Урок 1

Массивы в Паскале. Одномерные массивы

Предположим, что программа работает с большим количеством однотипных данных. Скажем около ста разных целых чисел нужно обработать, выполнив над ними те или иные вычисления. Как вы себе представляете 100 переменных в программе? И для каждой переменной нужно написать одно и тоже выражение вычисления значения? Это очень неэффективно.

Есть более простое решение. Это использование такой структуры (типа) данных как **массив**. Массив представляет собой последовательность ячеек памяти, в которых хранятся однотипные данные. При этом существует всего одно имя переменной связанной с массивом, а обращение к конкретной ячейке происходит по ее индексу (номеру) в массиве.

Нужно четко понимать, что индекс ячейки массива не является ее содержимым. Содержимым являются хранимые в ячейках данные, а индексы только указывают на них. Действия в программе над массивом осуществляются путем использования имени переменной, связанной с областью данных, отведенной под массив.

Итак, *массив – это именованная группа однотипных данных, хранящихся в последовательных ячейках памяти*. Каждая ячейка содержит элемент массива. Элементы нумеруются по порядку, но необязательно начиная с единицы (хотя в языке программирования Pascal чаще всего именно с нее). Порядковый номер элемента массива называется **индексом** этого элемента.

Помним, все элементы определенного массива имеют один и тот же тип. У разных массивов типы данных могут различаться. Например, один массив может состоять из чисел типа integer, а другой – из чисел типа real.

Индексы элементов массива обычно целые числа, однако могут быть и символами, а также описываться другими порядковыми типами. Т.е. для индекса можно использовать тип, в котором определена дискретная последовательность значений, и все эти значения можно пересчитать по порядку. Индексировать можно как константами и переменными, так и выражениями, результат вычисления которых дает значение перечислимого типа.

Если индекс массива может приобретать все допустимые значения определенного перечислимого типа, то при описании массива возможно задание имени типа вместо границ изменения индекса. При этом границами индекса будут первое и последнее значения в описании типа индекса. Границы изменения индексов могут задаваться с помощью ранее объявленных констант. Рекомендуется предварительно объявлять тип массива в разделе описания типов.

Массив можно создать несколькими способами.

…

**const** n = 200;

**type**

months = (jan, feb, mar, apr, may, jun, jul, aug, sep, oct, nov, dec);

years = 1900..2100;

people = **array**[years] **of** **longint**;

**var**

growth: **array**[months] **of** **real**;

hum: people;

notes: **array**[1..n] **of** **string**;

Обращение к определенному элементу массива осуществляется путем указания имени переменной массива и в квадратных скобках индекса элемента.

Простой массив является одномерным. Он представляет собой линейную структуру.

**var** ch: **array** [1..11] **of** **char**;

h: **char**;

i: **integer**;

**begin**

**for** i := 1 **to** 11 **do** read (ch[i]);

**for** i := 1 **to** 11 **do** write (ch[i]:3);

readln

**end**.

В примере выделяется область памяти под массив из 11 символов. Их индексы от 1 до 11. В процессе выполнения программы пользователь вводит 11 любых символов (например, ‘q’, ’w’, ’e’, ’2’, ’t’, ’9’, ’u’, ’I’, ’I’, ’o’, ’p’), которые записываются в ячейки массива. Текущее значение переменной i в цикле for используется в качестве индекса массива. Второй цикл for отвечает за вывод элементов массива на экран.

***Пример:***программа поиска максимума из двух целых чисел.

[?](http://bigspawn.blogspot.com/)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | Var x,y,m,n: integer;  Function MaxNumber(a,b: integer): integer;    Var max: integer;  Begin    If a>b Then max := a    Else max := b;    MaxNumber := max;  End;  Begin    write('Введите x,y ');    readln(x,y);    m := MaxNumber(x,y);    n := MaxNumber(2,x+y);    writeln('m=',m,'n=',n);  End. |

Урок 2

# Массивы в Pascal

**Массив в программировании** является набором элементов одного типа (однотипных).  
  
Различают несколько видов массивов – *одномерный* (векторный) и *многомерный*.  
  
Элементы в массиве характеризуются своими именами и порядковыми номерами - индексами.  
  
Индекс – это порядковый номер элемента в массиве.  
  
В Паскале каждому элементу присваивается  один или несколько индексов, которые описывают положение элемента в массиве.

## Одномерный массив

Синтаксис массива в Pascal таков:

[?](http://bigspawn.blogspot.com/2012/01/massiv-v-pascal.html)

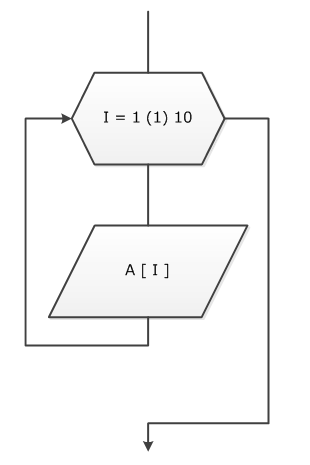
|  |  |
| --- | --- |
| 1 | Var a : array [1..10] Of integer ; |

Где :  
1 – нижний индекс  
10 – верхний индекс  
A – имя переменной массива  
[ ] – диапазон значений  
Integer – тип данных  
A[ i ] – обращение к элементу массива в Pascal  
  
Тип элементов массива может быть любым допустимым типом в Pascal, кроме файлов (даже массивом).  
  
Пример массива : A = {1,-5,230,55,-88,0,100}  
  
Когда описывается массив, его верхний индекс должен быть строго определён.  
  
При описании массива идёт распределение памяти, и компилятор должен знать какое количество памяти нужно выделить под описанный массив.  
  
Ограничения по количеству индексов в массиве Pascal не ограничивается. Однако сам массив не должен быть больше чем 65537 байт.  
  
Массив также можно объявлять в разделе описания типов:

[?](http://bigspawn.blogspot.com/2012/01/massiv-v-pascal.html)

|  |  |
| --- | --- |
| 1  2 | Type mass = array [1..100] Of real ;  Var a,b,c : mass ; |

Доступ к элементам массива осуществляется в цикле.  
  
Наиболее рациональным способом обработки элементов массива в Паскаль является оператор цикла с параметром.

[](http://4.bp.blogspot.com/-6QxkE3a2hNQ/TyJ3dpLZB9I/AAAAAAAAAPQ/KNT4IaJMClE/s1600/%D0%9C%D0%B0%D1%81%D1%81%D0%B8%D0%B2.png)

Как думаете почему? Да потому, что нам известно конечное число элементов в массиве.

### Алгоритмы заполнения массива в Pascal

1. Ввод элементов массива с помощью компьютера осуществляется с помощью такой конструкции :

[?](http://bigspawn.blogspot.com/2012/01/massiv-v-pascal.html)

|  |  |
| --- | --- |
| 1  2 | For i := 1 To 10 Do  read(A[i]); |

1. Задание массива случайным образом.  
     
   Массив можно задать случайным образом с помощью датчика случайных величин.  
     
   Для запуска датчика случайных величин в Паскаль нужно прописать специальную конструкцию - **Randomize**;  
     
   Новое значение генерируется с помощью функции Random(n) ,где n – целое число. В таком случае генерируется любое число с диапазоном от 0 до n.

[?](http://bigspawn.blogspot.com/2012/01/massiv-v-pascal.html)

|  |  |
| --- | --- |
| 1 | K := Random (100); |

Если функция Random используется без параметра, то она генерирует вещественное число (тип real) в диапазоне 0 < X < 1

[?](http://bigspawn.blogspot.com/2012/01/massiv-v-pascal.html)

|  |  |
| --- | --- |
| 1 | X := Random ; |

#### Заполнение массива случайным образом

Данная конструкция в Pascal реализует заполнение массива случайно.

[?](http://bigspawn.blogspot.com/2012/01/massiv-v-pascal.html)

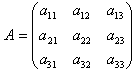
|  |  |
| --- | --- |
| 1  2  3  4  5  6 | Randomize ;  For i := 1 To 10 Do  Begin      A[i] := random\*100-70 ;      write(A[i]:6:2) ;  End ; |

Урок 3

### Двумерные массивы Паскаля – матрицы

Двумерный массив в Паскале трактуется как одномерный массив, тип элементов которого также является массивом (массив массивов). Положение элементов в двумерных массивах Паскаля описывается двумя индексами. Их можно представить в виде прямоугольной таблицы или матрицы.

Рассмотрим двумерный массив Паскаля размерностью 3\*3, то есть в ней будет три строки, а в каждой строке по три элемента:



Каждый элемент имеет свой номер, как у одномерных массивов, но сейчас номер уже состоит из двух чисел – номера строки, в которой находится элемент, и номера столбца. Таким образом, номер элемента определяется пересечением строки и столбца. Например, a 21 – это элемент, стоящий во второй строке и в первом столбце.

### Описание двумерного массива Паскаля.

Существует несколько способов объявления двумерного массива Паскаля.

Мы уже умеем описывать одномерные массивы, элементы которых могут иметь любой тип, а, следовательно, и сами элементы могут быть массивами. Рассмотрим следующее описание типов и переменных:

**Пример описания двумерного массива Паскаля**

**Type**  
Vector = **array** [1..5] **of** <тип\_элементов>;  
Matrix= **array** [1..10] **of** vector;  
**Var** m: matrix;

Мы объявили двумерный массив Паскаля m, состоящий из 10 строк, в каждой из которых 5 столбцов. При этом к каждой i -й строке можно обращаться m [ i ], а каждому j -му элементу внутри i -й строки – m [ i , j ].

Определение типов для двумерных массивов Паскаля можно задавать и в одной строке:

**Type**  
Matrix= **array** [1..5] **of** **array** [1..10] **of** < тип элементов >;  
или еще проще:  
**type**  
matrix = **array** [1..5, 1..10] **of** <тип элементов>;

Обращение к элементам двумерного массива имеет вид: M [ i , j ]. Это означает, что мы хотим получить элемент, расположенный в i -й строке и j -м столбце. Тут главное не перепутать строки со столбцами, а то мы можем снова получить обращение к несуществующему элементу. Например, обращение к элементу M [10, 5] имеет правильную форму записи, но может вызвать ошибку в работе программы.

### Основные действия с двумерными массивами Паскаля

Все, что было сказано об основных действиях с одномерными массивами, справедливо и для матриц. Единственное действие, которое можно осуществить над однотипными матрицами целиком – это присваивание. Т.е., если в программе у нас описаны две матрицы одного типа, например,

**type**  
matrix= **array** [1..5, 1..10] **of** integer;  
**var**  
   a , b : matrix ;

то в ходе выполнения программы можно присвоить матрице a значение матрицы b ( a := b ). **Все остальные действия выполняются поэлементно, при этом над элементами можно выполнять все допустимые операции, которые определены для типа данных элементов массива.** Это означает, что если массив состоит из целых чисел, то над его элементами можно выполнять операции, определенные для целых чисел, если же массив состоит из символов, то к ним применимы операции, определенные для работы с символами.

### Ввод двумерного массива Паскаля.

Для последовательного ввода элементов одномерного массива мы использовали цикл for, в котором изменяли значение индекса с 1-го до последнего. Но положение элемента в двумерном массиве Паскаля определяется двумя индексами: номером строки и номером столбца. Это значит, что нам нужно будет последовательно изменять номер строки с 1-й до последней и в каждой строке перебирать элементы столбцов с 1-го до последнего. Значит, нам потребуется два цикла for , причем один из них будет вложен в другой.

Рассмотрим пример ввода двумерного массива Паскаля с клавиатуры:

**Пример программы ввода двумерного массива Паскаля с клавиатуры**

**type**  
   matrix= **array** [1..5, 1..10] **of** integer;  
**var**  
   a, : matrix;  
   i, j: integer; { индексы массива }  
**begin**  
   **for** i :=1 **to** 5 **do** {цикл для перебора всех строк}  
      **for** j :=1 **to** 10 **do** {перебор всех элементов строки по столбцам}  
         readln ( a [ i , j ]); {ввод с клавиатуры элемента, стоящего в i -й строке и j -м столбце}

Двумерный массив Паскаля можно заполнить случайным образом, т.е. использовать функцию random (N), а также присвоить каждому элементу матрицы значение некоторого выражения. Способ заполнения двумерного массива Паскаля выбирается в зависимости от поставленной задачи, но в любом случае должен быть определен каждый элемент в каждой строке и каждом столбце.

### Вывод двумерного массива Паскаля на экран.

Вывод элементов двумерного массива Паскаля также осуществляется последовательно, необходимо напечатать элементы каждой строки и каждого столбца. При этом хотелось бы, чтобы элементы, стоящие в одной строке, печатались рядом, т.е. в строку, а элементы столбца располагались один под другим. Для этого необходимо выполнить следующую последовательность действий (рассмотрим фрагмент программы для массива, описанного в предыдущем примере):

**Пример программы вывода двумерного массива Паскаля**

**for** i :=1 **to** 5 **do** {цикл для перебора всех строк}  
**begin**  
   **for** j :=1 **to** 10 **do** {перебор всех элементов строки по столбцам}  
      **write** ( a [ i , j ]:4); {печать элементов, стоящих в i -й строке матрицы в одной экранной строке, при этом для вывода каждого элемента отводится 4 позиции}  
   writeln ; {прежде, чем сменить номер строки в матрице, нужно перевести курсор на начало новой экранной строки}  
**end** ;

**Замечание (это важно!):** очень часто в программах студентов встречается ошибка, когда ввод с клавиатуры или вывод на экран массива пытаются осуществить следующим образом: readln (a), writeln (a), где а – это переменная типа массив. При этом их удивляет сообщение компилятора, что переменную этого типа невозможно считать или напечатать. Может быть, вы поймете, почему этого сделать нельзя, если представите N кружек, стоящих в ряд, а у вас в руках, например, чайник с водой. Можете вы по команде «налей воду» наполнить сразу все кружки? Как бы вы ни старались, но в каждую кружку придется наливать отдельно. Заполнение и вывод на экран элементов массива также должно осуществляться последовательно и поэлементно, т.к. в памяти ЭВМ элементы массива располагаются в последовательных ячейках.

**Практическая работа № 22**

|  |
| --- |
| **Тема:** Составление программы для вычислений в одномерном массиве |
| **Цель**:   * владение умением понимать программы, написанные на выбранном для изучения универсальном алгоритмическом языке высокого уровня; знанием основных конструкций программирования; умением анализировать алгоритмы с использованием таблиц; |
| **Оборудование -**справочники с формулами |
| **Компьютерная программа** Среда программирования PascalABC , Текстовый редактор |

**Теория**

массив - это упорядоченный набор однотипных значений – компонент массива

Тип компонент определяет тип массива и называется базовым.

Базовый тип массива бывает:

* целочисленный;
* действительный
* символьный
* строковый
* логический

**Формат записи элемента массива:  
<имя\_массива*>[<индекс\_1>; <индекс\_2>;… <индекс\_n>]***

Количество индексов определяет **размерность** массива. тип индекса имеет простой порядковый тип

Массиву присваивается имя (А, Spisok, R,Temperatura) посредством которого можно ссылаться на него, как на единое целое, так и на любую из его компонент по индексу

Ввод и вывод в массиве

|  |  |
| --- | --- |
| **Ввести десять значений целочисленного массива:** | **Вывести 50 значений массива действительных чисел:** |
| 123.png | 213.png |
|  |  |
| Var A: array [1..25] of integer;  i: integer;  ………………………… for i:=1 to 10 do  begin  write(‘A[‘ , i ,’]=‘);  readln(a[ i ]);  end; | Var B: array [1..60] of real;  i: integer;  ………………………… for i:=1 to 50 do  writeln(‘B[‘ , i ,’]=‘, b[ i ]:5:3); |

**Постановка задачи**

Пусть заданы:

* массив, имеющий имя, количество элементов, тип
* характеристики массива, такие как сумма, произведение
* формула, содержащая характеристики массива

Необходимо вычислить значение заданной формулы

ВАРИАНТЫ

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № варианта | Имя массива | Количество элементов | Условия отбора | | | | Формула |
| S,k1-сумма и количество | P,k2- произведение и количество | Min элемент и его номер | Max,i\_max |
| 0 | B | m | однозначных |  |  | k |  |
| 14 | G | r |  | Заканчивающихся цифрой 7 | k |  | W=+mink |
| 2 | E | m | двузначные | Четные отрицательные |  |  |  |
| 3 | B | n | заканчиваются цифрой 4 | нечетные двузначные |  |  |  |
| 4 | Q | g | Четные двузначные | Заканчиваются цифрой 2 |  |  |  |
| 15 | D | m |  | отрицательных однозначных элементов массива |  | k |  |
| 5 | R | S1 | нечетные |  | w |  |  |
| 6 | Y | o | Кратны 10 | Принадл. отрзк [-50;400] |  |  |  |
| 7 | E | f |  | Кратны 13 | k |  |  |
| 8 | J | t | нечетных трехзначных | не кратных 3 |  |  |  |
| 9 | Y | m | Кратны 5 | Принадлежат отрезку [-20;20] |  |  |  |
| 10 | G | r |  | Принадлежат отрезку  [-1;0] |  | k |  |
| 11 | T | n | Принадлежат отрезку [-2.1;-2,2] | Принадлежат интервалу (0;1) |  |  |  |
| 12 | F | k | Кратны 3 |  |  | j |  |
| 13 | J | r | Заканчиваются нулем |  | k |  | W=+mink |
| 0 | B | m | однозначных |  |  | k |  |

**Ход работы**

1. Сделать контрольный тест
   * число элементов массива не менее 7;
   * тривиальные случаи, когда число элементов массива, удовлетворяющих определенным условиям, равно 0 или 1 не рассматривать.
2. Составить и отладить программу.
3. Запустить программу на исполнение
4. Сравнить результаты контрольного теста и программы (результаты программы должны быть приведены)
5. Сделать вывод.

Образец выполнения для варианта 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № варианта | Имя массива | Количество элементов | Условия отбора | | | | Формула |
| S,k1-сумма и количество | P,k2- произведение и количество | Min элемент и его номер | Max,i\_max |
| 0 | B | m | однозначных |  |  | k |  |

**Задание**

|  |  |
| --- | --- |
| Пусть задан массив: B[m] Вычислить | |
| Где S-Сумма однозначных элементов массива  K1-количество однозначных элементов | Max- максимальный элемент в массиве  k- номер максимального элемента |
|  | |



Контрольный тест и результат в стандартном случае

|  |  |
| --- | --- |
| S=18  K1=3  Max=31  K=6 |  |

Контрольный тест и результат в критическом случае

|  |  |
| --- | --- |
| S=9  K1=3  Max=61  K=3  Так как знаменатель обращается в нуль, появится текст:  Критический случай  Деление на 0!!! |  |

5. Вывод Экспериментальная проверка подтвердила правильность выбранного алгоритма и составленной программы

**Контрольные вопросы**

1. Задан массив:

*var C: array [1..10] of integer;*

Являются ли корректными имена и/или значения и/или индексы элементов массива :*C[0]=10; C[20]=20;…*

2. Задан массив:

*var R: array [1..34] of char;*

Являются ли корректными имена и/или значения и/или индексы элементов массива *:R[33]=‘x’; R[34]=‘y’;…*

3. Как изменится значение переменной S после выполнения фрагмента программы?

*s:=1; k:=0*

*for i:=1 to n do s:= s\*f[i] ;*

**Практическая работа № 23**

|  |
| --- |
| **Тема:** **Составление программы для вычислений в двумерном массиве** |
| **Цель**:   * владение компьютерными средствами представления и анализа данных; * владение умением понимать программы, написанные на выбранном для изучения универсальном алгоритмическом языке высокого уровня; знанием основных конструкций программирования; умением анализировать алгоритмы с использованием таблиц; * владение навыками алгоритмического мышления и понимание необходимости формального описания алгоритмов; * владение стандартными приемами написания на алгоритмическом языке программы для решения стандартной задачи с использованием основных конструкций программирования и отладки таких программ; использование готовых прикладных компьютерных программ по выбранной специализации. |
| **Оборудование -**справочники с формулами |
| **Компьютерная программа** Среда программирования PascalABC , Текстовый редактор , Электронная таблица |

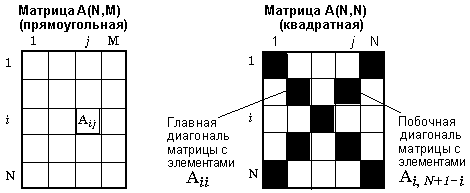
**Теория**

**Двумерный массив**

Каждый элемент двумерного массива характеризуется двумя индексами:

Превый индекс указывает на номер строки, а второй – номер столбца

Двумерные массивы удобней всего ассоциировать с таблицей или матрицей:



**Если количество строк и столбцов матрицы равны, то матрица называется квадратной.**

**В квадратной матрице существует понятие главной и побочной диагонали.**

**Таблица условий принадлежности элементам матрицы:**

|  |  |
| --- | --- |
|  |  |

**Ввод и вывод в двумерном массиве (матрице) осуществляется поэлементно с помощью вложенных циклов**

|  |  |
| --- | --- |
| **Формирование с помощью процедуры ввода** | **Вывод 1 способ** |
| **for i:=1 to n do**  **for j:=1 to m do**  **begin**  **write(‘[A’,i,j,’]=‘);**  **readln(A[i,j]);**  **end;** | **for i:=1 to n do**  **for j:=1 to m do**  **begin**  **write(A[i,j]:6:3);**  **writeln;** |

|  |  |
| --- | --- |
| **Формирование с помощью функции (например random)** | **Вывод 2 способ(с учетом принадлежности области ниже главной диагонали)** |
| **for i:=1 to n do**  **for j:=1 to m do**  **A[I,j]:=random(100);** | **for i:=1 to n do**  **for j:=1 to n do**  **begin**  **a[i,j]:=….;**  **if i>j then textcolor(11) else textcolor(4);**  **gotoXY(j\*7,i\*2);write(a[i,j]:9:3);**  **end;**  **writeln;** |

**Постановка задачи**

Пусть заданы:

* двумерный массив, имеющий имя, количество строк, столбцов, тип
* формула, формирующая элементы массива

Необходимо:

* вычислить и вывести элементы массива заданным способом
* вывести массив в виде матрицы заданным способом
* вычислить и вывести некоторые характеристики массива (сумма, произведение элементов, максимальный элемент, минимальный элемент), удовлетворяющие заданным условиям:

**ВАРИАНТЫ**

|  |  |  |
| --- | --- | --- |
| № варианта | Идентификатор массива | Задание |
|  | А(m,m) | * Сформировать матрицу из действительных чисел равных вывести матрицу таким образом, чтобы элементы на главной диагонали были окрашены в один цвет, а все остальные в другой цвет * найти минимальный элемент из расположенных над главной диагональю |
|  | C(m,m) | * Сформировать матрицу из целых чисел, равных вывести матрицу таким образом, чтобы элементы в четных строках были окрашены в один цвет, а в нечетных – в другой. * Вычислить и вывести количество кратных 7 элементов на главной и на побочной диагоналях |
|  | F(m,m) | * Сформировать матрицу из действительных чисел равных  вывести матрицу таким образом, чтобы элементы под и над главной диагональю были окрашены в один цвет, а все остальные – в другой * Найти количество больших 10 элементов во всей матрице |
|  | T(n,n) | * Сформировать матрицу из действительных чисел равных  вывести матрицу таким образом, чтобы элементы на побочной диагонали и на главной диагонали, были окрашены в один цвет –все остальные – в другой. * Найти максимальный элемент из расположенных на главной диагонали элементов |
|  | А(m,m) | * Сформировать матрицу из действительных чисел равных вывести матрицу таким образом, чтобы элементы на побочной диагонали были окрашены в один цвет, а все остальные в другой цвет * найти сумму элементов, расположенных выше побочной диагонали |
|  | C(m,m) | * Сформировать матрицу из целых чисел, равных целой части вывести матрицу таким образом, чтобы элементы в нечетных строках были окрашены в один цвет, а в нечетных – в другой. * найти максимальный элемент из расположенных на главной диагонали элементов. |
|  | F(m,m) | * Сформировать матрицу из действительных чисел равных  вывести матрицу таким образом, чтобы элементы под и над главной диагональю были окрашены в один цвет, а все ост- в другой * сумму отрицательных элементов, расположенных ниже побочной диагонали |
|  | T(n,n) | * Сформировать матрицу из действительных чисел равных   вывести матрицу таким образом, чтобы элементы на побочной диагонали и на главной диагонали, были окрашены в один цвет –все остальные – в другой. * Найти сумму элементов принадлежащих отрезку [2;5] и расположенных ниже главной диагонали |
|  | А(m,m) | * Сформировать матрицу из действительных чисел равных вывести матрицу таким образом, чтобы элементы под побочной и под главной диагональю были окрашены в один цвет, а все остальные в другой цвет * найти минимальный элемент из расположенных под главной диагональю |
|  | C(m,m) | * Сформировать матрицу из действительных чисел, равных  вывести матрицу таким образом, чтобы элементы в нечетных строках были окрашены в один цвет, а в нечетных – в другой. * Вычислить и вывести количество кратных 11 элементов |
|  | F(m,m) | * Сформировать матрицу из действительных чисел равных  вывести матрицу таким образом, чтобы элементы под и над главной диагональю были окрашены в один цвет, а все остальные – в другой * Найти количество меньших 11 элементов в первых трех строках матрицы |
|  | T(n,n) | * Сформировать матрицу из действительных чисел равных  вывести матрицу таким образом, чтобы элементы на побочной диагонали и на главной диагонали, были окрашены в один цвет –все остальные – в другой. * Найти максимальный элемент из расположенных на главной диагонали элементов |
|  | А(m,m) | * Сформировать матрицу из действительных чисел равных вывести матрицу таким образом, чтобы элементы над главной и над побочной диагоналях были окрашены в один цвет, а все остальные в другой цвет * найти сумму элементов, расположенных выше побочной диагонали |
|  | C(m,m) | * Сформировать матрицу из целых чисел, равных дробной части вывести матрицу таким образом, чтобы элементы в четных строках были окрашены в один цвет, а в нечетных – в другой. * найти минимальный элемент из расположенных под главной и над побочной диагоналями элементов. |
|  | F(m,m) | * Сформировать матрицу из действительных чисел равных  вывести матрицу таким образом, чтобы элементы на главной и побочной диагонали были окрашены в один цвет, а все ост- в другой * сумму неположительных элементов, расположенных ниже побочной диагонали |

**Ход работы:**

1. Сделать контрольный тест

* + Размерность массива не менее 5;
  + тривиальные случаи, когда число элементов матрицы, удовлетворяющих определенным условиям, равно 0 или 1 не рассматривать.

2. Составить и отладить программу.

3. Запустить программу на исполнение

4. Сравнить результаты контрольного теста и программы (результаты программы должны быть приведены)

5. Сделать вывод.

**Контрольные вопросы**

1. Записать условие ссотоящие в том, что на побочной диагонали только четные числа

2. Записать условие. Состоящее в том, что в четных строках матрицы только числа кратные 5

3. Как записать условие вычисления максимального значения под главной и побочной диагоналями?

**Образец выполнения задания для варианта 0:**

**Задание**:

1. Сформировать матрицу из действительных чисел равных вывести матрицу таким образом, чтобы элементы над главной диагональю были окрашены в один цвет, а все ост- в другой  
2. Найти:

1. количество положительных элементов во всей матрице
2. максимальный элемент из расположенных ниже главной диагонали
3. сумму элементов, расположенных выше главной диагонали и превышающих максимальный элемент из предыдущего задания

1. Контрольный тест (в Электронной таблице):

|  |  |
| --- | --- |
|  | Формулу распространить вниз и вправо |
| 4. Результат исполнения программы | |
|  | |

Резу

2. Блок- схема:

|  |  |
| --- | --- |
| Формирование и вывод массива, вычисление количества положительных элементов | Поиск максимального элемента из расположенных ниже главной диагонали |
|  |  |
| Вычисление суммы элементов, расположенных выше главной диагонали и превышающих максимальный элемент из предыдущего задания |
|  |

3. Программа



5. Вывод

Результаты программы и контрольного теста совпали. Это является свидетельством того, что выбранный алгоритм можно использовать для любой размерности двумерного массива

**Практическая работа № 24**

|  |
| --- |
| **Тема:** **Составление программ с использованием символьных и строковых процедур и функций** |
| **Цель**:   * владение компьютерными средствами представления и анализа данных; * владение умением понимать программы, написанные на выбранном для изучения универсальном алгоритмическом языке высокого уровня; знанием основных конструкций программирования; умением анализировать алгоритмы с использованием таблиц; * владение навыками алгоритмического мышления и понимание необходимости формального описания алгоритмов; * владение стандартными приемами написания на алгоритмическом языке программы для решения стандартной задачи с использованием основных конструкций программирования и отладки таких программ; использование готовых прикладных компьютерных программ по выбранной специализации. |
| **Оборудование -**справочники с формулами |
| **Компьютерная программа** Среда программирования PascalABC , Текстовый редактор |

**Теория**

Строка (string) -Упорядоченная последовательность символов

Каждый символ строки занимает 1 байт памяти (в кодах ASCII). Длина строки - количество символов в строке лежит в диапазоне [0;255]

Различают:

|  |  |
| --- | --- |
| Строковые константы | Строковые переменные |
| **последовательность символов, заключенная в апострофы**  Например:   * ‘строковая константа’ * ‘delta’ * ‘Вася’ * ‘2367’ * ‘.:(%?б>’ * ‘’ - пустая строка (имеет нулевую длину) | **Описываются в разделе описания переменных**   * **Var** <имя\_переменной>:string[<максимальная\_длина\_строки>]; * **Var name:string[20];** * **Var stroka:string;**   -в этом случае имеется в виду, что строка может иметь любую максимальную длину (но не больше 255) |

Состав строки

* **Строка состоит из символов**

Простые Типы string и   
 char совместимы.   
**Cтроки(string) и символы(char) могут употребляться в одних и тех же выражениях**

* Элементы строки -символы (тип-char) идентифицируются с именем строки индексом например:
  + Name[11]
  + Z[i]
  + Slovo[k+1]
* **Выражение name[1] возвращает первый символ строки, а выражение name[0] возвращает длину строки, но не число, а символ, например:**
  + **Если Name:=‘ВАСЯ’  
     то Name[1]=‘В’, name[0]=‘4’**

Внутреннее представление строки можно пояснить рисунком:

0

1

2

3

…

i

i+1

…

n-1

n

Chr(i)

Занятая часть строки

Свободная часть строки

Наряду с символами, собственно представляющими содержание строки, в ней хранится показатель ее длины в нулевом байте, поэтому реальная длина строки на 1 байт больше декларируемой.

Нулевой сегмент, доступен для изменения с помощью присваивания **z[0]:=chr(65)**, при этом целое число типа byte[0-255] преобразуется к типу char. Это необходимо, так как каждый элемент строки, в том числе и показатель длины в нулевом сегменте, должен быть представлен как символ.

**Обработка Строк:**

Функции :

* **Chr(X:Byte): Char** Возвращает символ с заданным порядковым номером X.
* **Сoncat(S1**[**, S2, ..., SN**]**): String** Выполняет сцепку (конкатенацию) последовательности строк.
* **Copy(S: String; N,M: Integer): String** Возвращает подстроку из строки S, начиная с позиции N и длиной M символов.
* **Length(S: String): Byte** Возвращает количество символов в строке S.
* **Ord(X: Char): LongInt** Возвращает порядковый номер символа X в таблице кодов символов.

**Pos(SubS, S: String): Byte** Возвращает номер позиции, начиная с которой в строке S располагается подстрока SubS (если значение функции равно нулю, то S не содержит SubS).

Процедуры:

|  |  |
| --- | --- |
| **Delete(Var S:String; N,M:Integer)** | **Удаляет M символов из строки S, начиная с позиции N.** |
| **Insert(SubS: String; Var S:String; N:Integer)** | Вставляет подстроку SubS в строку S, начиная с позиции N. |
| **Str(X:Integer; Var S:String)** | Возвращает представление числа X в его символьной форме S. |
| **Val(S:String; Var X, Code: Integer)** | Возвращает представление символов строки S в ее числовой форме X. Параметр Code содержит признак ошибки преобразования (если Code = 0, ошибки нет). |

Условимся считать, что:

•         Слова – группы символов, разделенные пробелами;

•         Предложение –это группа символов, содержащая слова, пробелы

•         Условимся, что предложение:

–       не может начинаться с пробела (иначе это строка, из которой мы пробел удалим);

–       завершается одним из знаков препинания  
 (. ! ?) или возвратом каретки (#13)?

**Постановка задачи:**

1. Составить программу предусматривающую:
2. Ввод данных
3. Обработка данных
4. Все данные выводить с именами
5. Оформить отчет в следующей последовательности:
6. Задание
7. Текст программы
8. Образ экрана в стандартном и критическом случае
9. Анализ результатов выполнения программы

**Варианты**

1. Дана строка. Вывести в ней те слова, которые начинаются и заканчиваются одной буквой. Предусмотреть критический случай
2. Дана строка, подсчитать в ней количество вхождений букв r,k,tпредусмотреть критический случай
3. Дана строка, заканчивающаяся точкой. Вывести на монитор пятибуквенные слова. предусмотреть критический случай
4. Дана строка, определить и вывести длину самого длинного и длину самого короткого слова, если таких слов несколько, вывести все из них
5. Дана строка, содержащая английский текст  
   Найти и вывести количество слов, содержащих слог go  
   предусмотреть критический случай
6. В строке имеется одна точка с запятой. Подсчитать количество символов до точки с запятой и после нее. Предусмотреть критический случай
7. В записке слова зашифрованы: каждое записано наоборот. Расшифровать сообщение
8. Упорядочить массив заданных английских слов по алфавиту
9. Если длина введенной строки четная, удалить 2 первых и 2 последних символа.
10. Если длина введенной строки нечетная, удалить символ, стоящий посередине строки.
11. В строке заменить все двоеточия (:), точкой с запятой (;). Подсчитать количество замен.
12. Если длина введенной строки кратна 5, первую часть строки поменять местами со второй.
13. Даны три строки: S1, S2, S3. Заменить в строке S1 первое вхождения строки S2 на S3.
14. Заменить в заданном тексте буквосочетание "*min*" на "*max*".
15. Дана строка. Определить, сколько раз в нее входит группа букв abc

**Образец выполнения задания**

В заданном предложении на английском языке вывести слово, в котором доля гласных (*A, E, I, O, U -*строчных или прописных) максимальна.

 program predlogenie;



|  |
| --- |
| Результат программы в стандартном случае |
| Результат программы в критическом случае |

**Контрольные вопросы**

1. Что такое **Сoncat**?
2. Что такое **Length(s)**?
3. После выполнения процедуры Val(S:String; Var X, Code: Integer) Какой результат примет Х, при S=’100’
4. После выполнения процедуры Val(S:String; Var X, Code: Integer) Какой результат примет Code, при S=’Code’?

**Практическая работа № 25**

|  |
| --- |
| **Тема:** **Составление программ, использующих процедуры ввода- вывода и обработки массивов.** |
| **Цель**:   * владение компьютерными средствами представления и анализа данных; * владение умением понимать программы, написанные на выбранном для изучения универсальном алгоритмическом языке высокого уровня; знанием основных конструкций программирования; умением анализировать алгоритмы с использованием таблиц; * владение навыками алгоритмического мышления и понимание необходимости формального описания алгоритмов; * владение стандартными приемами написания на алгоритмическом языке программы для решения стандартной задачи с использованием основных конструкций программирования и отладки таких программ; использование готовых прикладных компьютерных программ по выбранной специализации. |
| **Оборудование -**справочники с формулами |
| **Компьютерная программа** Среда программирования PascalABC , Текстовый редактор |

**Теория**

**Подпрограмма –**

**это именованная логически законченная группа операторов языка, которую можно вызвать для выполнения по имени любое количество раз из различных мест программы.**

**В языке программирования Pascal существует два вида подпрограмм:**

|  |  |
| --- | --- |
| **Процедуры - Procedure** | **Функции Function** |
| -это подпрограмма, являющаяся независимой и поименованной частью программы, предназначенной для **выполнения определенных действий.**  Процедуры бывают:   * встроенные (стандартные) * пользовательские | это подпрограмма для определения функциональной зависимости для **возвращения результата**  Функции бывают :   * стандартные (встроенные) такие как sin, mod, pi * пользовательские |
| Пользовательские процедуры состоят из:  Заголовка процедуры  **PROCEDURE** N[(P1:TYPE1; P2:TYPE2; VAR P3:TYPE3 ...)];   * N - имя процедуры * Pi - формальные параметры * TYPE i – типы формальных параметров   Процедура помещается в главной программе после раздела **var** и перед кодовым сегментом программы  Тела процедуры  Тело процедуры имеет ту же структуру, что и главная программа:  Разделы:   * **label** * **const** * **type** * **var** * **кодовый сегмент (от begin** до **end**). | Функция обладает всеми свойствами и особенностями процедур, но при этом существует два отличия:  В заголовке функции, начинающегося служебным словом Function, обязательно указывается имя типа возвращаемого значения  function exist(x,y,z:real):boolean;  function Pifagor(a,b:real):real;  Begin  Pifagor:=sqrt(sqr(a)+sqr(b));  End; |
| **Отличие процедуры от функции**   * Функция в отличие от процедуры возвращает единственное скалярное значение * Результат вычисления функции присваивается имени, а в процедуре -входит в список параметров. * Имя функции имеет тип. * Обращение к функции представляет операнд, а процедуры- оператор. | |

**Постановка задачи:**

Пусть имеется два массива. Первый массив формируется автоматически, Второй- вручную путем ввода значений с клавиатуры

**ВАРИАНТЫ**

1. Написать программу для нахождения величин:   
   P1, k1 – произведение и количество четных отрицательных элементов массива С(n)  
   S2,k2 –– сумма квадратов и количество двузначных чисел целочисленного массива Е(m)
2. Имеются два массива: массив абсцисс X(n) и массив ординат Y(n) Написать программу для нахождения величины

Px, – произведение абсцисс точек, принадлежащих внутренности единичного круга с центром в точке (2,-3)   
Sy –– cумма ординат точек, принадлежащих внутренности единичного круга с центром в точке (2,-3) k- количество таких точек

1. Написать программу для нахождения величин:  
   P1, k1 – произведение и количество нечетных неоднозначных элементов массива A(n)  
   S2,k2 ––сумма и количество четырехзначных элементов массива В(m)
2. Вычислить сумму SUM и количество t элементов массива D(m), меньших среднего геометрического Ge и k – количество элементов, превышающих среднее арифметическое *Ar* элементов массива S(z)
3. Написать программу для нахождения величин   
   P1, k1 – произведение и количество кратных 7 положительных элементов массива С(n)  
   S2,k2 –– сумма и количество отрицательных трехзначных чисел целочисленного массива Е(m)
4. Вычислить среднее геометрическое кратных 3 элементов массива F(k) – G= и P – произведение отрицательных двузначных элементов массива J(r)
5. Написать программу для нахождения величин   
   S1, k1 – сумма и количество неположительных элементов массива A(n)  
   S2,k2 –– сумма и количество принадлежащих отрезку [-5;25] элементов массива В(m)
6. Написать программу для нахождения величин

P1, k1 – произведение и количество четных двузначных, не кратных 3 элементов массива A(n)  
P2,k2 –– произведение и количество нечетных трехзначных элементов массива В(m)

1. Написать программу для нахождения величин   
    S1, k1 – сумма и количество неотрицательных не кратных 13 элементов массива A(n)  
   S2,k2 –– сумма и количество неположительных кратных 11 элементов массива В(m)
2. Написать программу для нахождения величин:   
   S1, k1 – сумма квадратов и количество однозначных элементов массива С(n)  
   P2,k2 –– произведение и количество четных трехзначных чисел целочисленного массива Е(m)
3. Написать программу для нахождения величин:   
   S1, k1 – сумма и количество элементов массива A(n), принадлежащих отрезку [a,b]  
   S2,k2 –– сумма и количество элементов массива В(m), не принадлежащих отрезку [c,d]  
   a,b,c,d-произвольные числа
4. Определить минимальный элемент массива действительных чисел G(r) и найти его номер k. Вычислить произведение Р и количество s элементов массива T(m), кратных k из первого массива
5. Определить максимальный элемент действительного массива G(r) и найти его номер k. Вычислить произведение Р не кратных 5 элементов второго массива D(t)
6. Вычислить среднее арифметическое нечетных элементов целочисленного массива R(s) ( и номера v w наибольшего и наименьшего элементов второго массива F(l) соответственно

**Ход работы:**

1. Нарисовать блок-схему
2. Составить и отладить программу с использованием процедур.
3. Запустить программу на исполнение
4. Сделать контрольный расчет
   1. число элементов массива не менее 7;
   2. тривиальные случаи, когда число элементов массива, удовлетворяющих определенным условиям, равно 0 или 1 не рассматривать.
5. Сделать контрольный тест с помощью электронных таблиц или калькулятора
6. Сравнить результаты контрольного теста и программы (результаты программы должны быть приведены)
7. Сделать вывод.

Образец выполнения для варианта 0

**Задание:**

Пусть заданы два массива: A[n], B[m]

1 массив-случайный

2 массив – ручного ввода

Вычислить 

Где S1-Сумма положительных элементов первого массива

P2-Произведение нечетных элементов второго массива

K1-количество положительных элементов в 1 массиве

K2-количество нечетных элементов во втором массиве

Предусмотреть критический случай

|  |  |
| --- | --- |
| 1. Контрольный тест | 2. Блок-схема |
| Ввиду того, что в программе предусмотрен датчик случайных чисел, невозможно предугадать контрольный тест  Программу следует проверить после запуска |  |

3. Программа:

Program dva\_massiva\_proced;

uses crt;

var Z :real;

A:array [1..50] of integer;

B:array [1..50] of integer;

i,k1,k2,m,n ,s1,P2 :integer;

Procedure Input\_rnd\_mas(const s:char; var klv:integer; var msv:array [1..50] of integer);

Begin

write ('vvedite kolichestvo elementov sluchainogo massiva :'); ReadLn(klv);

Writeln('Massiv', s:3);

randomize;

For i := 1 to klv do msv[i]:=round(-30+60\*random);

end;

Procedure Input\_mas(const s:char; var klv:integer; Var msv:array[1..50] of integer);

begin

write ('vvedite kolichestvo elementov nesluchainogo massiva :'); ReadLn(klv);

Writeln('Massiv', s:3);

For i:=1 to klv do

Begin

Write(s,'[',i,']=');readln (msv[i]);

End;

end;

Procedure Output(const s:char;klv:integer;msv:array [1..50] of integer);

begin

WriteLn('Vivod Massiva',s:3);

For i := 1 to klv do

WriteLn(s,'[' , i , ']= ',msv[i]:5);

end;

Procedure sum\_pologit(klv:integer;msv:array [1..50] of integer;var sum:integer; var klv1:integer);

begin

Sum:=0;Klv1:=0;

for i:=1 to klv do

if msv[i]>0 then

begin

Sum:=Sum+msv[i] ;

Klv1:=Klv1+1;

end;

writeln('sum=',sum:5);

writeln('kolichestvo=',klv1:3);

end;

Procedure proizv\_nechet(klv:integer;msv:array [1..50] of integer;var proizv:integer; var klv1:integer);

begin

proizv:=1;Klv1:=0;

for i:=1 to klv do

if (msv[i]mod 2 = 1) then

begin

proizv:=proizv\*msv[i] ;

Klv1:=Klv1+1;

end;

writeln('proizv=',proizv:5);

writeln('kolichestvo=',klv1:3);

end;

**begin**

**Input\_rnd\_mas('a',n,a);**

**Output ('a',n,a);**

**sum\_pologit(n,a,s1,k1);**

**Input\_mas('b',m,b);**

**{Output ('b',m,b); }**

**proizv\_nechet(m,b,P2,k2)**

**end.**

|  |
| --- |
| Результат выполнения программы: |
|  |

**Контрольные вопросы**

1. Отличие процедуры и функции
2. Привести примеры стандартных функций
3. Привести примеры стандартных процедур

# Массивы. Тексты программ

### Ввод и вывод

**const** Sz = 100; *// Размер массива*

**var**

a: **array** [1..Sz] **of** **integer**;

N: **integer**; *// Количество элементов в массиве*

i: **integer**;

**begin**

write('Введите количество элементов в массиве: ');

readln(N);

write('Введите элементы массива: ');

**for** i:=1 **to** N **do**

read(a[i]);

write('Вывод элементов массива: ');

**for** i:=1 **to** N **do**

write(a[i],' ');

**end**.

### Заполнение случайными числами

**const** Sz = 100; *// Размер массива*

**var**

a: **array** [1..Sz] **of** **integer**;

N: **integer**; *// Количество элементов в массиве*

i: **integer**;

**begin**

N := 20;

**for** i:=1 **to** N **do**

a[i] := Random(100);

writeln('Элементы массива: ');

**for** i:=1 **to** N **do**

write(a[i],' ');

**end**.

### Заполнение арифметической прогрессией

**const**

Sz = 100;

a0 = 5; *// Первый элемент арифметической прогрессии*

d = 3; *// Разность арифметической прогрессии*

**var**

a: **array** [1..Sz] **of** **integer**;

N: **integer**; *// Количество элементов в массиве*

**begin**

N := 20;

a[1] := a0;

**for** **var** i:=2 **to** N **do**

a[i] := a[i-1] + d;

writeln('Арифметическая прогрессия: ');

**for** **var** i:=1 **to** N **do**

write(a[i],' ');

**end**.

### Заполнение степенями двойки

**const**

Sz = 100;

**var**

a: **array** [1..Sz] **of** **integer**;

N: **integer**;

**begin**

N := 20;

a[1] := 2;

**for** **var** i:=2 **to** N **do**

a[i] := a[i-1] \* 2;

writeln('Степени двойки: ');

**for** **var** i:=1 **to** N **do**

writeln(i:3,a[i]:9);

**end**.

### Заполнение числами Фибоначчи

**const**

Sz = 100;

**var**

a: **array** [1..Sz] **of** **integer**;

N: **integer**;

**begin**

N := 20;

a[1] := 1;

a[2] := 1;

**for** **var** i:=3 **to** N **do**

a[i] := a[i-2] + a[i-1];

writeln('Числа Фибоначчи: ');

**for** **var** i:=1 **to** N **do**

write(a[i],' ');

**end**.

### Инвертирование массива

**const**

Sz = 100;

**var**

a: **array** [1..Sz] **of** **integer**;

N: **integer**;

**begin**

N := 20;

**for** **var** i:=1 **to** N **do**

a[i] := Random(100);

writeln('Элементы массива: ');

**for** **var** i:=1 **to** N **do**

write(a[i],' ');

writeln;

**for** **var** i:=1 **to** N **div** 2 **do**

Swap(a[i],a[N-i+1]);

writeln('После инвертирования: ');

**for** **var** i:=1 **to** N **do**

write(a[i],' ');

**end**.

### Минимальный элемент в массиве и его индекс

**const**

Sz = 100;

**var**

a: **array** [1..Sz] **of** **real**;

N: **integer**;

min: **real**;

minind: **integer**;

**begin**

N := 20;

**for** **var** i:=1 **to** N **do**

a[i] := Random(100);

writeln('Элементы массива: ');

**for** **var** i:=1 **to** N **do**

write(a[i],' ');

writeln;

min := a[1];

minind := 1;

**for** **var** i:=2 **to** N **do**

**if** a[i]<min **then**

**begin**

min := a[i];

minind := i;

**end**;

writeln('Минимальный элемент: ',min);

writeln('Индекс минимального элемента: ',minind);

**end**.

### Минимальный четный элемент и его индекс

**const**

Sz = 100;

**var**

a: **array** [1..Sz] **of** **integer**;

N: **integer**;

min: **integer**;

minind: **integer**;

**begin**

N := 20;

**for** **var** i:=1 **to** N **do**

a[i] := Random(100);

writeln('Элементы массива: ');

**for** **var** i:=1 **to** N **do**

write(a[i],' ');

writeln;

min := **integer**.MaxValue;

**for** **var** i:=1 **to** N **do**

**if** (a[i]<min) **and** (a[i] **mod** 2 = 0) **then**

**begin**

min := a[i];

minind := i;

**end**;

**if** min = **integer**.MaxValue **then**

writeln('Четных элементов нет')

**else**

**begin**

writeln('Минимальный четный элемент: ',min);

writeln('Индекс минимального четного элемента: ',minind);

**end**;

**end**.

### Запись четных элементов массива в новый массив

**const**

Sz = 100;

**var**

a,b: **array** [1..Sz] **of** **integer**;

aN: **integer**; *// Количество элементов в массиве a*

bN: **integer**; *// Количество элементов в массиве b*

**begin**

aN := 20;

**for** **var** i:=1 **to** aN **do**

a[i] := Random(100);

writeln('Элементы массива: ');

**for** **var** i:=1 **to** aN **do**

write(a[i],' ');

writeln;

bN := 0;

**for** **var** i:=1 **to** aN **do**

**if** a[i] **mod** 2 = 0 **then**

**begin**

bN += 1;

b[bN] := a[i];

**end**;

writeln('Четные элементы массива: ');

**for** **var** i:=1 **to** bN **do**

write(b[i],' ');

**end**.

### Слияние отсортированных массивов в отсортированный

#### Способ 1.

**const**

aN = 10; *// Количество элементов в массиве a*

bN = 6; *// Количество элементов в массиве b*

cN = aN + bN; *// Количество элементов в массиве c*

**var**

a: **array** [1..aN] **of** **integer** := (1,5,12,15,47,89,98,112,171,180);

b: **array** [1..bN] **of** **integer** := (13,44,58,71,73,111);

c: **array** [1..сN] **of** **integer**;

ai,bi,ci: **integer**;

**begin**

writeln('Элементы массива a: ');

**for** **var** i:=1 **to** aN **do**

write(a[i],' ');

writeln;

writeln('Элементы массива b: ');

**for** **var** i:=1 **to** bN **do**

write(b[i],' ');

writeln;

ci := 1;

ai := 1;

bi := 1;

**while** (ai<=aN) **and** (bi<=bN) **do**

**begin**

**if** a[ai]<b[bi] **then**

**begin**

c[ci] := a[ai];

ai += 1;

**end**

**else**

**begin**

c[ci] := b[bi];

bi += 1;

**end**;

ci += 1;

**end**;

**while** ai<=aN **do**

**begin**

c[ci] := a[ai];

ai += 1;

ci += 1;

**end**;

**while** bi<=bN **do**

**begin**

c[ci] := b[bi];

bi += 1;

ci += 1;

**end**;

writeln('Результат слияния: ');

**for** **var** i:=1 **to** cN **do**

write(c[i],' ');

**end**.

#### Способ 2. С барьерным элементом

**const**

aN = 10; *// Количество элементов в массиве a*

bN = 6; *// Количество элементов в массиве b*

cN = aN + bN; *// Количество элементов в массиве c*

**var**

a: **array** [1..aN+1] **of** **integer** := (1,5,12,15,47,89,98,112,171,180,0); *// последний элемент - барьерный*

b: **array** [1..bN+1] **of** **integer** := (13,44,58,71,73,111,0);

c: **array** [1..cN] **of** **integer**;

ai,bi,ci: **integer**;

**begin**

writeln('Элементы массива a: ');

**for** **var** i:=1 **to** aN **do**

write(a[i],' ');

writeln;

writeln('Элементы массива b: ');

**for** **var** i:=1 **to** bN **do**

write(b[i],' ');

writeln;

a[aN+1] := **integer**.MaxValue; *// барьерный элемент - самый большой*

b[bN+1] := **integer**.MaxValue;

ci := 1;

ai := 1;

bi := 1;

**for** ci:=1 **to** cN **do**

**if** a[ai]<b[bi] **then**

**begin**

c[ci] := a[ai];

ai += 1;

**end**

**else**

**begin**

c[ci] := b[bi];

bi += 1;

**end**;

writeln('Результат слияния: ');

**for** **var** i:=1 **to** cN **do**

write(c[i],' ');

**end**.

### Сдвиг элементов влево

**const** N = 10;

**var** a: **array** [1..N] **of** **integer**;

**begin**

**for** **var** i:=1 **to** N **do**

a[i] := Random(100);

writeln('Элементы массива a: ');

**for** **var** i:=1 **to** N **do**

write(a[i],' ');

writeln;

**for** **var** i:=1 **to** N-1 **do**

a[i] := a[i+1];

a[N] := 0;

writeln('После сдвига влево: ');

**for** **var** i:=1 **to** N **do**

write(a[i],' ');

writeln;

**end**.

### Сдвиг элементов вправо

**const** N = 10;

**var** a: **array** [1..N] **of** **integer**;

**begin**

**for** **var** i:=1 **to** N **do**

a[i] := Random(100);

writeln('Элементы массива a: ');

**for** **var** i:=1 **to** N **do**

write(a[i],' ');

writeln;

**for** **var** i:=N **downto** 2 **do**

a[i] := a[i-1];

a[1] := 0;

writeln('После сдвига влево: ');

**for** **var** i:=1 **to** N **do**

write(a[i],' ');

writeln;

**end**.

### Удаление элемента

**const** N = 10;

**var**

a: **array** [1..N] **of** **integer**;

K: **integer**;

**begin**

**for** **var** i:=1 **to** N **do**

a[i] := Random(100);

writeln('Элементы массива a: ');

**for** **var** i:=1 **to** N **do**

write(a[i],' ');

writeln;

K := Random(1,N);

**for** **var** i:=K **to** N-1 **do**

a[i] := a[i+1];

writeln('После удаления элемента с индексом ',K,':');

**for** **var** i:=1 **to** N-1 **do**

write(a[i],' ');

writeln;

**end**.

### Вставка элемента

**const**

N = 10;

Elem = 666;

**var**

a: **array** [1..N+1] **of** **integer**;

K: **integer**;

**begin**

**for** **var** i:=1 **to** N **do**

a[i] := Random(100);

writeln('Элементы массива a: ');

**for** **var** i:=1 **to** N **do**

write(a[i],' ');

writeln;

K := Random(1,N);

**for** **var** i:=N **downto** K **do**

a[i+1] := a[i];

a[K] := 666;

writeln('После вставки элемента ',Elem,' в позицию ',K,':');

**for** **var** i:=1 **to** N+1 **do**

write(a[i],' ');

writeln;

**end**.

### Подсчет количества элементов, удовлетворяющих условию

**const** N = 20;

**var**

a: **array** [1..N] **of** **integer**;

K,Count: **integer**;

**begin**

**for** **var** i:=1 **to** N **do**

a[i] := Random(10);

writeln('Элементы массива a: ');

**for** **var** i:=1 **to** N **do**

write(a[i],' ');

writeln;

K := Random(10);

Count := 0;

**for** **var** i:=1 **to** N **do**

**if** a[i] = K **then**

Count += 1;

writeln('Количество элементов, равных ',K,': ',Count);

**end**.

### Есть ли элемент, удовлетворяющий условию

**const** N = 10;

**var**

a: **array** [1..N] **of** **integer**;

K: **integer**;

IsFound: **boolean**;

**begin**

**for** **var** i:=1 **to** N **do**

a[i] := Random(15);

writeln('Элементы массива a: ');

**for** **var** i:=1 **to** N **do**

write(a[i],' ');

writeln;

K := Random(15);

IsFound := **False**;

**for** **var** i:=1 **to** N **do**

**if** a[i] = K **then**

**begin**

IsFound := **True**;

break

**end**;

**if** IsFound **then**

writeln('Элемент ',K,' найден')

**else** writeln('Элемент ',K,' не найден')

**end**.

### Сортировка пузырьком

**const** N = 10;

**var**

a: **array** [1..N] **of** **integer**;

**begin**

**for** **var** i:=1 **to** N **do**

a[i] := Random(100);

writeln('Элементы массива a: ');

**for** **var** i:=1 **to** N **do**

write(a[i],' ');

writeln;

**for** **var** i:=n **downto** 2 **do**

**for** **var** j:=1 **to** i-1 **do**

**if** a[j+1]<a[j] **then**

Swap(a[j+1],a[j]);

writeln('После сортировки пузырьком: ');

**for** **var** i:=1 **to** N **do**

write(a[i],' ');

writeln;

**end**.

### Сортировка выбором

**const** N = 10;

**var**

a: **array** [1..N] **of** **integer**;

K: **integer**;

IsFound: **boolean**;

**begin**

**for** **var** i:=1 **to** N **do**

a[i] := Random(100);

writeln('Элементы массива a: ');

**for** **var** i:=1 **to** N **do**

write(a[i],' ');

writeln;

**for** **var** i:=1 **to** N-1 **do**

**begin**

**var** min := a[i];

**var** ind := i;

**for** **var** j:=i+1 **to** N **do**

**if** a[j]<min **then**

**begin**

min := a[j];

ind := j;

**end**;

a[ind] := a[i];

a[i] := min;

**end**;

writeln('После сортировки выбором: ');

**for** **var** i:=1 **to** N **do**

write(a[i],' ');

writeln;

**end**.