

Практическая работа № 2

Объявление массивов

Тема: Тема 2.11. Языки программирования. Особенности написания программ на процедурно - ориентированных языках.

Цели занятия: Изучить принципы работы с одномерными массивами на языке программирования Pascal. Получение навыков применения основных алгоритмов для решения задач с использованием массивов.

Ход практического занятия:

1. Ознакомиться с методическими рекомендациями.

2. Выполнить задание, согласно номеру своего варианта. Исходные данные необходимо оформить в виде массива. При выполнении задания ввод исходных данных и вывод результатов сопровождать комментариями (какие данные нужно ввести и что получается в результате).

Составить блок-схему программы. Оформить отчет

3. Составьте отчет в соответствии с заданием.

4. Сделать выводы по выполненной работе.

Методические указания

В практике программирования часто встречаются задачи, в которых требуется применение структурированной информации: таблицы, результаты наблюдений, проекции векторов, числовые матрицы, каталоги библиотек и т.д. Для работы с такими данными практически во всех языках программирования существует понятие массива.

Массив – это регулярная структура данных, которая состоит из пронумерованных компонент одного и того же типа. Этот тип мы будем называть *базовым типом*.

Массивы могут быть одномерными:

A ₁	A ₂	A ₃	A ₄	A ₅	A ₆	A ₇	A ₈
----------------	----------------	----------------	----------------	----------------	----------------	----------------	----------------

и многомерными (например, двумерными):

A ₁₁	A ₁₂	A ₁₃	A ₁₄
A ₂₁	A ₂₂	A ₂₃	A ₂₄
A ₃₁	A ₃₂	A ₃₃	A ₃₄

С точки зрения машинной реализации, все массивы – одномерные, разница лишь в том, как пронумерованы элементы массива.

Описание одномерного массива, если считать его элементы целыми числами выглядит следующим образом:

A : array [1..8] of integer;

здесь **array** – ключевое слово, которое и обозначает собственно массив, в квадратных скобках указан *диапазон* первого и единственного *индекса*.

В Pascal'е в качестве диапазона индекса может выступать любой отрезок перечислимого типа, например 'А'..'Н', либо 0..7. Однако на практике чаще всего удобнее в качестве индексов использовать отрезок целого типа, причем нижний (меньший) индекс разумно выбирать единицей или нулем.

Одной из самых неприятных ошибок программирования – является ошибка обращения к несуществующему элементу массива, или как говорят, ошибка выхода индекса за допустимый диапазон. Поэтому предыдущее определение массива А лучше переписать так:

```
Const N = 8;
```

```
Var A : array [1..N] of integer;
```

и в дальнейшем в программе при работе с массивом использовать не конкретные числа, а *константы*, которые определяют диапазон индексов, кроме того, программу можно будет легко модифицировать для работы с массивом другой размерности, так как необходимо будет изменить всего лишь одну строчку!

Иногда формальность описания следует развить, выделив *описание типа отдельно*, это будет абсолютно необходимо, если вы собираетесь использовать в процедурах и функциях параметры-массивы.

```
Const N = 8;
```

```
Type TA = array [1..N] of integer;
```

```
Var A : TA;
```

Дополнительные удобства этого подхода заключаются в том, что массивы, описанные в разных местах как массивы типа TA, будут являться совместимыми по типу, а в случае описания массивов А и В одинаковым способом, но без объявления типамассива, они будут считаться несовместимыми. Например,

```
Const N = 8;
```

```
Type TA = array [1..N] of integer;
```

```
Var A : TA;
```

```
Var B : TA;
```

здесь А и В – массивы одного и того же типа. А здесь:

```
Const N = 8;
```

```
Var A : array [1..N] of integer;
```

```
Var B : array [1..N] of integer;
```

здесь А и В – массивы будут считаться разных типов. Хотя следующее описание определяет массивы одинаковых типов:

```
Const N = 8;
```

```
Var A,B : array [1..N] of integer;
```

В качестве базового типа допустим абсолютно любой тип, в том числе и массив, т.е. допустим массив массивов:

```
Const M = 5;
```

```
N = 8;
```

```
Var A : array [1..M] of array [1..N] of integer;
```

Подобная ситуация встречается довольно часто, поэтому для нее существует разумное сокращение:

```
Const M = 5;  N = 8;  
Var  A : array [1..M,1..N] of integer;
```

Следует учесть, что многомерные массивы, даже при небольших диапазонах индексов имеют тенденцию занимать много памяти.

Рассмотрим выполнение элементарных манипуляций с массивами. Самая простая задача – заполнение всех элементов одним и тем же значением:

```
{Инициализация массива} for i:=1 to N do A[i]:=0;
```

Обратите внимание, как осуществляется доступ к элементам массива – после имени массива в квадратных скобках указывается индекс, который может быть произвольным выражением, лишь бы его значение не выходило за указанный при описании диапазон. Подобная конструкция допустима везде, где допустима простая переменная.

Цикл **for** – чрезвычайно удобная и полезная вещь при работе с массивами. Оператор вида **for** i:=1 **to** N **do** – можно «переводить» как «выполнить для всех элементов массива».

Если два массива одного типа, то допустимо присваивание одного массива другому одним оператором:

```
B:=A;
```

Следующие два примера показывают, как осуществить ввод-вывод с небольшим сервисом:

```
{ввод массива} for i:=1 to N do begin  
  write('Введите ',i,'-й элемент: '); readln(A[i]) end;
```

Из этого примера видно, что массив вводится поэлементно, и как организовать нехитрый сервис. Вывод производится аналогично:

```
{вывод массива}  
for i:=1 to N do writeln('A/',i,'/',A[i]);
```

Теперь рассмотрим самую первую нашу задачу на обработку массива – поиск максимального элемента. Поступим следующим образом: пусть максимальный элемент массива – первый, заведем для него специальную переменную; затем будем просматривать поочередно последующие элементы, и если окажется, что нам встретится элемент больший, чем уже определенное число, то заменим его на этот элемент массива. Таким образом, когда мы просмотрим весь массив, окажется, что наша переменная содержит искомое значение:

```
{определение максимального значения} max:=A[1];  
for i:=2 to N do if A[i]>max then max:=A[i]; writeln('Maximum=',max);
```

Напишем теперь целиком программу, используя полученные знания:

```
Program Massiv;  
Const N = 10;  
Var  
  A : array [1..N] of integer;  
  i, max : integer;  
begin  
  for i:=1 to N do // Ввод массива  
    begin  
      write('Введите ',i,'-й элемент: ');  
      readln(A[i])
```

```

    end;
    max:=A[1]; // Поиск максимального значения
    for i:=2 to N do if A[i]>max
        then max:=A[i]; writeln('Maximum=',max);
    end.

```

Пример 1. В автопарке, имеющем 18 машин марки КАМАЗ, каждый из КАМАЗов перевез за день определенный объем груза. Определить суммарный объем перевозок грузов за день. При решении задачи будем использовать тип массива КАМАЗ для описания всех КАМАЗов автопарка; переменную P[i] для описания объема груза, перевезенного i-той машиной за день (I меняется от 1 до 18). Блок-схема алгоритма для решения данной задачи будет выглядеть следующим образом (см. **Рис. 1**):

Текст следующий вид:

```

Program Pr1;
    Uses wincrt;
    Type КАМАЗ=array [1..18] of real; Var
    i:integer;
    p:КАМАЗ;
    S:real;
Begin
    S:=0;
    For i:=1 to 18 do
        Begin
            Writeln('Введите объем перевозок',I,'-ой машины, т');
            Readln(p[i]);
            S:=S+p[i];
        End;
    Writeln('Суммарный объем перевозок S=',S:8:2,'т');
End.

```

Накопление суммы в данном примере будет проводиться по шагам, при вводе значения объема перевозок для очередной машины сумма будет увеличиваться на данную величину. Аналогично реализуется и алгоритм нахождения произведения элементов массива (с заменой начального значения суммы S:=0 на начальное значение произведения S:=1 и с заменой операции сложения элементов массива «+» на операцию умножения «*»).

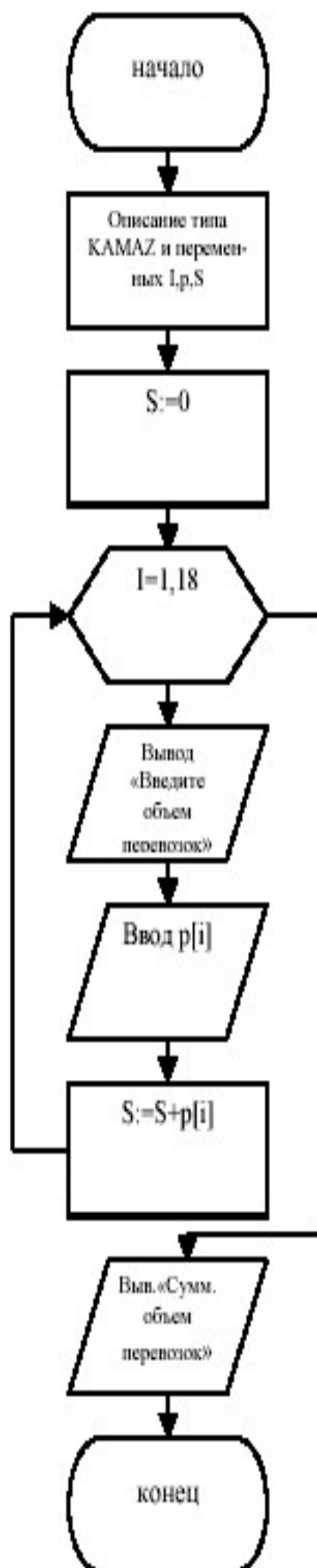


Рис. 1 Блок-схема программы для Примера 1
Нахождение количества элементов массива, удовлетворяющих заданному условию

Задача 2. Известны результаты экзамена по информатике одной группы из 22 студентов. Определить, сколько студентов сдали экзамен на 4 и 5. Один из вариантов решения этой задачи следующий:

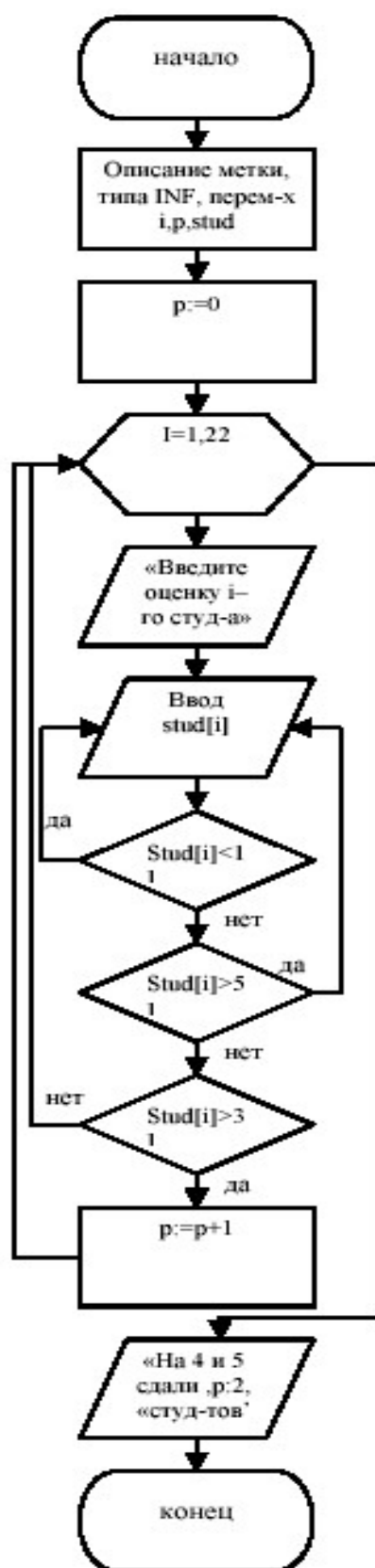


Рис. 2 Блок-схема программы для Примера 2

На Рис. 2 представлена блок-схема алгоритма поставленной задачи. Текст программы:

```
Program pr3;  
Uses wincrt;  
Label 1;  
  
    Type INF=array[1..22] of integer;  
    Var  
    stud:INF;  
    i,p:integer;  
    begin  
    p:=0;  
    for i:=1 to 22 do  
  
begin  
    1: writeln('Введите оценку ',i,'-го студента');  
    readln(stud[i]);  
    if (stud[i]<1) or (stud[i]>5)  
    then goto 1;  
        if stud[i]>3 then p:=p+1;  
        end;  
    writeln('На 4 и 5 сдали экзамен ',p:2,' студентов');  
    end.
```

В данной программе для обозначения списка оценок по информатике использовался тип массива INF, для обозначения оценок конкретных студентов – переменная stud. Программа предусматривает проверку корректности вводимых данных: при попытке ввода несуществующей по пятибалльной системе оценки, программа повторяет ее ввод. Для этого используется оператор перехода GOTO, где имя метки, к которой осуществляется переход (в данном случае 1), описывается в разделе описания меток LABEL.

Сортировка массива по возрастанию

Задача 3. Предположим, известны результаты соревнований по стрельбе, в которых принимали участие 9 человек. Расположить данные результаты в порядке возрастания набранных при стрельбе очков.

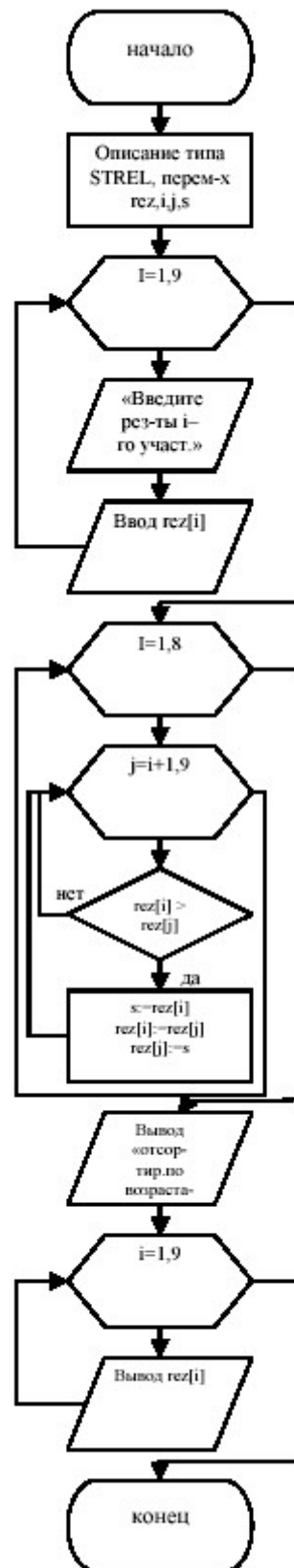


Рис. 3. Блок-схема программы для Примера 3

Алгоритм решения данной задачи является наиболее сложным из приведенных выше примеров и требует использования вложенных циклов.

Один из способов сортировки массивов заключается в следующем.

Сначала первый элемент массива в цикле сравнивается по очереди со всеми оставшимися элементами. Если очередной элемент массива меньше по величине, чем первый, то эти элементы переставляются местами. Сравнение продолжается далее уже для обновленного первого элемента. В результате окончания данного цикла будет найден и установлен на первое место самый наименьший элемент массива. Далее продолжается аналогичный процесс уже для оставшихся элементов массива, т.е. второй элемент сравнивается со всеми остальными и, при необходимости, переставляется с ними местами. После определения и установки второго элемента массива, данный процесс продолжается для третьего элемента, четвертого элемента и т.д. Алгоритм завершается, когда сравниваются и упорядочиваются предпоследний и последний из оставшихся элементов массива. Блок-схема представлена на Рис. 3

Программа реализации изложенного алгоритма может иметь следующий вид:

```
Program pr4;  
Uses crt;  
Type STREL=array[1..9]of integer; Var  
rez:strel; i,j,s:integer;  
Begin  
  For i:=1 to 9 do  
    begin  
      writeln('Введите результаты ',i,'-го участника'); readln(rez[i]); end; for i:=1 to 8 do  
      for j:=i+1 to 9 do if rez[i]>rez[j] then  
        begin  
          s:=rez[j]; rez[j]:=rez[i]; rez[i]:=s;  
        end;  
      writeln('Отсортированные по возрастанию результаты:'); for i:=1 to 9 do write  
      (rez[i]:5,' '); end.
```

Здесь STREL – тип массива результатов стрельбы участников, rez[i] – переменная для описания результатов i-го участника (i

рис. 3. Блок-схема меняется от 1 до 9).

Вспомогательная переменная s используется для перестановки местами элементов массива.

Варианты заданий

Выполните задание по варианту, назначенному преподавателем.

1. Подсчитать среднемесячную зарплату сотрудника предприятия.
2. Дан объем продукции, выпущенной заводом за год (помесячно). Найти наименьший объем. В качестве результата вывести номер месяца и объем выпущенной продукции.
3. Курс доллара в течение года менялся в диапазоне от 63 руб. до 70руб. Найти наибольшее значение курса доллара. В качестве результата вывести номер месяца и значение курса доллара.
4. Известен месячный план выпуска некоторой продукции и объемы выпущенной этой продукции заводом за год (помесячно). Определить, когда завод не выполнил план. Результат получить в виде: номер месяца и объем выпущенной продукции.
5. Даны результаты сдачи экзамена по информатике группы студентов (в группе 20 студентов). Подсчитать количество студентов, не сдавших экзамен.
6. Известна среднемесячная зарплата 10 сотрудников отдела. Расположить данные в порядке убывания.
7. Известны данные по продаже компьютеров в течение недели. Найти общее количество проданных компьютеров.
8. Известны данные по продаже компьютеров в течение недели. Расположить эти данные в порядке возрастания.
9. Даны результаты сдачи экзамена по информатике группы студентов (в группе 20 студентов). Подсчитать количество студентов, сдавших экзамен без троек.
10. Даны результаты сдачи экзамена по информатике группы из 15 студентов. Подсчитать количество студентов, не сдавших экзамен, в численном и в процентном соотношении.

Контрольные вопросы:

1. Ответить письменно на вопросы
2. Что понимают под массивом данных?
3. Что называют размерностью массива?
4. Что понимают под индексом элемента массива?
5. Какой массив называется одномерным?
6. Приведите примеры одномерных массивов.
7. Как описываются одномерные массивы на языке PASCAL?
8. Как задается диапазон изменения индексов массива?
9. Как обозначаются индексы массивов на языке PASCAL?

Оформление результатов работы

1. Оформить работу в соответствии с заданиями.
2. Ответить на контрольные вопросы.
3. Сформулировать выводы по результатам работы.
4. Сдать и защитить работу.