

Практическая работа № 5.

ПРОГРАММИРОВАНИЕ С ИСПОЛЬЗОВАНИЕМ СТРОК

Цель практической работы: изучить правила работы с компонентом ListBox.
Написать программу для работы со строками.

5.1. Тип данных *string*

Для хранения строк в языке C# используется тип `string`. Так, 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 класса `String`, с помощью которого над строками можно выполнять ряд более сложных операций. Например, метод `IndexOf` может осуществлять поиск подстроки в строке, а метод `Substring` 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:

Escape-последовательность	Действие
\"	Кавычка
\\	Обратная косая черта
\n	Новая строка
\r	Возврат каретки
\t	Горизонтальная табуляция

5.2. Компонент *ListBox*

Компонент **ListBox** представляет собой список, элементы которого выбираются при помощи клавиатуры или мыши. Список элементов задается свойством **Items**. **Items** – это элемент, который имеет свои свойства и свои методы. Методы **Add**, **RemoveAt** и **Insert** используются 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**.

5.3. Порядок выполнения индивидуального задания

Задание: Написать программу подсчета числа слов в произвольной строке. В качестве разделителя может быть любое число пробелов. Для ввода строк

использовать **ListBox**. Строки вводятся на этапе проектирования формы, используя окно свойств. Вывод результата организовать в метку **Label**.

Панель диалога будет иметь вид:

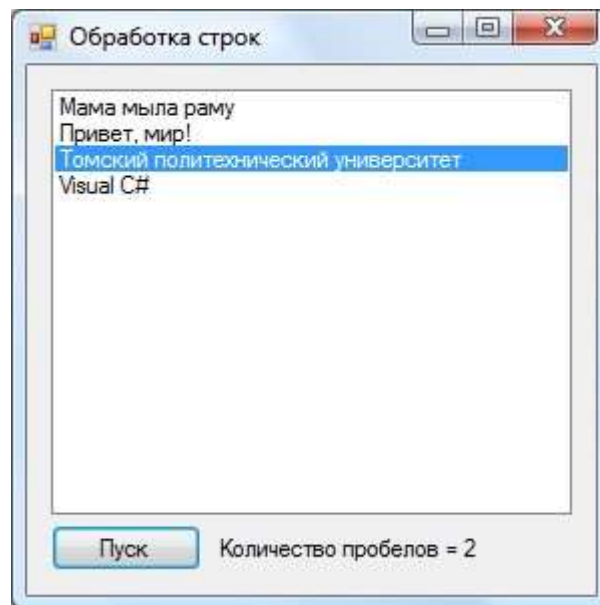


Рис. 5.1. Окно программы обработки строк

Текст обработчика нажатия кнопки «Пуск» приведен ниже.

```
private void button1_Click(object sender, EventArgs e)
{
    // Получаем номер выделенной строки
    int index = listBox1.SelectedIndex;
    // Считываем строку в переменную str
    string str = (string)listBox1.Items[index];
    // Узнаем количество символов в строке
    int len = str.Length;
    // Считаем, что количество пробелов равно 0
    int count = 0;
```

```

// Устанавливаем счетчик символов в 0
int i = 0;
//Организуем цикл перебора всех символов в строке
while (i < len - 1)
{
    // Если нашли пробел, то увеличиваем
    // счетчик пробелов на 1
    if (str[i] == ' ')        count++;
    i++;
}
label1.Text = "Количество пробелов = " +
count.ToString(); }

```

5.4. Индивидуальные задания

Во всех заданиях исходные данные вводить с помощью **ListBox**. Строки вводятся на этапе проектирования формы, используя окно свойств. Вывод результата организовать в метку **Label**.

1. Дана строка, состоящая из групп нулей и единиц. Посчитать количество нулей и единиц.
2. Посчитать в строке количество слов.
3. Найти количество знаков препинания в исходной строке.
4. Дана строка символов. Вывести на экран цифры, содержащиеся в строке.
5. Дана строка символов, состоящая из произвольных десятичных цифр, разделенных пробелами. Вывести вывести количество четных чисел в этой строке.
7. Дана строка символов. Вывести на экран количество строчных русских букв, входящих в эту строку.

8. Дана строка символов. Вывести на экран только строчные русские буквы, входящие в эту строку.

9. Дана строка символов, состоящая из произвольного текста на английском языке, слова разделены пробелами. В каждом слове заменить первую букву на прописную.

10. Дана строка символов, состоящая из произвольного текста на английском языке, слова разделены пробелами. Удалить первую букву в каждом слове.

11. Дана строка символов, состоящая из произвольного текста на английском языке, слова разделены пробелами. Поменять местами *i*- и *j*-ю буквы. Для ввода *i* и *j* на форме добавить свои поля ввода.

12. Дана строка символов, состоящая из произвольного текста на английском языке, слова разделены пробелами. Поменять местами первую и последнюю буквы каждого слова.

13. Дана строка символов, состоящая из произвольного текста на английском языке, слова разделены пробелами. Заменить все буквы латинского алфавита на знак '+’.

14. Дана строка символов, содержащая некоторый текст на русском языке. Заменить все большие буквы буквы 'А' на символ '*’.

15. Дана строка символов, содержащая некоторый текст. Разработать программу, которая определяет, является ли данный текст палиндромом, т.е. читается ли он слева направо так же, как и справа налево (например, «А роза упала на лапу Азора»).

16. Дана строка символов, состоящая из произвольного текста на английском языке, слова разделены пробелами. Сформировать новую строку, состоящую из чисел длин слов в исходной строке.