Массивы

Пример объявления массива

Статический массив – упорядоченная последовательность фиксированного количества переменных одного типа, имеющая общее имя.

Описание массива:

<идентификатор>: **array** [<диапазон индексов>] **of** <тип элементов>;

Пример объявления массива 10-ти целых чисел.

```
Var
    a : array [1..10] of integer;
```

i	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
A[i]	284	345	91	-34	456	3	45	-456	28	23

индекс

элемент

Объявление массивов с использованием раздела описания типов

Пример объявления массива :

```
Var
      a : array [1..50] of real;
      b,c : array [1..20] of integer;
...
```

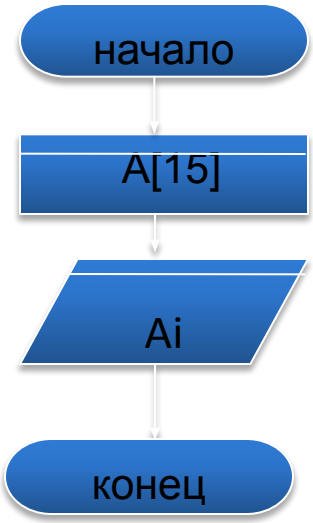
Аналогичное описание массивов с использованием раздела описания типов:

```
Type
    mas1=array[1..50] of real;
    mas2=array[1..20] of integer;
Var
    a : mas1;
    b,c : mas2;
...
```

Ввод – вывод элементов массива

Генерация элементов массива случайными числами.

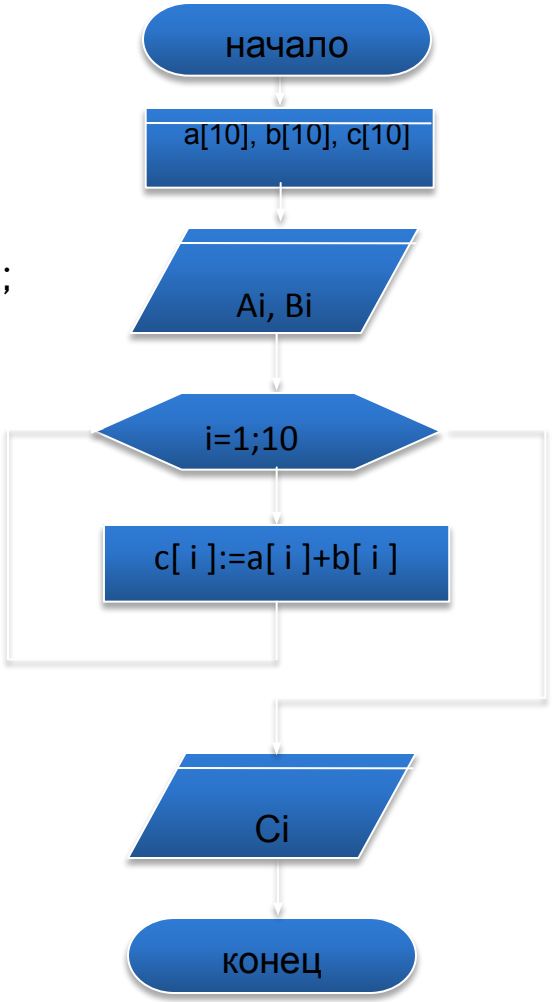
random(n) – функция генерации случайного числа в диапазоне от 0 до n-1.




```
Var
        a : array [1..15] of integer;
        i : integer;
Begin
randomize;
writeln ('Элементы целочисленного массива A[15] сформированные случайными числами диапазона от -100 до 100.');
for i:=1 to 15 do
    begin
    a[ i ]:=random(201)-100;
    write (a[ i ]:6);
    end
End.
```

Суммирование двух одномерных массивов

```
Type
     massiv=array[1..10] of integer;
Var
          a , b, c: massiv;
          i : integer;
Begin
randomize;
writeln (‘ Массив A ‘);
for i:=1 to 10 do
     begin
     a[ i ]:=random(51);
     write (a[ i ]:5);
     end;
writeln (‘ Массив B ‘);
for i:=1 to 10 do
          begin
          b[ i ]:=random(151)-70;
          write (b[ i ]:5);
          end;
for i:=1 to 10 do
          c[ i ]:=a[ i ]+b[ i ];
writeln (‘ Массив C ‘);
for i:=1 to 10 do
           write (c[ i ]:6);
End.
```



Нахождение индексов элементов с заданным свойством

Рассмотрим задачу Нахождения и вывода на экран номеров (индексов) четных элементов.

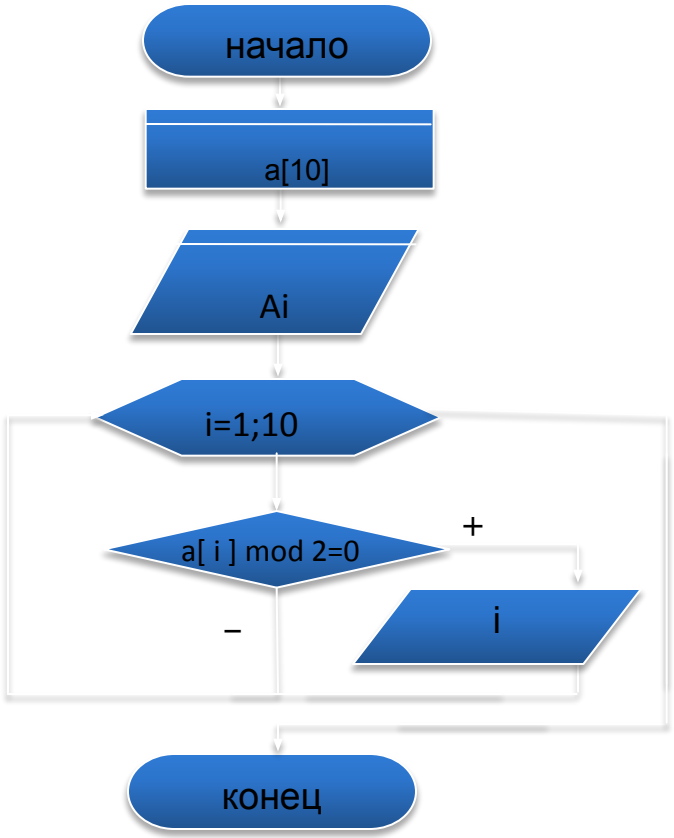
Для решения задачи необходимо просмотреть весь массив, и если просматриваемый элемент является четным, то выводить его индекс.

i	1	2	3	4	5	6	7	8	9	10
Ai	2	10	15	23	7	19	44	3	1	24

1	2	7	10

Нахождение индексов элементов с заданным свойством

```
Const
     n=10;
Type
     massiv=array[1..n] of integer;
Var
          a : massiv;
          i: integer;
Begin
writeln (‘ Введите элементы
массива A ‘);
for i:=1 to n do
     begin
     write (‘a[‘, i , ‘] =‘);
     read (a[ i ]);
     end;
for i:=1 to n do
     if a[ i ] mod 2=0 then
       write (i:4);
End.
```



Поиск минимального и максимального элементов одномерного массива

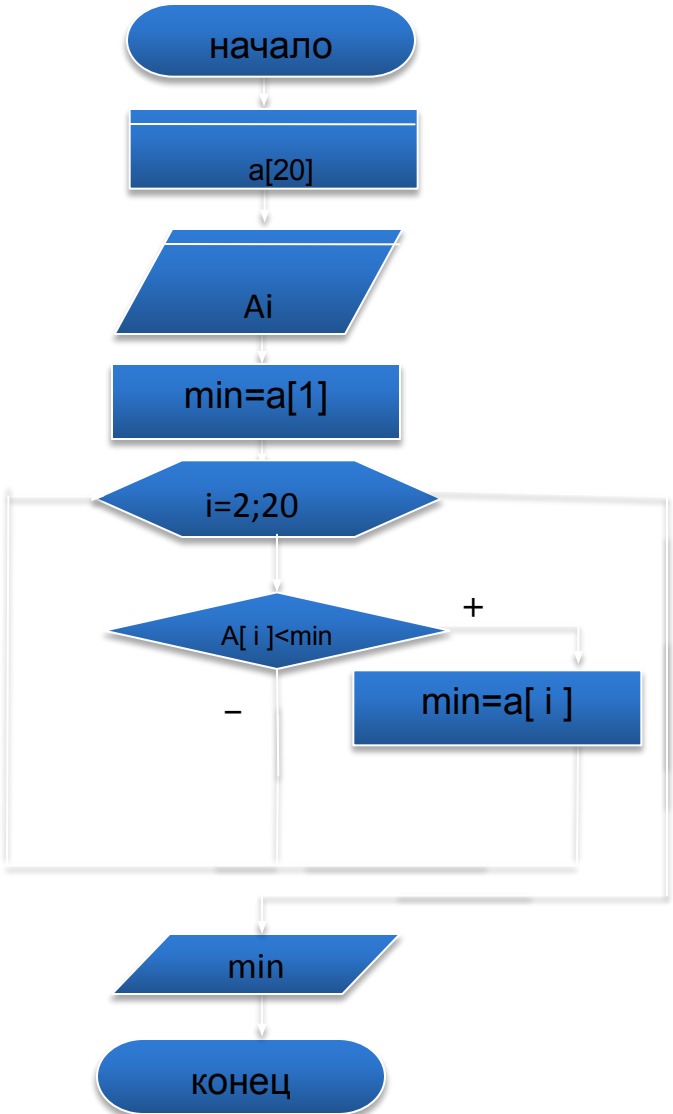
Поиск максимального и минимального элементов массива относится к классическим задачам обработки данных с использованием массива. Суть алгоритма поиска минимального элемента состоит в том, что предположительно за минимальный объявляют первый элемент массива и перебирая все элементы изменяют значение минимального элемента текущим, в том случае, если он оказался меньше минимального на данном этапе. Задача нахождения максимального элемента имеет подобное тривиальное решение.

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ai	1	0	-5	4	12	3	30	-2	16	45	11	-7	0	23	0	0	12	26	1	1

Min
-7

Поиск минимального элемента одномерного массива

```
Type
     mas=array[1..20] of integer;
Var
          a : mas;
          min, i : integer;
Begin
randomize;
writeln (‘ Массив  ‘);
for i:=1 to 10 do
     begin
     a[ i ]:=random(101)-50;
     write (a[ i ]:6);
     end;
min:=a[1];
for i:=2 to 20 do
      if  a[ i ] < min then
               min:=a[ i ];
writeln (min)
End.
```



Удаление элементов массива

Удалить элемент в статическом массиве – невозможно. Поэтому используют перемещение всех элементов, начиная с "удаляемого", записывая на их место следующие (i+1) элементы. Вводят так же переменную, которая обозначает индекс последнего элемента и при каждом шаге удаления элемента ее уменьшают на 1. Рассмотрим задачу на удаление всех отрицательных элементов массива.

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ai	1	0	-5	4	12	3	3.2	-2	16	45	1.1	-7	-1	23	0	0	12	2.6	-1.3	1

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Ai	1	0	4	12	3	3.2	16	45	1.1	23	0	0	12	2.6	1

Удаление элементов массива

```
Type
     mass=array[1..20] of real;
Var
          a: mass;
          i, j, m : integer;
Begin
randomize;
writeln (‘ Массив А ‘);
for i:=1 to 20 do
     begin
     a[ i ]:=(random(201)-80)/(random(100)+1);
     write (a[ i ]:6:2);
     end;
m:=20;
for i:=1 to 20 do
     begin
     if a[ i ]<0 then
          begin
          for j:=i to 20 do
               a[ j ]=a[ j+1 ];
          dec(m)
          end;
          if a[ i ]<0 then
               dec(i)
     end;
writeln (‘ Массив А без отрицательных элементов ‘);
for i:=1 to m do
      write (a[ i ]:6:2)
End.
```

Вызов:	dec(x,n)	Значение функции: как x
Параметры:	x:перечисляемый тип; n:integer;	
Действие:	Значением функции является значение x, уменьшенное на n. Если n отсутствует, x уменьшается на 1 (x :=x-1, образование отрицательного приращения)	

Включение элементов массива

Включить элемент в статический массив так же невозможно. Поэтому изначально размер массива должен быть больше на количество предполагаемых элементов для включения в массив. При включении элемента следует в цикле перебирать элементы от последнего элемента до индекса, куда будет включен элемент и переписывать значения текущего (i-го) элемента на место последующего (i+1). Следует так же ввести переменную для хранения индекса последнего элемента, которую при каждом включении увеличивают на 1. Рассмотрим задачу на включение значения T в массив, которое должно располагаться за максимальным элементом массива.

T
5

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ai	1	0	-5	4	1	3	3	-2	16	45	1.1	-7	-1	23	0	0	12	2.6	-1.3	1

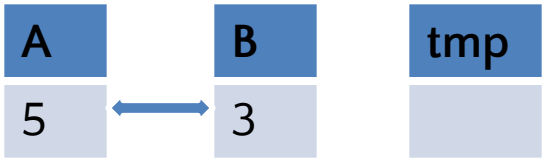
i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Ai	1	0	-5	4	1	3	3	-2	16	45	5	1.1	-7	-1	23	0	0	12	2.6	-1.3	1

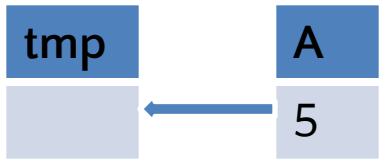
Включение элементов массива

```
Type
    mass=array[1..21] of real;
Var
        a: mass;
        i, j , i_max : integer;
        t, max : real;
Begin
randomize;
writeln (‘ Массив А ‘);
for i:=1 to 20 do
    begin
    a[ i ]:=(random(201)-80)/(random(100)+1);
    write (a[ i ]:6:2);
    end;
readln (t);
max:=a[1];
i_max:=1;
for i:=2 to 20 do
    if a[ i ]>max then
        begin
        max:=a[ i ];
        i_max:=i
        end;
for j:=21 downto i_max-1 do
        a[ j ]=a[ j-1 ];
a[i_max]=t;
writeln (‘ Массив А с включенным
элементом t ‘);
for i:=1 to 21 do
        write (a[ i ]:6:2)
End.
```

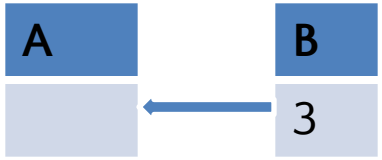
Перестановка элементов массива

Алгоритм перестановки элементов (обмена значениями) прост. Для его выполнения достаточно воспользоваться "временной" переменной, в которую сначала помещают значение первой переменной. Затем в первую переменную заносят значение второй (если не воспользоваться "временной" – значение первой переменной будет потеряно). И сохраненное значение первой переменной во "временной" заносят во вторую переменную. Эту операцию образно можно сравнить с операцией по переливанию двух разных жидкостей из двух пробирок, воспользовавшись третьей – пустой пробиркой.

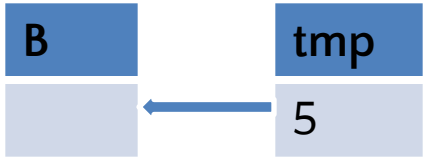




tmp:=a;



a:=b;



b:=tmp;

Перестановка элементов массива

Рассмотрим задачу обмена максимального и минимального элементов местами.

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ai	1	0	-5	4	1	3	3	-2	16	45	1.1	-7	-1	23	0	0	12	2.6	-1.3	1

max	i_max	min	i_min
45	10	-7	12

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ai	1	0	-5	4	1	3	3	-2	16	-7	1.1	45	-1	23	0	0	12	2.6	-1.3	1

Инвертирование массива

Инвертирование массива – это запись его элементов в обратном порядке. Для решения этой задачи можно воспользоваться другим массивом, в который можно записать элементы из данного массива в обратном порядке. Однако целесообразнее сделать это за наименьшее количество перестановок и не использовать дополнительного массива. Как это сделать? Можно двигаться от первого элемента до середины массива и менять местами первый элемент с последним, второй – с предпоследним и т.д. Получается что в цикле будут обмениваться элемент с i-м индексом с элементом у которого индекс равен n-i+1 , где n- индекс последнего элемента.

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ai	1	0	-5	4	1	3	3	-2	16	45	1.1	-7	-1	23	0	0	12	2.6	-1.3	1

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ai	1	-1.3	2.6	12	0	0	23	-1	-7	1.1	45	16	-2	3	3	1	4	-5	0	1

Инвертирование массива

```
Var
        a: array [1..20] of real;
        i , n : integer;
        tmp : real;
Begin
randomize;
writeln (‘ Массив А ‘);
n:=20;
for i:=1 to n do
    begin
    a[ i ]:=(random(201)-80)/(random(100)+1);
    write (a[ i ]:6:2);
    end;
for i:=1 to n div 2 do
  begin
    tmp:=a[ i ];
    a[ i ]:=a[ n-i+1 ];
    a[ n-i+1 ]:=tmp;
  end;
writeln (‘ Инвертированный массив А ‘);
for i:=1 to n do
        write (a[ i ]:6:2)
End.
```

Сортировка массива

```
{ сортировка массива "пузырьком" по возрастанию }
const
  n = 10; { количество элементов в массиве }
var
  a:array[1..n] of integer;
  i,j,buf:integer;
begin

  for i:=1 to n do
    begin
      a[i]:=random(10);
      write(a[i],' ');
    end;
  for i:=1 to n-1 do
    for j:=i+1 to n do
      if a[i]>a[j] then
        begin
          buf:=a[i];
          a[i]:=a[j];
          a[j]:=buf;
        end;
  writeln;
  writeln('Массив после сортировки пузырьковым методом: ');
  for i:=1 to n do
    write(a[i],' ');
end.
```

При сортировке массива методом пузырька, сравниваются два соседних элемента массива. В том случае, если элемент массива с номером iоказывается больше элемента массива с номером i+1, происходит обмен значениями при помощи вспомогательной переменной buf

Двумерный массив

При решении практических задач часто приходится иметь дело с различными таблицами данных, математическим эквивалентом которых служат матрицы. Такой способ организации данных, при котором каждый элемент определяется номером строки и номером столбца, на пересечении которых он расположен, называется ***двумерным массивом (матрицей)*** или ***таблицей***.

i\j	1	2	…	m-1	m
1	A[1,1]	A[1,2]			A[1,m]
2	A[2,1]	A[2,2]			
…			A[i,j]		
n-1					
n	A[n,1]				A[n,m]

Массив из N строк и M столбцов

```
Const
      n=6;
      m=8;
Var     a : array [1..n, 1..m] of integer;
```

Квадратные матрицы

В квадратной матрице количество строк и столбцов – **одинаково и равно** n.

Любая квадратная матрица имеет элементы **главной и побочной диагонали**.

Диагональные элементы главной диагонали :

a[1 , 1];a[2 , 2];a[3 , 3];….;a[n , n].

Элементами побочной диагонали являются :

a[1 , n];a[2 , n–1];a[3 , n–2];…;a[n–1 , 2];a[n , 1].

Квадратные матрицы

В качестве примера рассмотрим задачу формирования квадратной матрицы порядка n случайными числами и нахождения произведения элементов главной диагонали и суммы элементов ниже побочной диагонали.

Отметим элементы главной диагонали для нахождения их произведения.

i\j	1	2	3	…	n-1	n
1	A[1,1]	A[1,2]	A[1,3]			A[1,n]
2	A[2,1]	A[2,2]	A[2,3]			
3	A[3,1]	A[3,2]	A[3,3]			
…				A[i, j]		
n-1					. . .	
n	A[n,1]					A[n,n]

Квадратные матрицы

И элементы ниже побочной диагонали для поиска их суммы.

i\j	1	2	3	…	n-1	n
1	A[1,1]	A[1,2]	A[1,3]			A[1,n]
2	A[2,1]	A[2,2]	A[2,3]			
3	A[3,1]	A[3,2]	A[3,3]			
…				A[i, j]		
n-1					. . .	
n	A[n,1]					A[n,n]

Квадратные матрицы

```
Const
        n=9;
Var
        a : array [1..n, 1..n] of integer;
        I, j, s, p : integer;
Begin
randomize;
for i:=1 to n do
        begin
        for  j:=1 to n do
                begin
                a[ i, j ]:=random(101);
                write (a[ i , j ]:6);
                end;
        writeln;
        end
p:=1;
for i:=1 to n do
        p:=p*a[ i, i ];
s:=0;
for i:=2 to n do
        for j:=n-i+2 to n do
                s:=s+a[ i, j ];
writeln (p,s);

End.
```

Транспонирование матриц

В данном алгоритме **транспонирования матрицы** необходимо заменить строки матрицы ее столбцами, а столбцы – строками, т.е. вычислить

b[i,j] :=a[j,i], где i=1,…,n; j=1,…,m.

Матрица A

	1	2	3	4
1	0	5	-7	3
2	23	-45	90	5
3	102	22	-45	21
4	-4	-8	0	34
5	64	4	5	7
6	10	-45	-37	-23
7	-45	0	-3	1

Матрица B

	1	2	3	4	5	6	7
1	0	23	102	-4	64	10	-45
2	5	-45	22	-8	4	-45	0
3	-7	90	-45	0	5	-37	-3
4	3	5	21	34	7	-23	1

Транспонирование матриц

```
Const
      n=5;
      m=7;
Var
i, j : integer;
a : array [1..n,1..m] of integer;
b : array [1..m,1..n] of integer;
Begin
randomize;
writeln (‘Сформирована матрица A’);
for i:=1 to n do
      begin
      for  j:=1 to m do
        begin
        a[ i,j ]:=random(31)-15;
        write (a[ i,j ]:6);
        end;
      writeln(‘’);
end;
```

```
for i:=1 to n do
      for j:=1 to m do
        b[ j,i ]:=a[ i,j ];
writeln (‘Получена транспонированная
      матрица B’);
for i:=1 to m do
      begin
      for j:=1 to n do
        write(b[ i,j ]:6);
      writeln(‘’);
end;
End.
```

Умножение матрицы на вектор

Для вычисления произведения C матрицы A размером n*m на вектор B размером m необходимо вычислить , i=1,….,n.

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \\ a_{51} & a_{52} & a_{53} & a_{54} \end{pmatrix} \cdot \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{pmatrix} = \begin{pmatrix} a_{11} \cdot b_1 + a_{12} \cdot b_2 + a_{13} \cdot b_3 + a_{14} \cdot b_4 \\ a_{21} \cdot b_1 + a_{22} \cdot b_2 + a_{23} \cdot b_3 + a_{24} \cdot b_4 \\ a_{31} \cdot b_1 + a_{32} \cdot b_2 + a_{33} \cdot b_3 + a_{34} \cdot b_4 \\ a_{41} \cdot b_1 + a_{42} \cdot b_2 + a_{43} \cdot b_3 + a_{44} \cdot b_4 \\ a_{51} \cdot b_1 + a_{52} \cdot b_2 + a_{53} \cdot b_3 + a_{54} \cdot b_4 \end{pmatrix}$$

Использование вспомогательной переменной s позволяет уменьшить время выполнения программы за счет исключения обращения в цикле по j к элементам массива C.

Умножение матрицы на вектор

```
Const
      n=6;
      m=9;
Var
i, j, s : integer;
a : array [1..n,1..m] of integer;
b : array [1..m] of integer;
c: array [1..n] of integer;
Begin
randomize;
writeln ('Сформирована матрица
      A');
for i:=1 to n do
      begin
      for  j:=1 to m do
        begin
        a[ i,j ]:=random(101)-50;
        write (a[ i,j ]:6);
                end;
        writeln('');
        end;
writeln ('Сформирован вектор B');
for j:=1 to m do
        begin
        b[ j ]:=random(51)-30;
        write (b[ j ]:6);
        end;
for i:=1 to n do
        begin
        s:=0;
        for j:=1 to m do
                s:=s+a[ i,j ]*b[ j ];
        c[ i ]:=s;
        end;
writeln ('Получен вектор C ');
for i:=1 to n do
        write(c[ i ]:6);
End.
```

Умножение матрицы на матрицу

Для умножения матрицы A размером n*k на матрицу B размером k*m необходимо вычислить $c_{ij} = \sum_{l=1}^{k} a_{il} b_{lj}$, i=1,...,n; j=1,...,m.

```
Const
    n=3;
    m=4;
    k=5;
Var
i, j, s : integer;
a : array [1..n,1..k] of integer;
b : array [1..k,1..m] of integer;
c : array [1..n,1..m] of integer;
Begin
randomize;
writeln ('Сформирована матрица
    A');
for i:=1 to n do
begin
    for  j:=1 to k do
      begin
        a[ i,j ]:=random(101)-50;
            write (a[ i,j ]:6);
      end;
    writeln;
end;
writeln ('Сформирована матрица B');
for i:=1 to k do
begin
    for  j:=1 to m do
      begin
        b[ i,j ]:=random(351)-85;
```

Умножение матрицы на матрицу

```
      write (a[ i,j ]:6);
      end;
    writeln();
end;
for i:=1 to n do
    for j:=1 to m do
    begin
    s:=0;
    for l:=1 to k do
      s:=s+a[ i,l ]*b[ l,j ];
    c[ i,j ]:=s;
    end;
```

```
writeln ('Сформирована матрица C');
for i:=1 to n do
        begin
        for j:=1 to m do
                write (c[ i,j ]:6);
        writeln;
        end
End.
```

Удаление строки матрицы

Алгоритм удаления строки является сходным с алгоритмом удаление элементов одномерного массива, за тем исключением, что операция переноса элементов выполняется для каждого столбца при переборе строк. Рассмотрим программу удаления из матрицы A заданной с клавиатуры строки T.

```
Const
    n=10; m=5;
    Var i, j, t, n : integer;
a : array [1..n,1..m] of integer;
Begin
for i:=1 to n do
for  j:=1 to m do
  a[ i,j ]:=random(101)-50;

writeln ('Введите номер строки для
    удаления');
readln (t);
k:=n-1;
for i:=t to k do
      for j:=1 to m do
            a[ i,j ]=a[ i+1,j ];
writeln ('Получена матрица ');
for i:=1  to k do
      begin
      for j:=1 to m do
            write (a[ i,j ]);
      writeln;
end
End.
```

Многомерные массивы

Массивы могут быть более чем двумерными.

Пример:

```
...
a : array [1..5, 1..3, 1..16,  1..4 ] of real;
...
for i:=1 to 5 do
     for j:=1 to 3 do
       for k:=1 to 16 do
             for m:=1 to 4 do
                   a[ i,j,k,m ]:=random(101);
...
```