

М А С С И В Ы

Статические и динамические



Статические массивы

Массив - последовательный список определенного количества элементов. Все элементы в массиве принадлежат к одному типу данных.

Формат объявления массива:

array [indexType1, ...,indexTypen] **of** baseType

indexType - порядковый тип, размерность которого не превосходит 2GB. Обычно задается **диапазоном целых чисел.**



Статические массивы

Количество элементов массива равно произведению количеств элементов во всех измерениях

baseType - тип элементов массива

ПР: Type TArr=array[1..10, 3..6] of integer;
Var A,B: TArr
// В массивах A и B по 40 элементов

Массив-переменная

Массив-константа

Массив можно объявить сразу в разделе `var`

ПР: `Var A,B: array[1..10, 3..6] of integer;`

В разделе `const` можно определить массив-константу

`Type TStates = array[1..4] of string;`

`TMatr = array[1..2,1..3] of Integer;`

`Const States: TStates = ('Волгоград', 'Воронеж', 'Москва', 'Пермь');`

`Const Matr: TMatr = ((10, 20, 5), (3, 15, 9));`

Массив-переменная

Массив-константа

States – **одномерный** массив

Matr – **двумерный** массив

States[3] – элемент массива States
с индексом 3

Matr[1,2] – элемент массива Matr с
индексами 1, 2

Matr[1][2] – тот же элемент
массива Matr с индексами 1, 2



Работа с одномерными массивами

Присваивание

ПР: var A, B: array[1..10] of Integer;
begin A := B; end.

Ввод с клавиатуры и суммирование

ПР: var A: array[1..5] of Double;
Sum: Double; I: Integer;
begin Sum := 0;
for I := 1 to 5 do
begin ReadLn(A[I]);
Sum := Sum + A[I];
end; WriteLn('Sum = ', Sum);
end.



Работа с одномерными массивами

Функция **Low(A)** возвращает **нижнюю** границу массива A

Функция **High(A)** возвращает **верхнюю** границу массива A

Вывод элементов массива на экран

ПР:

```
var I: Integer;  
const A:array[1..5] of integer = (-3,4,5,-1,8);  
begin  
    for I := Low(A) to High(A) do  
        Write(A[I]:4);  
end;
```



Работа с двумерными массивами

Генерирование и нахождение произведения элементов

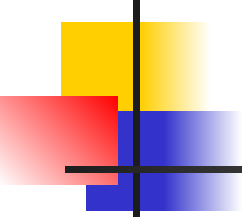
```
ПР: var A: array[1..2,1..3] of integer;  
      I,J: integer;  
      Prod: Longint;  
begin Randomize; Prod := 1;  
  for I := 1 to 2 do  
    for J := 1 to 3 do  
      begin  
        A[I,J]:=1+Random(3);  
        Prod := Prod*A[I,J]  
      end;  
    end.  
end.
```




Работа с двумерными массивами

Вывод элементов массива A на экран (в виде матрицы)

```
ПР:  for I := 1 to 2 do
      begin
        for J := 1 to 3 do
          Write(A[I,J]:4);
        WriteLn;
      end;
```



Массивы в параметрах процедур и функций

```
const Max = 63;  
type TStatistics = array [0..Max] of Double;  
function Average(const A: TStatistics): Double;  
  var I: Integer;  
  begin  
    Result := 0;  
    for I := Low(A) to High(A) do Result := Result + A[I];  
    Result := Result / (High(A) - Low(A) + 1);  
  end;
```

Параметр функции Average - массив известной размерности, тип которого задан в разделе **type**.



Открытые массивы-параметры

При описании открытого массива-параметра **границы массива опускаются**

ПР: function Average (const A: **array of Double**):
Double;

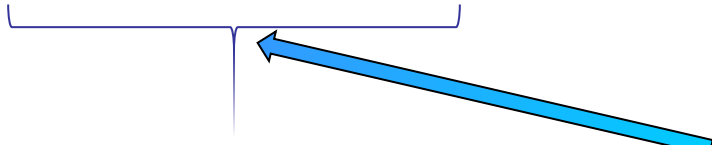
Внутри функции Average нижняя граница A равна нулю (**Low(A) = 0**).

Значение верхней границы (**High(A)**)
выясняется только при выполнении программы.

Открытые массивы-параметры

Пример использования функции Average:

```
var Statistics: array[1..10] of Double; Mean: Double;  
Begin . . . . .  
    Mean := Average(Statistics);  
    Mean := Average([0, Random, 1]); . . . . .  
end;
```



Конструктор открытого массива - **заклученный в квадратные скобки список выражений**

Открытые массивы передаются в подпрограммы только **по значению** или как **параметры-константы**



Динамические массивы

Динамический массив **объявляется без указания границ**

ПР: var DynArray: array of Integer;

Работа с динамическими массивами:

1. **DynArray** - ссылка на размещаемые в динамической памяти элементы массива
2. Изначально **память** под массив **не резервируется**
3. **SetLength** - процедура создания динамического массива (выделения памяти для его элементов)
ПР: SetLength(DynArray, 50); // 50 элементов



Динамические массивы

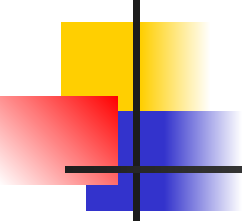
Работа с динамическими массивами:

4. Функция **Length** возвращает количества элементов
(равно $\text{High}(\text{DynArray}) + 1$)
5. Элементы динамического массива индексируются **от нуля** ($\text{Low}(\text{DynArray}) = 0$)
6. Освобождение памяти от динамического массива: **установить длину 0** или присвоить переменной-массиву значение **nil**



Динамические массивы

```
Var DynArray: array of Integer; K, N: integer; S: real;
Begin
    Randomize;
    Write('Введите количество элементов: '); ReadLn(N);
    SetLength(DynArray,N);
    S:=0;
    For K:=Low(DynArray) to High(DynArray) do
        begin DynArray[K]:=-5+Random(10);
            S:=S+DynArray[K]
        end;
    S:=S/Length(DynArray); Write(S);
end.
```



Двумерные динамические массивы

```
Var A: array of array of Integer;
```

```
.....
```

```
Begin
```

```
.....
```

```
  SetLength(A, 100); // 100 элементов
```

```
  for I := Low(A) to High(A) do SetLength(A[I], I);  
  // Память выделяется для каждой размерности
```

```
.....
```

```
End.
```