**ПРАКТИЧЕСКАЯ РАБОТА №7. ПРОГРАММИРОВАНИЕ С**

**ОДНОМЕРНЫМИ МАССИВАМИ**

*Цель лабораторной работы*: Написать приложение WPF для работы со одномерными массивами. В проект с ПР№1-6 добавить соответствующую страницу и переход на нее из главной страницы. Интерфейс разработать самостоятельно.

Массив - набор элементов одного и того же типа, объединенных общим именем. Массивы в С# можно использовать по аналогии с тем, как они используются в других языках программирования. Однако С#-массивы имеют существенные отличия: они относятся к ссылочным типам данных, более того - реализованы как объекты. Фактически имя массива является ссылкой на область кучи (динамической памяти), в которой последовательно размещается набор элементов определенного типа. Выделение памяти под элементы происходит на этапе инициализации массива. А за освобождением памяти следит система сборки мусора - неиспользуемые массивы автоматически утилизируются данной системой.

Рассмотрим в данной лабораторной работе одномерные массивы.

Одномерный массив - это фиксированное количество элементов одного и того же

типа, объединенных общим именем, где каждый элемент имеет свой номер.

Нумерация элементов массива в С# начинается с нуля, то есть, если массив

состоит из 10 элементов, то его элементы будут иметь следующие номера: 0, 1, 2,

3, 4, 5, 6, 7, 8, 9.

ПРАКТИЧЕСКАЯ РАБОТА

Одномерный массив в С# реализуется как объект, поэтому его создание

представляет собой двухступенчатый процесс. Сначала объявляется ссылочная

переменная на массив, затем выделяется память под требуемое количество

элементов базового типа, и ссылочной переменной присваивается адрес нулевого

элемента в массиве. Базовый тип определяет тип данных каждого элемента

массива. Количество элементов, которые будут храниться в массиве, определяется

размер массива.

В общем случае процесс объявления переменной типа массив, и выделение

необходимого объема памяти может быть разделено. Кроме того на этапе

объявления массива можно произвести его инициализацию. Поэтому для

объявления одномерного массива может использоваться одна из следующих форм

записи:

базовый\_тип [] имя\_\_массива;

Описана ссылка на одномерный массив, которая в дальнейшем может быть

использована для адресации на уже существующий массив. Например: int [] a;

базовый\_тип [] имя\_\_массива = new базовый\_тип [размер];

Объявлен одномерный массив заданного типа и выделена память под

одномерный массив указанной размерности. Адрес данной области памяти

записан в ссылочную переменную. Элементы массива равны нулю (В C#

элементам массива присваиваются начальные значения по умолчанию в

зависимости от базового типа. Для арифметических типов - нули, для ссылочных

типов - null, для символов - пробел). Например: int []a=new int [10];

базовый\_тип [] имя\_\_массива={список инициализации};

Выделена память под одномерный массив, размерность которого

соответствует количеству элементов в списке инициализации. Адрес этой области

памяти записан в ссылочную переменную. Значение элементов массива

соответствует списку инициализации. Например: int []a={0, 1, 2, 3};

Обращения к элементам массива происходит с помощью индекса, для этого

нужно указать имя массива и в квадратных скобках его номер. Например, a[0],

b[10], c[i].

Так как массив представляет собой набор элементов, объединенных общим

именем, то обработка массива обычно производится в цикле. Например:

int[] myArray = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 };

int i;

for (i = 0; i < 10; ++i)

ListBox.Items.Add(myArray[i]);

***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.

Варианты заданий







