Программирование на языке Паскаль

1. **Процедуры**
2. **Функции**

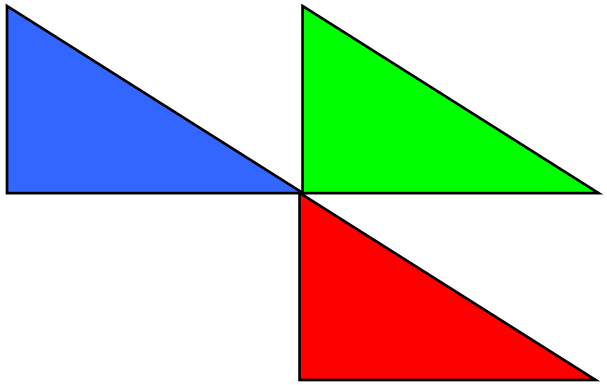
Ст. преподаватель
Швидченко С.А.

Программирование на языке Паскаль

Тема 1. Процедуры

Процедуры

Задача: Построить фигуру:



Можно ли решить известными методами?

Особенность: три похожие фигуры.

общее: размеры, угол поворота

отличия: координаты, цвет

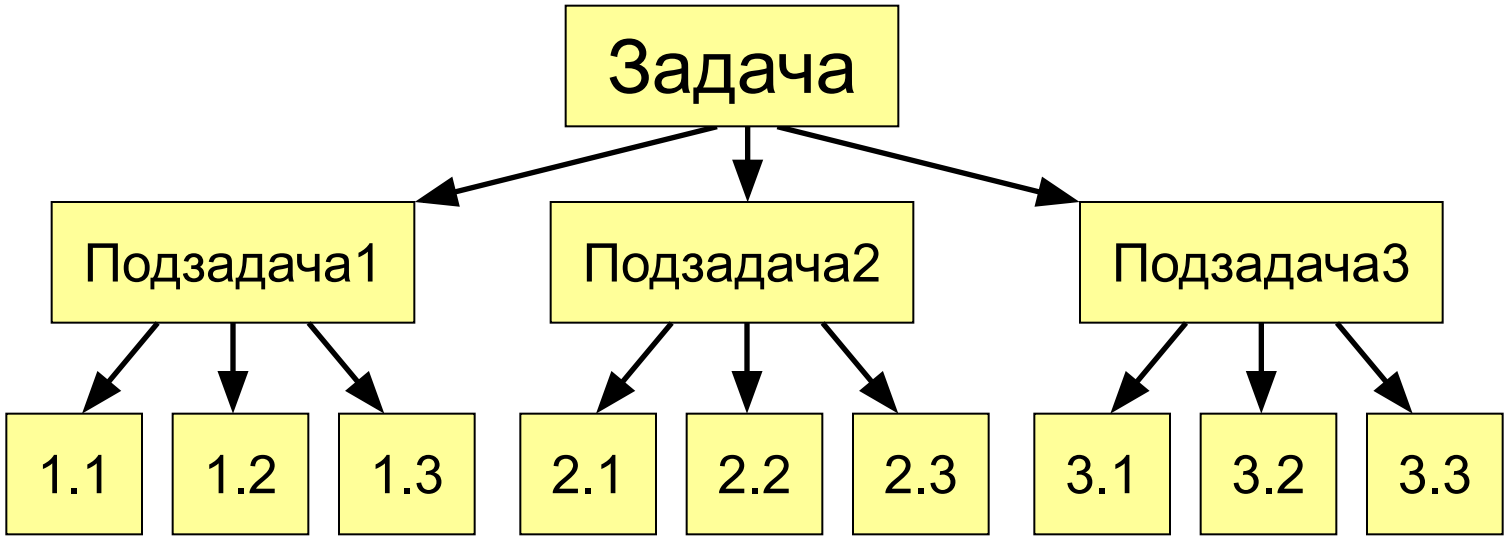
Сколько координат надо задать?

Процедуры

Процедура – это вспомогательный алгоритм, который предназначен для выполнения некоторых действий.

Применение:

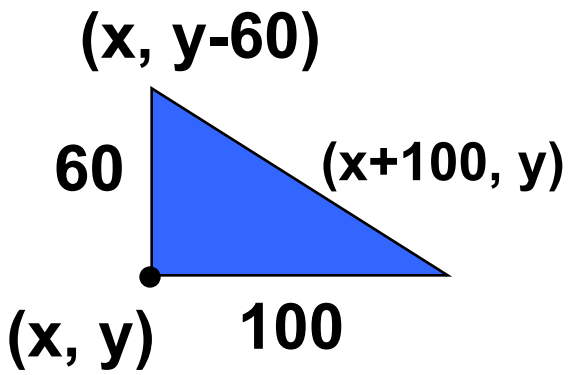
- выполнение одинаковых действий в разных местах программы
- разбивка программы (или другой процедуры) на подзадачи для лучшего восприятия



Процедуры

Порядок разработки:

- выделить одинаковые или похожие действия (три фигуры)
- найти в них **общее** (размеры, форма, угол поворота) и **отличия** (координаты, цвет)
- отличия записать в виде неизвестных переменных, они будут **параметрами** процедуры



заголовок

параметры

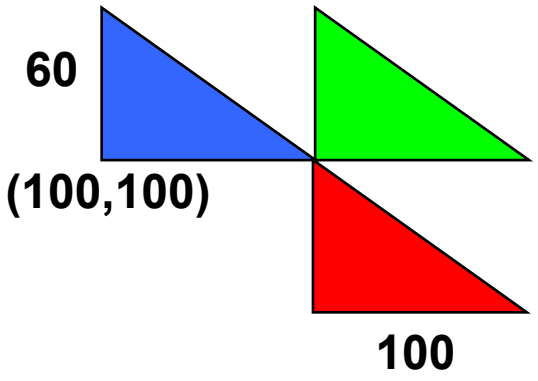
```
procedure Tr( x, y, r, g, b: integer);
begin
    MoveTo(x, y);
    LineTo(x, y-60);
    LineTo(x+100, y);
    LineTo(x, y);
    Brush(1, r, g, b);
    Fill(x+20, y-20);
end;
```

цвет

координаты

тело процедуры

Программа



формальные параметры

```
program qq;
 procedure Tr( x, y, r, g, b:
 integer);
 begin
    ...
 end;
begin
    Pen(1, 255, 0, 255);
    Tr(100, 100, 0, 0, 255);
    Tr(200, 100, 0, 255, 0);
    Tr(200, 160, 255, 0, 0);
end.
```

процедура

вызовы процедуры

фактические параметры

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);
```

- внутри процедуры параметры используются так же, как и переменные

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
```

Параметры-переменные

параметры могут изменяться

```
procedure Exchange ( var a, b: integer );
var c: integer;
begin
  c := a; a := b; b := c;
end;
```

Применение:

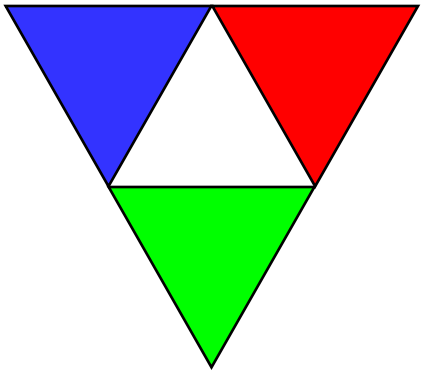
таким образом процедура (и функция) может возвращать несколько значений,

Запрещенные варианты вызова

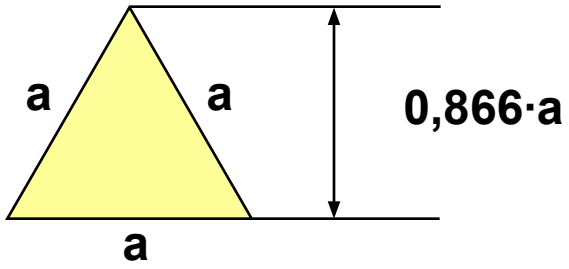
```
Exchange ( 2, 3 );        { числа }
Exchange ( x+z, y+2 );    { выражения }
```

Задания

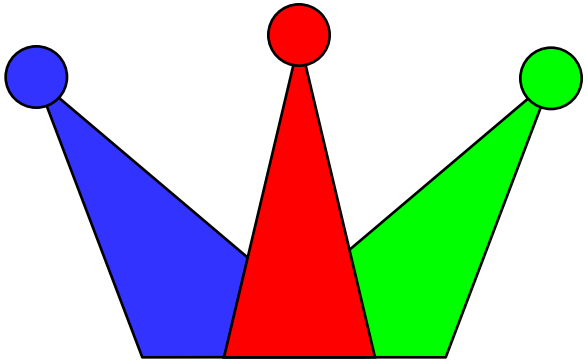
«4»: Используя процедуры, построить фигуру.



равносторонний треугольник



«5»: Используя процедуры, построить фигуру.



Программирование на языке Паскаль

Тема 2. Функции

Функции

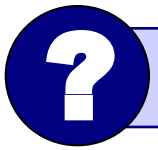
Функция – это вспомогательный алгоритм (подпрограмма), результатом работы которого является некоторое значение.

Примеры:

- **вычисление** $\sin x$, $\cos x$, $\sqrt{x}$
- **расчет значений по сложным формулам**
- **ответ на вопрос (простое число или нет?)**

Зачем?

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



В чем отличие от процедур?

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

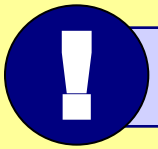
Особенности:

- можно объявлять и использовать **локальные переменные**

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

- знач вается в
  пере **функции**; объявлять ее **НЕ НАДО**:

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



В *Delphi:* `Result := a;`

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!

Задания

«4»: Составить функцию, которая определяет сумму всех чисел от 1 до N и привести пример ее использования.

Пример:

```
Введите число:
100
сумма = 5050
```

«5»: Составить функцию, которая определяет, сколько зерен попросил положить на N-ую клетку изобретатель шахмат (на 1-ую – 1 зерно, на 2-ую – 2 зерна, на 3-ю – 4 зерна, …)

Пример:

```
    Введите номер клетки:
28
На 28-ой клетке 134217728 зерен.
```

Задания (вариант 2)

«4»: Составить функцию, которая определяет наибольший общий делитель двух натуральных и привести пример ее использования.

Пример:

```
Введите два числа:
14 21
НОД(14,21)=7
```

«5»: Составить функцию, которая вычисляет функцию синус как сумму ряда (с точностью 0.001)

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots$$

x **в радианах!**

Пример:

```
Введите угол в градусах:
45
sin(45) = 0.707
```

Логические функции

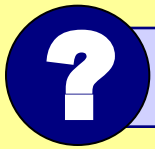
Задача: составить функцию, которая определяет, верно ли, что заданное число – простое.

Особенности:

- ответ – логическое значение (`True` или `False`)
- результат функции можно использовать как логическую величину в условиях (`if, while`)

Алгоритм: считаем число делителей в интервале от 2 до N-1, если оно не равно нулю – число составное.

```
count := 0;
for i := 2 to N-1 do
  if N mod i = 0 then
    count := count + 1;
if count = 0 then
     { число N простое}
else { число N составное }
```



Как улучшить?

Логические функции

```
program qq;
var N: integer;
  function Prime (N: integer): boolean;
  var count, i: integer;
  begin
    i := 2; count := 0;
    while i*i <= N do begin
      if N mod i = 0 then count := count + 1;
      i := i + 1;
    end;
    Prime := (count = 0);
  end;
begin
  writeln('Введите целое число');
  read(N);
  if Prime(N) then
       writeln(N, ' - простое число')
  else writeln(N, ' - составное число');
end.
```

результат – логическое значение

перебор только до $\sqrt{N}$

условие – это логическое значение

вызов функции

Задания

«4»: Составить функцию, которая определяет, верно ли, что сумма его цифр – четное число.

Пример:

```
Введите число:          Введите число:
136                     245
Сумма цифр четная.      Сумма цифр нечетная.
```

«5»: Составить функцию, которая определяет, верно ли, что в заданном числе все цифры стоят по возрастанию.

Пример:

```
   Введите число:       Введите число:
258                     528
Верно.                  Неверно.
```