Масиви в Паскал

Публикувано от **elena_pazardjik** на **22.08.2012**

1. Масиви в PASCAL.

Данните, с които работи една програма, са единични (*скаларни*) данни и структури от данни. Единични данни са число, буква или логическа стойност. Скаларни типове са вградените типове **integer, real, char** и **boolean**, а също и тип диапазон.

Дефиниция 1

Структури от данни са организирани и подредени по някакви правила съвкупности от данни. Те биват статични и динамични. *Статични* са такива структури, чието организация и брой елементи се задава при тяхната дефиниция и не може повече да се променя.

Езикът Pascal подържа следните вградени структури:

- масив,

- низ от символи,

- записи

- файл.

Дефиниция 2

Масив е крайна, наредена съвкупност от еднотипни данни.

Еднотипните елементи на масива могат да бъдат, както скаларни данни, така и структури от данни с изключение на файлове. Те могат да са, както от вградените в езика типове, така и от типове дефинирани от програмиста. Според наредбата на елементите си масивите биват едномерни, двумерни, тримерни и т. н. Масивът е статична структура от данни. Броят на елементи му се задава при неговата декларация и трябва да бъде константен. Могат да се дефинират типове-масиви и направо променливи-масиви.

2. Видове масиви:

2.1. Едномерни масиви.

Променлива едномерен масив се дефинира по следния начин

<име>: ***array*** *[целочислен_диапазон]* ***of*** *<тип>;*

където целочислен_диапазон определя, от къде до къде се изменят номерата (индексите) на елементите на масива, а тип определя типа на тези елементи. Границите на диапазона трябва да са целочислени константи и от тях и от типа компилатора определя, колко памет да отпусне за масива. Едномерният масив отговаря на математическото понятие вектор.

Пример .3:

```
const
max = 26;
type
nomer = 1 .. max;
uspeh = array [ nomer ] of real;
var
a: uspeh;
b, c: array[-3 .. 4] of real;
```

Масивът a е от успехите на 26 ученика, а масивите b и c са от по 8 реални числа. Вътре в тялото на

програмата се работи с елементите на масива по същия начин както с променливи от съответния тип. Обръщението към отделните елементи става по следния начин

<име>[<индекс>]

където име е името на променливата масив, а индекс е целочислен израз със стойност в диапазона на номерацията на неговите елементи. Програмистът внимателно трябва да следи индекса да не излиза извън рамките на този диапазон, защото в едни случаи това води до прекъсване на програмата, а в други - до погрешна работа на програмата.

Пример 4 (виж Пример 3):

```
var
i, j: integer;
begin
.....
readln(a[2 * i + j - 1]); { стойността на 2*i + j - 1 трябва да от 1 до 26 }
b[-2] := -5.3;
c[5] := b[-2] / 4; { Грешка ! Индекс 5 е извън диапазона на c. }
for i := 1 to max dowriteln(a
);
.....
end.
```

Една от най-често използваните дейности с масиви е обхождане на масив. Обхождане на една структура е достигане с някаква цел до всеки неин елемент. Обхождането на едномерен масив става чрез цикъл - обикновено **for**. В края на последния пример се обхожда масива a, с цел извеждане на екрана всички негови елементи.

Забележка. *sqr е библиотечна функция за повдигане на квадрат.*

2.2. Двумерен масив.

В този масив елементите са подредени по редове и стълбове. Той отговаря на математическото понятие матрица или таблица. Променлива двумерен масив се декларира по следния начин

<име>: ***array*** *[<целочислен_диапазон1>, <целочислен_диапазон2>]* ***of*** *<тип>;*

където целочислен_диапазон1 задава диапазона на номерация на редовете на масива, а целочислен_диапазон2 - на стълбовете.

Пример 5:

```
const
      max = 26;
   type
      predmet=(BEL,  Mat, IT, Inf, RE, AE, FVS)
      nomer = 1 .. max;
      ocenki= array [nomer, predmet ] of 2..6;
   var
      m: ocenki;
       x, y: array [-3 .. 4, -1 .. 9] of real;
```

Масивът m е от оценките на 26 ученика по предметите БЕЛ, Математика, ИТ, Информатика, Руски език, Английски език, Физическо възпитание и спорт, а масивите x и y са от по 88 реални числа. Обръщението към елемент но двумерен масив става чрез два индекса:

име[индекс1, индекс2]

където индекс1 и индекс2 са целочислени изрази със стойности съответно в целочислен_диапазон1 и целочислен_диапазон2. И в този случай програмистът трябва да следи тези стойности да не напускат съответните диапазони.

Пример 6 (виж Пример 6.5):

```
 var
      i, j: integer;
   begin
```

```
.....
m[25, IT] := 5;      {оценката на 25 номер по ИТ е 5}
i := 10;
j := 15;
x[i - 7, j div 5 -1] := 3.43;
 for i := -3 to 4 do
     for j := -1 to 9 do
        readln( y[i, j\);
.....
end.
```

Обхождането на двумерен масив става чрез двоен цикъл т. е. цикъл вътре в тялото на друг цикъл. В последния пример имаме обхождане на масива y с цел въвеждане на стойности на елементите му от клавиатурата. Обхождането на двумерни масиви става по два начина: по редове и по стълбове. В горния пример имаме обхождане по редове, където във външния цикъл се изменя първия индекс (за редовете), а във вътрешния цикъл се изменя втория индекс (за стълбовете). Т. е. за i-то завъртане на външния цикъл имаме i-ти ред и с вътрешният цикъл преминаваме през елементите на този ред. При обхождане по стълбове във външния цикъл се изменя втория индекс, а във вътрешния - първия.

2.3. Тримерни масиви.

Те се декларират и ползват подобно на двумерните. При деклариране си имат три диапазона за изменение на индексите си. При ползване към техните елементи се обръщаме с три индекса. Тримерните масиви се обхождат с троен цикъл.

В този курс ще разглеждаме едномерните масиви.

3. Основни операции с масиви

- Въвеждане на елементите на масив

- Извеждан е на елементите на масив
- Намиране на сумата на елементите
- Намиране на средната стойност на елементите

Пример 7. В ПГ “Васил Левски” има 26 ученика в 9-ти клас. Да се състави програма, която:

- въвежда успехите на учениците от 9-ти клас в масив;
- Извежда ги в колонка;
- Намира средния успех;

```
Program  klas_9;

const
      max = 26;
   type
      nomer = 1 .. max;
      uspeh = array [ nomer ] of real;
   var
      a: uspeh; i:nomer; S, Sr:real;
       Begin
       {Въвеждане на масива}
       For  i 1 to max do
             Begin
                   Write(‘Въведи успеха на ‘, i,’ номер’);
                   Read(a); Writeln
             End;
     {Извеждане  на масива}
     For  i 1 to max do writeln(a:7:2);
{Намиране на сумата от елементите на масива}
     S 0; For i 1 to max do S S+a;
{Намиране на средния успех}
```

```
    Sr S/max;
Write('средния успех на 9-ти клас е:', Sr:7:2);
End.
```

4. Намиране на минимален и максимален елемент в масив

За да намерим минималния (максималния) елемент на масив използваме алгоритъм, който се основава на следната идея:

1. Последователно се преглеждат елементите, като най-малката (най-голямата) стойност се помни в променливата min (max);
2. Всеки елемент се сравнява с min (max) и ако е по-малък (по-голям) от него стойността на min (max) се променя и става равна на този елемент;
3. След като се обходят всички елементи се намира най-малкия (най-големия) елемент на масива
4. За начална стойност на min (max) обикновено се взема първия елемент на масива

Пример: да се намери максималния успех на учениците от 9 клас и да се изведе номера на съответния ученик.

```
Program  max_uspeh;

const
      max_nom = 26;
   type
      nomer = 1 .. max_nom;
      uspeh = array [ nomer ] of real;
   var
      a: uspeh; i, nom :nomer; max:real;
    Begin
      {Въвеждане на масива}
      For  i 1 to max_nom do
            Begin
                  Write('Въведи успеха на ', i,' номер');
```

```
                    Read(a); Writeln
                End;
        {Намиране на максималния успех}
        max a[1]; nom 1;
For  i 2 to max_nom do
        if  a>max  then
  begin
     max a; nom I;
  end;
Writeln('Най-висок успех ', max, 'има ученик с номер ',nom:3);
End.
```

5. Сортиране на масив.

2. ***Същност*** - всеки процес на подреждане на елементите на едно множество в определен ред се нарича сортировка. Особено място в този процес заемат броят на операциите, броя сравнения между елементите(n^2)
2. ***Видове***

В намаляващ ред /от най-големия елемент до най-малкия/

В нарастващ ред /от най-малкия елемент до най-големия/

3. Методи на сортиране

Метод на прякото вмъкване;

Метод на мехурчето;

Метод на пряката селекция

Двоично търсене и други

Ще разгледаме метода на пряката селекция

Метод на пряката селекция- при този метод се търси елемента с най-малка (най-голяма) стойност и той се записва на първо място. После се търси втория най-малък (най-голяма) елемент и се поставя на второ място и т.н.

Пример: Да се подредят средните успехите на учениците от девети клас по-големина

Program sort_uspeh;

```
const
        nom = 26;
    type
        nomer = 1 .. nom;
        uspeh = array [ nomer ] of real;
    var
        a: uspeh; i, j, k :nomer; min:real;
Begin
         {Въвеждане на масива}
         For  i 1 to nom do
                  Begin
                           Write('Въведи успеха на ', i,' номер');
                           Read(a); Writeln
                  End;
         {Сортиране на елементите}
         For  i 1 to nom-1  do
            Begin
min a;  k i;
For  j i+1 to nom do                          if  a[j]
```

6. Търсене в масив.

Често срещана практическа задача е задачата за търсене в множество от определени данни.

Задачата за търсене в общия случай изисква да се намери(отдели) един или няколко елемента в съвкупност от елементи така , че те да отговарят на предварително зададено условие,наречено критерий на търсене

Алгоритъма за последователно търсене в масив се основава на следната идея:

1) *Последователно се обхождат всички елементи на масива*

2) *Всеки елемент се проверява дали отговаря на поставения критерий за търсене*

3) *Като резултат елементът x може да бъде намерен или не*

Пример: Да се намери средния успех на отличниците.

Program otlichnici;

```
 const
       nom = 26;
    type
       nomer = 1 .. nom;
       uspeh = array [ nomer ] of real;
    var
       a: uspeh; i, Br :nomer;
Begin
        {Въвеждане на масива}
        For  i 1 to nom do
                Begin
                        Write('Въведи успеха на ', i,' номер');
                        Read(a); Writeln
                End;
        {намиране средния успех на отличниците}
        Br:=0;
For  i 1 to nom  do
             if  a>=5.50  then
begin
S:=S+A;
Br:=Br+1
                           end;
                  Sr:=S/Br;
Writeln('Средния успех на отличниците е :' ,Sr:8:2);
      End.
```

Алгоритми за броене

Много близки до задачите за търсене на задачите, които изискват преброяване на елементите на

масив, имащи определени свойства или отговарящи на някакъв критерий. Алгоритмът, по който се решават задачите за броене е много близък до алгоритъма за търсене**:**

1) Предварително на променлива от цял тип Br се присвоява стойност 0(Br := 0).

2) Последователно се обхождат всички елементи на масива, като всеки елемент се проверява дали отговаря на предварително зададения критерий.

3) Ако елемента удовлетворява зададения критерий се „преброява", т.е.стойността на променливата Br се увеличава с 1 (Br:=Br+1)

Пример: Да се намери броя на отличниците в 9 клас

Program Br_uchenici;

```
 const
       nom = 26;
    type
       nomer = 1 .. nom;
       uspeh = array [ nomer ] of real;
    var
       a: uspeh; i, Br :nomer;
Begin
       {Въвеждане на масива}
       For  i 1 to nom do
              Begin
                     Write('Въведи успеха на ', i,' номер');
                     Read(a); Writeln
              End;
       {намиране броя на отличниците}
       Br:=0;
For  i 1 to nom  do
            if  a>=5.50  then Br:=Br+1
                  end;
End.
```

Powered by
Bukvar.bg

© 2010-2024