(text);    // вырезаем первые два символа  text = text.Remove(0, 2); |

Первая версия метода Remove принимает индекс в строке, начиная с которого надо удалить все символы. Вторая версия принимает еще один параметр - сколько символов надо удалить.

**Замена**

Чтобы заменить один символ или подстроку на другую, применяется метод **Replace**:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | string text = "хороший день";    text = text.Replace("хороший", "плохой");  Console.WriteLine(text);    text = text.Replace("о", "");  Console.WriteLine(text); |

Во втором случае применения функции Replace строка из одного символа "о" заменяется на пустую строку, то есть фактически удаляется из текста. Подобным способом легко удалять какой-то определенный текст в строках.

**Смена регистра**

Для приведения строки к верхнему и нижнему регистру используются соответственно функции **ToUpper()** и **ToLower()**:

|  |  |
| --- | --- |
| 1  2  3  4 | string hello = "Hello world!";    Console.WriteLine(hello.ToLower()); // hello world!  Console.WriteLine(hello.ToUpper()); // HELLO WORLD! |

Строки типа StringBuilder

Чтобы компилятор не создавал каждый раз новый строковый объект, разработчики языка C# ввели другой строковый класс: StringBuilder. Приведённый выше пример с использованием этого класса будет выглядеть следующим образом:

StringBuilder s = new StringBuilder("Hello, "); s.Append("world!");

Конечно, визуально этот код выглядит более сложным, зато при активном использовании строк в программе он будет гораздо эффективнее. Помимо добавления строки к существующему объекту (Append) класс StringBuilder имеет ещё ряд полезных методов:

|  |  |
| --- | --- |
| Insert | Вставляет указанный текст в нужную позицию исходной строки |
|  |  |
| Remove | Удаляет часть строки |
| Replace | Заменяет указанный текст в строке на другой |

Если нужно преобразовать объект StringBuilder в обычную строку, то для этого можно использовать методToString():

StringBuilder s = new StringBuilder("Яблоко"); string a = s.ToString();

Элемент управления ListBox

Элемент управления ListBox представляет собой список, элементы которого выбираются при помощи клавиатуры или мыши. Список элементов задается свойствомItems.Items – это элемент, который имеет свои свойства и свои методы. МетодыAdd,RemoveAt иInsertиспользуются для добавления, удаления и вставки элементов.

Объект Items хранит объекты, находящиеся в списке. Объект может быть любым классом – данные класса преобразуются для отображения в строковое представление методом ToString(). В нашем случае в качестве объекта будут выступать строки. Однако, поскольку объект Items хранит объекты, *приведённые*к типу object, перед использованием необходимо *привести*их обратно к изначальному типу, в нашем случае string:

string a = (string)listBox1.Items[0];

Для определения номера выделенного элемента используется свойство SelectedIndex.

***Практические задания***

|  |  |
| --- | --- |
| Вариант | Задание |
| 1 | * Дана строка символов, состоящая из произвольных десятичных цифр, разделенных пробелами. Вывести количество четных чисел в этой строке. * Строка состоит из слов, разделенных одним или несколькими пробелами. Переставьте слова в алфавитном порядке |
| 2 | * Дана строка символов. Вывести на экран цифры, содержащиеся в строке. * Вывести текст, составленный из первых букв всех слов |
| 3 | * Дана строка символов, содержащая некоторый текст на русском языке. Заменить все большие буквы «А» на символ «\*». * Дана строка, содержащая полное имя файла (например, 'c:\WebServers\home\testsite\www\myfile.txt'). Выделите из этой строки имя файла без расширения |
| 4 | * Написать алгоритм генерации пароля. * Дана строка. Вставить после каждого символа два случайных символа |
| 5 | * Посчитать в строке количество слов. * В строке записано десятичное число. Запишите данное число римскими цифрами |
| 6 | * Найти количество знаков препинания в исходной строке. * Дан email в строке. Определить, является ли он корректным (наличие символа @ и точки, наличие не менее двух символов после последней точки и т.д.) |
| 7 | * Дана строка символов. Вывести на экран количество строчных русских букв, входящих в эту строку. * Дана строка. Удалите все цифры в ней. |
| 8 | * Дана строка символов. Вывести на экран только строчные русские буквы, входящие в эту строку. * Даны две строки, состоящие из слов, разделенных пробелами. Сформировать строку, состоящую из слов, которые: а) встречаются хотя бы в одной строке; б) встречаются только в первой строке; в) встречаются только в одной из строк |
| 9 | * Дан текст. Некоторые его фрагменты выделены группами символов ##. Заменить выделение группами символов '<' и '>)'. Пример: 'Это ##тестовый пример## для задачи ##на## строки' преобразуется в 'Это <тестовый> пример для задачи <на> строки' * Написать функцию генерации email |
| 10 | * Дана строка символов, состоящая из произвольного текста на английском языке, слова разделены пробелами. Удалить первую букву в каждом слове. * Исключить из строки группы символов, расположенные между символами «/\*», «\*/» включая границы . Предполагается, что нет вложенных скобок |
| 11 | * Даны две строки. Определите, можно ли из некоторых символов первой строки составить вторую строку. * Вывести слово, наиболее часто встречающееся в строке; последовательность слов, в которых буквы упорядочены по алфавиту |
| 12 | * Написать функцию генерации email. * Определить является ли введенный текст записью целого числа, записью вещественного числа |
| 13 | * Дана строка, состоящая из групп нулей и единиц. Посчитать количество нулей и единиц. * Отредактировать заданное предложение, удаляя из него слова, которые встречаются в предложении n раз |
| 14 | * Строка состоит из слов, разделенных одним или несколькими пробелами. Поменяйте местами наибольшее по длине слово и наименьшее. * Расстояние между двумя словами равной длины – количество букв, в которых различаются эти слова. В заданном предложении найти пару наиболее далеко удаленных слов заданной длины |
| 15 | * Строка состоит из слов, разделенных одним или несколькими пробелами. Переставьте слова по убыванию их длин. * Строка состоит из слов, разделенных одним или несколькими пробелами. Найдите слово наибольшей длины |
| 16 | * Дана строка. Если символы в ней упорядочены по алфавиту, то вывести 'yes', иначе вывести первый символ, нарушающий алфавитный порядок. * Перечислить все слова заданного предложения, которые состоят из тех же букв, что и первое слово предложения. |
| 17 | * Дан текст. Найдите в нем все числа, окруженные пробелами, и добавьте перед ними '<' и после них '>'. * Дана строка, содержащая сумму натуральных чисел. Найдите результат вычислений |
| 18 | * Дана строка символов, состоящая из произвольного текста на английском языке, слова разделены пробелами. В каждом слове заменить первую букву на прописную. * Написать программу, которая преобразует введенное целое число в его текстовое представление (например 125 – сто двадцать пять) |
| 19 | * Дан текст. Заменить все вхождения наибольшей цифры ее словесным написание. * Дана строка. Удалите k-ый символ в ней |
| 20 | * Даны две строки, состоящие из слов, разделенных пробелами. Сформировать строку, состоящую из слов, которые: а) встречаются хотя бы в одной строке; б) встречаются только в первой строке; в) встречаются только в одной из строк. * Дан текст. Заменить все вхождения наибольшей цифры ее словесным написанием |
| 21 | * Исключить из строки группы символов, расположенные между символами «/\*», «\*/» включая границы . Предполагается, что нет вложенных скобок. * Для встречающихся в заданном тексте пар рядом расположенных символов указать, сколько раз встречается в тексте каждое из таких двухбуквенных сочетаний |
| 22 | * Заданное натуральное число от 1 до 1999 вывести римскими цифрами. * Дана строка. Вставить после каждого символа два случайных символа |
| 23 | * Из заданного текста выбрать и напечатать те символы, которые встречаются в нем ровно два раза (в том порядке, как они встречаются в тексте). * Дано натуральное число. Получить строку, в которой тройки цифр этого числа разделены пробелом, начиная с правого конца. Например, число 1234567 преобразуется в 1 234 567 |
| 24 | * Дана строка, представляющая из себя арифметическое выражение, состоящее из чисел, скобок и арифметических операций. Проверьте данное выражение на правильность расстановки скобок. * Строка состоит из слов, разделенных одним или несколькими пробелами. Найдите слово наибольшей длины |
| 25 | * Написать генерацию строк длины 10, причем первые 4 символа - цифры, следующие два символы - различные буквы, следующие 4 символа - нули или единицы, причем одна единица точно присутствует. * Дана строка. Заменить все символы 'a' и 'b' на 'A' и 'B' соответственно |

***Контрольные вопросы***

1. Какой тип данных используется для работы со строками?
2. Основные функции для работы со строками.

***Содержание отчета***

1. Титульный лист

2. Цели, задачи работы

Для каждого задания:

3. Текст задания

4. Листинг программы

5. Результаты работы

6. Ответы на контрольные вопросы