**Лабораторная работа №2**

**Массивы одномерные и многомерные. Строки.**

**Массивы**

Базовый способ хранения информации в компьютерной программе – это хранение в переменной. Но бывают случаи, когда необходимо хранить набор одинаковых данных. К примеру, имена в телефонной книге, список покупок, какие-либо наборы чисел, координаты и т.п. Для таких целей можно использовать массивы.

**Массив** – это структурный тип данных, набор фиксированного числа элементов, имеющих один и тот же тип (базовый тип). Элементами массива могут быть данные любого, но только одного типа (как примитивного, так и объектного), т.е. может существовать массив целых чисел, массив вещественных чисел, массив строк, и т.д. Любой тип информации, который может храниться в переменной, можно хранить в массиве.

Число элементов массива фиксируется при объявлении и в процессе выполнения программы не меняется. Каждый отдельный элемент массива имеет свой номер (индекс), определяющий его положение в массиве и обеспечивающий доступ к элементу. Индексы представляют собой выражения любого целочисленного типа.

Массивы могут иметь одно или несколько измерений. Одномерный массив можно представить как последовательность однотипных объектов (вектор), а двумерный – как таблицу (матрицу) размера M\*N, где M - число строк, N - столбцов.

Также как при объявлении переменных, при объявлении массива указывается тип данных, которые будут храниться в массиве и его имя. Возле типа данных указывается пары квадратных скобок, чтобы отличать массивы от переменных. Общий формат объявления массива:

<тип>[] <имя массива>;

Здесь массивы объявляются, но в них ничего не хранится, и даже не выделена память для хранения элементов массива, то есть массив объявлен, но не инициализирован. Для выделения памяти необходимо использовать ключевое слово new, либо можно непосредственно указать элементы массива в фигурных скобках {}.

При использовании ключевого слова new необходимо еще раз указать тип данных элементов массива и указать их количество. Формат:

<имя массива> = new <тип>[<количество элементов>];

letters = new char[33];

Объявление и инициализацию массива можно объединить:

<тип>[] <имя массива> = new <тип>[<количество элементов>];

char[] letters = new char[33];

В этом примере создается массив символов (типа char) с названием letters. В массиве создано 33 элементов, в которых можно хранить буквы алфавита. Каждому элементу массива задано начальное значение. В данном случае, начальное значение будет ‘\0’. Массивы с любыми числовыми типами будут иметь начальное значение 0, а типа boolean – false.

Для не слишком больших массивов начальные данные можно задать с помощью скобок {}, но тогда делать это нужно при объявлении:

<тип>[] <имя массива> = {<элемент1>, <элемент2>, … <элементN>};

К примеру:

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

В этом случае количество элементов в массиве явно не указывается, а определяется количеством указанных элементов.

После того, как массив создан, в него нельзя добавлять больше указанного в размере количества элементов.

Элементы массива можно использовать точно так же, как любую переменную. Для того, чтобы получить элемент массива, нужно обратиться к нему по индексу, указав его в квадратных скобках после имени массива:

digits[3]

Элементы в массиве индексируются с нуля, поэтому индекс последнего элемента на 1 меньше количества элементов массива, указанного при объявлении.

Массив может состоять не только из элементов какого-то встроенного типа (int, double и пр.), но и, в том числе, из объектов какого-либо существующего класса и даже из других массивов.

Массив, который в качестве своих элементов содержит другие массивы, называется многомерным массивом.

Чаще всего используются двумерные массивы. Такие массивы можно легко представить в виде матрицы, каждая строка которой является обычным одномерным массивом, а объединение всех строк — двумерным массивом, в каждом элементе которого хранится ссылка на какую-то строку матрицы. Для того, чтобы обратиться к элементу 2-мерного массива нужно указать 2 индекса.

Объявляются массивы так:

int[] a1; // обычный, одномерный массив - вектор

int[,] a2; // двумерный массив

При инициализации массива можно указать явно размер каждого его уровня:

a2 = new int[3,4]; // матрица из 3 строк и 4 столбцов

Или можно объединить объявление и инициализацию:

int[,] a2 = new int[3,4];

Также можно сразу заполнить матрицу значениями при объявлении. Делается это аналогично одномерным массивам с помощью использования фигурных скобок:

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

В этом примере отчетливо видно, что мы создаем одномерный массив, элементами которого являются также одномерные массивы в скобках.

Для обращения к элементу двумерного массива указываем два индекса:

// обращение к элементу с индексом строки 0 и индексом столбца 1

a[0,1] = 100;

Можно указать только размер первого уровня:

// матрица из 5 строк. Сколько элементов в каждой строке пока неизвестно

int[,] b = new int[5,];

В последнем случае, можно создать двумерный массив, который не будет являться матрицей из-за того, что в каждой его строке будет разное количество элементов. К строкам матрицы можно обращаться как к одномерным массивам – для этого необходимо указать только первый индекс (индекс строки) двумерного массива.

Также можно создать массив, не являющийся матрицей, явно указав его элементы. Например, так:

int[,] с = {{1,2}, {1,2,3,4,5}, {1,2,3}};

При этом можно обратиться к элементу с индексом 4 во второй строке с[1][4], но если мы обратимся к элементу с[0][4] или с[2][4] — произойдёт ошибка выхода за границы массива, поскольку таких элементов просто нет. Притом ошибка это будет происходить уже во время исполнения программы (т. е. компилятор её не увидит). Обычно всё же используются двумерные массивы с равным количеством элементов в каждой строке, то есть матрицы.

**Строки**

Строковый тип данных string относится к числу самых важных в C#. Этот тип определяет и поддерживает символьные строки. В целом ряде других языков программирования строка представляет собой массив символов. А в C# строки являются объектами. Следовательно, тип string относится к числу ссылочных.

Строковые объекты являются неизменяемыми: их нельзя изменить после создания. Может показаться, что все методы String и операторы C# изменяют строку, но в действительности они возвращают результаты в новый строковый объект. Когда содержимое s1 и s2 объединяется для формирования одной строки, две исходные строки не изменяются, как показано в следующем примере. Оператор += создает новую строку, которая содержит объединенное содержимое. Этот новый объект присваивается переменной s1, а исходный объект, который был присвоен s1, освобождается для сборки мусора, так как ни одна переменная не ссылается на него.

string s1 = "A string is more ";

string s2 = "than the sum of its chars.";

s1 += s2;

System.Console.WriteLine(s1);

// Output: A string is more than the sum of its chars.

Так как "изменение" строки на самом деле является созданием новой строки, создавать ссылки на строки следует с осторожностью. Если вы создадите ссылку на строку, а затем "измените" исходную строку, ссылка будет по-прежнему указывать на исходный объект, а не на новый объект, который был создан при изменении строки. Это поведение проиллюстрировано в следующем коде:

string s1 = "Hello ";

string s2 = s1;

s1 += "World";

System.Console.WriteLine(s2);

//Output: Hello

Самый простой способ построить символьную строку — воспользоваться строковым литералом. Например, в следующей строке кода переменной ссылки на строку str присваивается ссылка на строковый литерал:

string str = "Пример строки";

В данном случае переменная str инициализируется последовательностью символов "Пример строки". Объект типа string можно также создать из массива типа char. Например:

char[] chararray = {'e', 'x', 'a', 'm', 'p', 'l', 'e'};

string str = new string(chararray);

Как только объект типа string будет создан, его можно использовать везде, где только требуется строка текста, заключенного в кавычки.

Типы String и StringBuilder

Хотя StringBuilder и String оба представляют последовательности символов, они реализуются по-разному. String является неизменяемым типом. То есть каждая операция, которая, в String , изменяет объект, на самом деле создает новую строку.

Для подпрограмм, выполняющих обширную обработку строк (например, приложения, которые изменяют строку несколько раз в цикле), многократное изменение строки может привести к значительному снижению производительности. Альтернативой является использование StringBuilder, которое является изменяемым строковым классом. Изменяемость означает, что после создания экземпляра класса его можно изменить путем добавления, удаления, замены или вставки символов. StringBuilder Объект поддерживает буфер для размещения расширения в строке. При наличии свободного места новые данные добавляются в буфер. в противном случае выделяется новый буфер большего размера, данные из исходного буфера копируются в новый буфер, а новые данные добавляются в новый буфер.

Хотя класс обычно обеспечивает лучшую производительность, не следует автоматически заменять String на StringBuilder, когда требуется управлять строками. Производительность StringBuilder зависит от размера строки, объема памяти, выделяемого для новой строки, системы, в которой выполняется приложение, и типа операции. Вы должны быть готовы к тестированию приложения, чтобы определить, предлагает ли StringBuilder значительное повышение производительности.

Принцип работы StringBuilder

Свойство указывает количество символов, которое в StringBuilder данный момент содержит объект. StringBuilder.Length При добавлении символов к StringBuilder объекту его длина увеличивается до тех пор, пока не будет равен размеру StringBuilder.Capacity свойства, определяющему количество символов, которое может содержать объект. Если число добавленных символов приводит к превышению размера StringBuilder объекта до его текущей емкости, выделяется новая память, значение Capacity свойства удваивается, в StringBuilder объект добавляются новые символы и его свойство Length регулируется. Дополнительная память для StringBuilder объекта выделяется динамически до тех пор, пока не достигнет значения, StringBuilder.MaxCapacity определенного свойством. По достижении максимальной емкости для StringBuilder объекта не может быть выделена память, и при попытке добавить символы или расширить ее после максимальной емкости будет создано ArgumentOutOfRangeException OutOfMemoryException исключение или.

Теперь посмотрим на примере использование и преимущества класса StringBuilder:

StringBuilder sb = new StringBuilder("Название: ");

Console.WriteLine("Длина строки: {0}", sb.Length); // 10

Console.WriteLine("Емкость строки: {0}", sb.Capacity); // 16

sb.Append(" Руководство");

Console.WriteLine("Длина строки: {0}", sb.Length); // 22

Console.WriteLine("Емкость строки: {0}", sb.Capacity); // 32

sb.Append(" по C#");

Console.WriteLine("Длина строки: {0}", sb.Length); // 28

Console.WriteLine("Емкость строки: {0}", sb.Capacity); // 32

При создании объекта StringBuilder выделяется память по умолчанию для 16 символов, так как длина начальной строки меньше 16.

Дальше применяется метод Append - этот метод добавляет к строке подстроку. Так как при объединении строк их общая длина - 22 символа - превышает начальную емкость в 16 символов, то начальная емкость удваивается - до 32 символов.

Если бы итоговая длина строки была бы больше 32 символов, то емкость расширялась бы до размера длины строки.

Далее опять применяется метод Append, однако финальная длина уже будет 28 символов, что меньше 32 символов, и дополнительная память не будет выделяться.

Простой пример, чтобы продемонстрировать разницу в скорости при использовании Stringконкатенации против StringBuilder:

System.Diagnostics.Stopwatch time = new Stopwatch();

string test = string.Empty;

time.Start();

for (int i = 0; i < 100000; i++)

{

test += i;

}

time.Stop();

System.Console.WriteLine("Using String concatenation: " + time.ElapsedMilliseconds + " milliseconds");

Результат:

Использование конкатенации строк: 15423 миллисекунд

StringBuilder test1 = new StringBuilder();

time.Reset();

time.Start();

for (int i = 0; i < 100000; i++)

{

test1.Append(i);

}

time.Stop();

System.Console.WriteLine("Using StringBuilder: " + time.ElapsedMilliseconds + " milliseconds");

Результат:

Использование StringBuilder: 10 миллисекунд

В результате первая итерация заняла 15423 мс, а вторая итерация с использованием StringBuilder10 мс.

В *приложении 1* приведены ссылки на описание базовых операций со строками и методами класса String.

**Варианты заданий**

## Задание 1

Вариант № 1 --------------------------------------------------------------------------------------------

1) Преобразовать заданный целочисленный массив S, уменьшив в два раза все

элементы кратные 2 ( если таковые есть ). Выдать сообщение о количестве

изменённых элементов.

2) Даны три последовательности чисел чисел A1,...,An; B1,...,Bn; C1,...,Cn;

Составить новую последовательность D1,...,Dn, каждый элемент которой

определяется по правилу : Di = MAX(Ai,Bi,Ci), i=1..9.

3) Найти номер первого нулевого элемента массива A1,...,An и произведение

элементов, расположенных до него, а среди элементов, расположенных правее

первого нулевого, найти максимальный элемент.

Вариант № 2 ----------------------------------------------------------------------------------------------

1) В массиве D1...Dn найти сумму элементов, расположенных на четных местах.

2) Найти номер первого нулевого элемента массива X1,...,Xn и сумму элементов,

предшествующих ему.

3) В данном массиве чисел C1,...,Cn поменять местами максимальный элемент с

последним отрицательным.

Вариант № 3) -----------------------------------------------------------------------------------------------

1) В массиве чисел C1,...,Cn найти произведение отрицательных чисел.

2) Даны три последовательности чисел A1,...,An ; B1,...,Bn ; C1,...,Cn .

Составить новую последовательность, в которой чередовались бы числа всех

трех последовательностей : D1=A1; D2=B1; D3=C1; D4=A2; ... D(3n)=Cn

3) Из массива X1,...,Xn сформировать два массива : в один записать числа, располо-

женные до минимального элемента ,во второй числа, расположенные после мини-

мального элемента.

Вариант № 4 ) ----------------------------------------------------------------------------------------------

1) В массиве А1, ... , Аn найти количество элементов равных единице, стоящих на чётных местах.

2) Известно, что в массиве А1,А2,...,Аn количество отрицательных чисел равно

количству положительных. Составить новый массив так, чтобы чередовались

положительные и отрицательные числа.

3) В массиве A1, A2, ... , An найти максимальный элемент и его местоположение в

массиве.

Вариант № 5 ) ----------------------------------------------------------------------------------------------

1) Дана последовательность А1, ... , Аn. Заменить на нули все элементы,

расположенные между максимальным и минимальным элементом.

2) Найти сумму элементов последовательности B1, B2, ... , Bn, расположенных

правее последнего отрицательного элемента, и номер этого элемента.

3) Составить новый массив, состоящий из пяти последних положительных элементов

последовательности Y1,Y2,...Yn домноженных на номер максимального элемента

данной последовательности.

Вариант № 6 ) ----------------------------------------------------------------------------------------------

1) В массиве А1, ... , Аn найти минимум среди положительных и максимум среди отрица-

тельных элементов .

2) Дан массив целых чисел X1..Xn, в котором есть одна группа одинаковых элементов,

расположенных подряд. Подсчитать количество элементов в этой группе.

3) Составить новый массив, состоящий из элементов исходного вектора, значения которых

совпадают с их индексами.

Вариант № 7 ) ----------------------------------------------------------------------------------------------

1) В массиве С1, С2, ... , Сn найти сумму чисел, больших единицы.

2) По вектору С ( С1, ... , Сn ) получить вектор Х ( Х1, ... , Хn ) по правилу :

Х1=С1; Х2=С3; ... ; Х(n/2)=С(n-1), Х(n/2+1)=Сn, Х(n/2+2)=С(n-2), ... , Хn=С2.

Первая половина - нечетные , вторая - четные элементы исходного в обратном

порядке. Если размерность C - нечетная, то середина : C( trunc(n/2))

3) Дана последовательность чисел В1, В2, ... , Вn. Найти сумму S1 элементов до

максимального элемента и сумму S2 элементов, расположенных правее него.

Вариант № 8 ) ----------------------------------------------------------------------------------------------

1) В массиве D1, D2, ... , Dn найти количество чисел, меньших единицы.

2) Из данного массива чисел Х1, Х2, ..., Хn исключить первое отрицательное

число. Оставшиеся числа переписать в массив Y1, Y2, ... , Y(n-1).

3) В данном массиве чисел А1, А2, ... , Аn поменять местами минимальный и

максимальный элементы.

Вариант № 9 ) ----------------------------------------------------------------------------------------------

1) В массиве С1, С2, ..., Сn найти произведение чисел, больших 2.

2) Из данного массива чисел Х1, ... , Хn исключить последнее положительное

число. Оставшиеся числа переписать в массив Z1, ... , Z(n-1).

3) Найти сумму положительных элементов последовательности D1, ... , Dn ,

расположенных до первого нулевого элемента, заменить этой суммой

минимальный элемент массива.

Вариант № 10 ) ----------------------------------------------------------------------------------------------

1) В массиве А1, ..., Аn найти сумму чисел, расположенных на местах, кратных 3.

2) Найти номер первого положительного элемента массива В1, ... ,Вn и сумму

элементов, расположенных правее него.

3) Из отрицательных элементов массива Х1, Х2 , ... , Хn, расположенных левее

минимального элемента, сформировать новый массив.

## Задание 2

Вариант № 1

-----------------------------------------------------------------------

1. Столбцы целочисленной матрицы M[4,4] содержат как четные, так и не-

четные элементы. Поменять местами 1-й столбец со столбцом, в котором

будет минимальная сумма чётных элементов.

-----------------------------------------------------------------------

2. Значения элементов нечётных строк матрицы M[m,n] заменить на разность

номеров строки и столбца

-----------------------------------------------------------------------

Вариант № 2

1. Разработать программу, которая в матрице размерности M\*N меняет

местами строку, содержащую элемент с минимальным значением со

строкой, содержащей элемент с максимальным значением.

-----------------------------------------------------------------------

2. В матрице Z[m,n] определить число положительных элементов и их сумму

по столбцам.

-----------------------------------------------------------------------

Вариант № 3

1. Найти сумму элементов главной диагонали квадратной матрицы и

сравнить с суммой элементов первой строки.

-----------------------------------------------------------------------

2. Разработать программу нахождения местоположения(номер строки и столб-

ца) второго по величине максимального элемента матрицы(M\*N).

-----------------------------------------------------------------------

Вариант № 4

1. Преобразовать квадратную матрицу, поменяв местами столбец с наибольшим

количеством элементов кратных 3 со столбцом, в котором сумма четных

элементов минимальная.

-----------------------------------------------------------------------

2. Разработать программу подсчета количества отрицательных элементов

целочисленной матрицы A[M\*N], стоящих на пересечении четных строк и

столбцов.

-----------------------------------------------------------------------

Вариант № 5

1. Заменить максимальный элемент главной диагонали на среднюю сумму положительных

элементов матрицы P[n,n].

-----------------------------------------------------------------------

2. В матрицах A[m,n], B[m,n] найти и поменять местами минимальный элемент

среди чётных столбцов и нечётных строк соответственно

-----------------------------------------------------------------------

Вариант № 6

1. В матрице Q[m,n] переставить местами 2 столбца, где минимальная сум-

ма и максимальное произведение элементов.

------------------------------------------------------------------------

2. Дана матрица A[m,n] и вектор E[n]. Сформировть новую матрицу

A1[m+1,n], которая состоит из матрицы A и включения вектора E после

строки, в которой находится максимальное количество 0-вых элементов.

------------------------------------------------------------------------

Вариант № 7

1. В матрице А[n,n] определить масимальный среди отрицательных и

минимальный среди положительных элементов побочной диагонали.

Найти среднее произведение и среднеарифметическую сумму положительных

элементов главной диагонали.

-----------------------------------------------------------------------

2. Дан вектор E(n) и матрица A[m,n].Найти местоположение и значение

максимального элемента матрицы и поменять местами элементы строки,

где он расположен, с соответствующими элементами массива E.

-----------------------------------------------------------------------

Вариант № 8

1. В матрице A[m,n] заменить все отрицательные числа на максимальный

элемент.

-----------------------------------------------------------------------

2. В матрице A[m,n] найти минимальный среди положительных элементов и

заменить на него все чётные элементы чётных столбцов матрицы.

----------------------------------------------------------------------

Вариант № 9

1. Образовать одномерный массив из среднеарифметических значений

элементов матрицы A[m,n] по столбцам.

-----------------------------------------------------------------------

2. В матрице M[n,n] и одномерном массиве L(n) поменять местами элементы

главной диагонали с соответствующими элементами вектора L, при этом

сменив знаки на противоположные.

------------------------------------------------------------------------

Вариант № 10

1. В матрице A[m,n] определить номер строки с наибольшим количеством

положительных элементов.

------------------------------------------------------------------------

2. В матрице A[m,n] определить число чётных и нечётных элементов до пер-

вого нуля, считая по строкам.

------------------------------------------------------------------------

## Задание 3.1

Вариант №1

Из двух предложений удалить слова, встречающиеся в обоих предложениях. Вывести на экран полученные предложения и удаленные слова.

------------------------------------------------------------------------

Вариант №2

Расположить слова данного предложения в порядке убывания длин.

------------------------------------------------------------------------

Вариант №3

Найти слова, имеющие наибольшее число вхождений в предложении.

------------------------------------------------------------------------

Вариант №4

Даны два предложения, причем второе состоит из слов первого, записанных в другом порядке. Найти этот порядок.

------------------------------------------------------------------------

Вариант №5

Из заданного текста выбрать и напечатать те слова, которые встречаются в нем ровно один раз.

------------------------------------------------------------------------

Вариант №6

Записать слова предложения в обратном порядке, исключив запятые «,».

------------------------------------------------------------------------

Вариант №7

В заданном предложении имеются два слова, одно из которых является обращением другого (т.е. "перевертышем" -> «нос» - "сон»). Найти эту пару слов.

------------------------------------------------------------------------

Вариант №8

Найти множество всех слов, которые встречаются в каждом из двух заданных предложений.

------------------------------------------------------------------------

Вариант №9

Для каждого слова заданного текста указать, сколько раз оно встречается в тексте. Сообщение об одном слове должно печататься не более одного раза.

------------------------------------------------------------------------

Вариант №10

В предложении поменять местами слова с максимальной и минимальной длинами.

## Задание 3.2

Написать программу, реализующую следующие функции работы с массивом строк (путем ввода соответствующих команд в консоли):

1. Создание строки и добавление её строки в массив на заданную позицию i (с соответствующим сдвигом всех следующих за i элементов массива).
2. Вывод всех строк массива на экран.
3. Вывод i строки массива на экран.
4. Редактирование i строки массива (добавление нового слова, замена существующего слова, удаление слова).
5. Замена слова в каждой строке массива.
6. Удаление строки из массива.

Задание выполнить, используя методы класса StringBuilder.

Приложение 1

Справочные материалы

Одномерные массивы

<https://docs.microsoft.com/ru-ru/dotnet/csharp/programming-guide/arrays/single-dimensional-arrays>

Многомерные массивы

<https://docs.microsoft.com/ru-ru/dotnet/csharp/programming-guide/arrays/multidimensional-arrays>

foreach при работе с массивами

<https://docs.microsoft.com/ru-ru/dotnet/csharp/programming-guide/arrays/using-foreach-with-arrays>

Строки

<https://docs.microsoft.com/ru-ru/dotnet/csharp/programming-guide/strings/>

Базовые операции со строками

<https://docs.microsoft.com/ru-ru/dotnet/standard/base-types/basic-string-operations>

Методы String

<https://msdn.microsoft.com/ru-ru/library/system.string_methods%28v=vs.110%29.aspx?f=255&MSPPError=-2147217396>