Программирование на языке Паскаль

Подпрограммы

При разработке программы иногда появляются **повторяемые группы действий** или возникает **необходимость расчленить программу** на функциональные модули, сделать ее структуру иерархической.

Для этого во всех языках программирования используют **подпрограммы**.

Подпрограмма — это специальным образом оформленный алгоритм, который может многократно использоваться при решении более общей задачи.

В Паскале подпрограмма является частью основной программы, ее описание располагается между разделом var главной программы и ее программным блоком (первым begin).

Подпрограмм может быть несколько, их описания располагаются в произвольном порядке одно за другим.

В Паскале различают два вида подпрограмм: ***процедуры*** и ***функции***. Основное различие между ними заключается в том, что ***процедура*** получает в результате своей работы любое количество данных, а ***функция*** — только одно значение.

- ***Глобальные переменные*** - переменные, объявленные в основной программе, доступны всем операторам программы, а так же операторам процедур и функций.

- ***Локальные переменные*** - переменные, объявленные в процедуре или функции. Они доступны только операторам процедур или функций.

Программирование на языке Паскаль

Процедуры

Процедуры

Procedure <имя> (список формальных параметров);

const
type

var

Могут быть опущены

begin
<операторы>;
end;

Вызов процедуры – это упоминание ее имени в тексте основной программы.

Процедуры

Особенности:

- все процедуры расположены **выше** основной программы
- в заголовке процедуры перечисляются **формальные** параметры, они обозначаются именами, поскольку могут меняться

```
procedure Tr( x, y, r, g, b: integer);
```

- при вызове процедуры в скобках указывают **фактические** параметры (числа или арифметические выражения) **в том же порядке**

```
Tr (200, 100, 0, 255, 0);
```

Процедуры

Особенности:

- для каждого формального параметра после двоеточия указывают его тип

```
procedure A (x: real; y: integer; z: real);
```

- если однотипные параметры стоят рядом, их перечисляют через запятую

```
procedure A (x, z: real; y, k, l: integer);
```

Пример

```
procedure A (x, z: real; var l: real);
```

Переменные, которые используются в формуле

Переменная, в которую записывается результат

Процедуры

Особенности:

- в процедуре можно объявлять дополнительные **локальные** переменные, остальные процедуры не имеют к ним доступа

```
program qq;

  procedure A(x, y: integer);
    var a, b: real;
    begin
      a := (x + y)/6;
      ...
    end;
begin
   ...
end.
```

локальные переменные

Параметры-переменные

Задача: составить процедуру, которая меняет местами значения двух переменных.

Особенности:

надо, чтобы изменения, сделанные в процедуре, стали известны вызывающей программе

```
program qq;
var x, y: integer;
  procedure Exchange ( a, b: integer );
  var c: integer;
  begin
    c := a; a := b; b := c;
  end;
begin
  x := 1; y := 2;
  Exchange ( x, y );
  writeln ( 'x = ', x, ' y = ', y );
end.
```

эта процедура работает с **копиями** параметров

```
x = 1 y = 2
```

Программирование на языке Паскаль

Функции

Функции

Функция – это вспомогательный алгоритм (подпрограмма), результатом работы которого является некоторое значение.

3 категории функций:

- **стандартные функции** (abs(x), sqrt(x), sqr(x), sin(x), cos(x), и т.д.);
- **функции программиста** (объявлять свою собственную функцию и в дальнейшем использовать её так же как и стандартную);
- **библиотечные функции** (стандартные библиотечные модули).

Функции

Зачем?

- для выполнения одинаковых расчетов в различных местах программы
- для создания общедоступных библиотек функций

Структура функции

Function <имя> (<параметры>):<тип результата>;

const …;

…..

var … ;

Блок описания локальных переменных

Begin

<операторы>

Имя := выражение;

End;

В разделе операторов должен находится, хотя бы один оператор, присваивающий имени функции значение.

Функции

Задача: составить функцию, которая вычисляет наибольшее из двух значений, и привести пример ее использования

Функция:

формальные параметры

```
function Max (a, b: integer): integer;
begin
  if a > b then Max := a
  else          Max := b;
end;
```

это результат функции

Функции

Особенности:

- заголовок начинается словом **function**

```
function Max (a, b: integer): integer;
```

- формальные параметры описываются так же, как и для процедур

```
function qq( a, b: integer; x: real ): real;
```

- можно использовать параметры-переменные

```
function Max ( var a, b: integer): integer;
```

- в конце заголовка через двоеточие указывается **тип результата**

- функ

```
function Max (a, b: integer): integer;
```

ммы

Функции

Особенности:

- можно объявлять и использовать **локальные переменные**

```
function qq (a, b: integer): float;
var x, y:
real;
begin
   ...
end;
```

- значение, вычисленное в функции, возвращается в переменную, имя которой совпадает с **названием функции**; объявлять ее **НЕ НАДО**:

```
function Max (a, b: integer): integer;
begin
   ...
   Max :=
   a;
end;
```

Программа

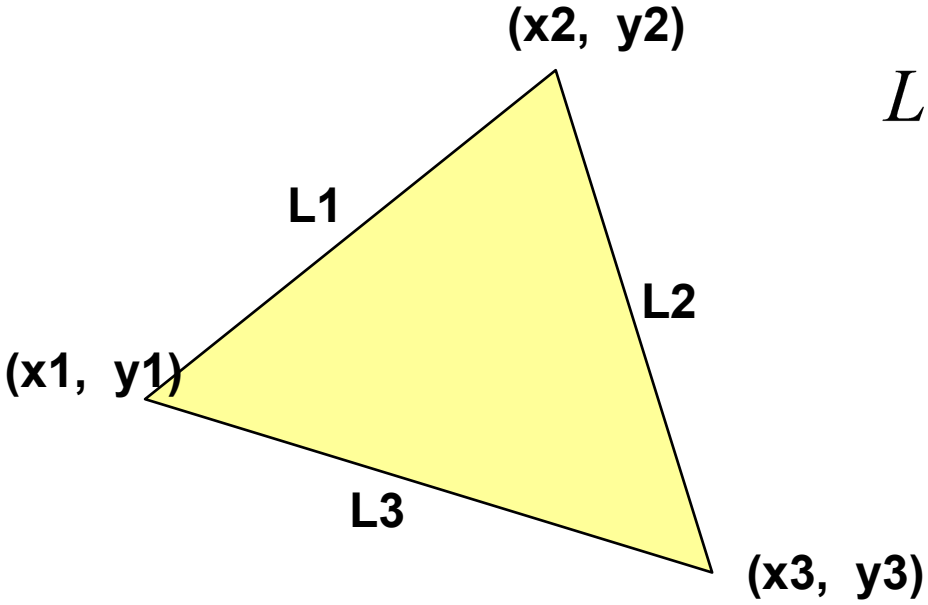
```
program qq;
var a, b, c : integer;
    function Max (a, b: integer): integer;
    begin
      ...
    end;
begin
  writeln('Введите два числа');
  read(a, b);
  c := Max ( a, b );
  writeln('Наибольшее число ', c );
end.
```

фактические параметры

вызов функции

Имена переменных, функций и процедур не должны совпадать!

Найти S и P треугольника, вершины которого заданы координатами



$$L1 = \sqrt{(x1 - x2)^2 + (y1 - y2)^2}$$

$$p = \frac{a + b + c}{2}$$

полупериметр

$$S = \sqrt{p * (p - L1) * (p - L2) * (p - L3)}$$

Решаем используя Процедуру

```
var    x1,y1,x2,y2,x3,y3:real;
       P,S,PP,l1,l2,l3:real;

Procedure   dl(a,b,c,d:real;  var l:real);
  begin
     l:=sqrt(sqr(a-c)+sqr(b-d));
  end;

begin
   read(x1,y1,x2,y2,x3,y3);
   dl(x1,y1,x2,y2,l1);
   dl(x2,y2,x3,y3,l2);
   dl(x3,y3,x1,y1,l3);
   P:=l1+l2+l3;
   PP:=P/2;
   S:=sqrt(PP*(PP-l1)*(PP-l2)*(PP-l3));
   writeln(P:10:2,S:10:2);
end.
```

Решаем используя Функцию

```
var    x1,y1,x2,y2,x3,y3:real;
       P,S,PP,l1,l2,l3:real;

function   dl(a,b,c,d:real):real;
  begin
     dl:=sqrt(sqr(a-c)+sqr(b-d));
  end;

begin
   read(x1,y1,x2,y2,x3,y3);
   l1:=dl(x1,y1,x2,y2);
   l2:=dl(x2,y2,x3,y3);
   l3:=dl(x3,y3,x1,y1);
   P:=l1+l2+l3;
   PP:=P/2;
   S:=sqrt(PP*(PP-l1)*(PP-l2)*(PP-l3));
   writeln(P:10:2,S:10:2);
end.
```

Решаем используя Процедуру

```
var  k,a,f1,s:integer;
procedure   fact(n:integer; var f:integer);
                  var i:integer;
    begin
        f:=1;
        for i:=1 to n do
        f:= f*i;
    end;
begin
  read(k);
  a:=2;
  while a<=k do
     begin
      fact(a,f1);
      s:=s+f1;
      a:=a+2;
     end;
  writeln(s);
end.
```

Решаем используя Функцию

```
var  k,a,f1,s:integer;

function fact(n:integer):integer;
      var f,i:integer;
  begin
    f:=1;
    for i:=1 to n do
    f:= f*i;
    fact:=f;
  end;
```

```
begin
  read(k);
  a:=2;
  while a<=k do
     begin
      f1:=fact(a);
      s:=s+f1;
      a:=a+2;
     end;
  writeln(s);
end.
```