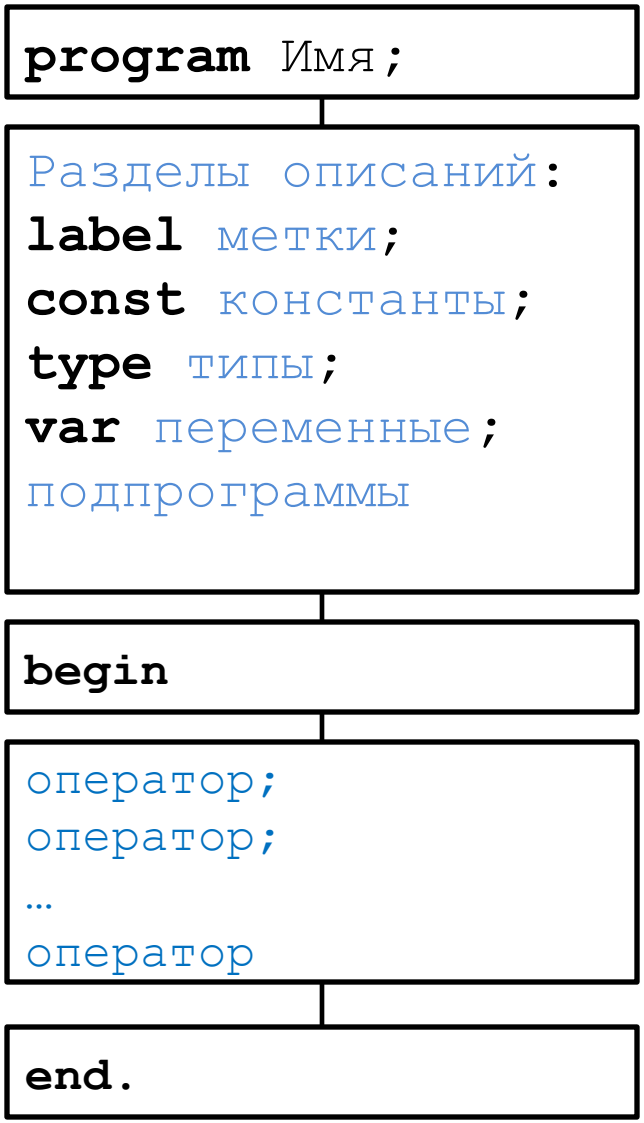
Основы программирования: язык Pascal

Структура программы
Операторы

Структура программы

- Структура программы



Метки

```
label
  метка1, метка2, метка3;
```

Метками помечаются операторы программы, к которым можно перейти оператором goto.

Пример:

```
program MyProg1;
label
  lab1;
begin
  if 1 <= 2 then
    goto lab1;
  writeln('Hello, World!');
  lab1: writlen ('Hello, Boris!');
end.
```

Константы

```
const
   Имя1 = значение;
   Имя2 = значение;…
```

Константа – именованный объект программы, имеющий неизменное значение.

Пример:

```
program MyProg2;
const
  n = 5;
  r = 1.5;
  m : byte = n - 1;
begin
  writeln(n, r, m);
end.
```

Типы

```
type
  ИмяТипа1 = ОпределениеТипа1;
  ИмяТипа2 = ОпределениеТипа2;…
```

Тип описывает структуру данных, определяется множеством значений, операциями и способом представления в памяти.

Пример:

```
program MyProg3;
type
  Int = integer;
  Vector = array[1..10] of real;
var
  i, j : Int;
  a : Vector;
begin
end.
```

Переменные

```
var
   Имя1, Имя2, … : Тип1;
   Имя3, Имя4, … : Тип2;
```

Переменная – именованный объект программы определенного типа, значение которого можно присваивать и изменять.

Пример:

```
program MyProg4;
var
  i, j : Integer;
  a : Vector;
  x, y : array[1..10] of real;
begin
end.
```

Подпрограммы - процедуры

```
procedure Имя(формальные_параметры);
  локальные_данные_процедуры;
begin
  операторы;
end;
```

Подпрограмма – именованный относительно независимый блок программы, который можно вызывать и которому можно передавать параметры.

Процедура – подпрограмма, вызываемая оператором вызова и не возвращающая в точку вызова какого-либо значения.

Пример:

```
program MyProg5;
procedure Print(i : integer);
begin
  writeln('Number is = ', i);
end;
begin
  Print(1);
  Print(100);
  Print(-125)
end.
```

Подпрограммы - функции

```
function ИмяФункции(формальные_параметры):Тип;
  локальные_данные_функции;
begin
  операторы; {в т.ч. ИмяФункции := возвр.значение}
end;
```

Функция – подпрограмма, возвращающая в точку вызова некоторое скалярное значение.

Пример:

```
program MyProg6;
function Sum(a, b : integer):integer;
begin
  Sum := a + b;
end;
begin
  writeln(Sum(1, 2));
  writeln(Sum(10, Sum(20, 30)));
end.
```

Операторы

Операторы

Оператор – простейшая инструкция, из которых состоит программа.

Операторы бывают:

1. Операторы действия (:=, read, write)
2. Операторы управления (if, case, goto, begin-end, for, while, repeat, with, вызов)

1. Оператор присваивания:

```
Переменная := выражение;
```

Особенности:

- Тип переменной должен *соответствовать* типу выражения !
- Порядок исполнения:
  1. Вычисление выражения
  2. Приведение типа (если требуется)
  3. Присвоение значения переменной

Оператор присваивания:

Соответствие типа:

- **точное совпадение** (напр.: integer:=integer)
- **родственные** типы (напр.: integer:=byte) – от меньшего к большему
- неявно **приводимые** типы (напр.: real:=integer)

Оператор присваивания:

- Пример. Проверьте корректность присваивания:

```
program MyProg7;
var
  i: integer;
  b: byte;
  r : real;
  d : double;
begin
  i := 100;
  b := 300;
  i := r;
  r := a;
  i := b;
  b := i;
  b := (byte)i;
  i := 8 / 2;
  b := 8 div 2;
end.
```

2. Оператор ввода:

```
read(список_переменных);
readln(список_переменных);
```

Особенности:

- Оператор ожидает ввода с консоли (клавиатуры) ввода значений переменных
- Типы значений должны соответствовать типам переменных (иначе – ошибка времени исполнения)

3. Оператор вывода:

```
write(список_выражений);
writeln(список_выражений);
```

Особенности:

- Оператор выводит на консоль (дисплей) значения **выражений скалярного типа**
- writeln еще переводит на новую строку
- **вывод по формату**:

  выражение : n

  вещест_выражение : n : m

n- общее количество знакомест; m – количество десятичных знаков (после запятой)

- Пример.

```
program MyProg8;
var
  a, b : integer;
  x : real;
begin
  writeln('Введите два целых числа:');
  readln(a, b);
  if b <> 0 then
  begin
    x := a / b;
    write('Отношение ', a:4, ' к ', b:4, 'равно':10);
    writeln(x:10:2);
  end
  else
    writeln(a:4, ' на ', b:4, 'не делится!');
end.
```

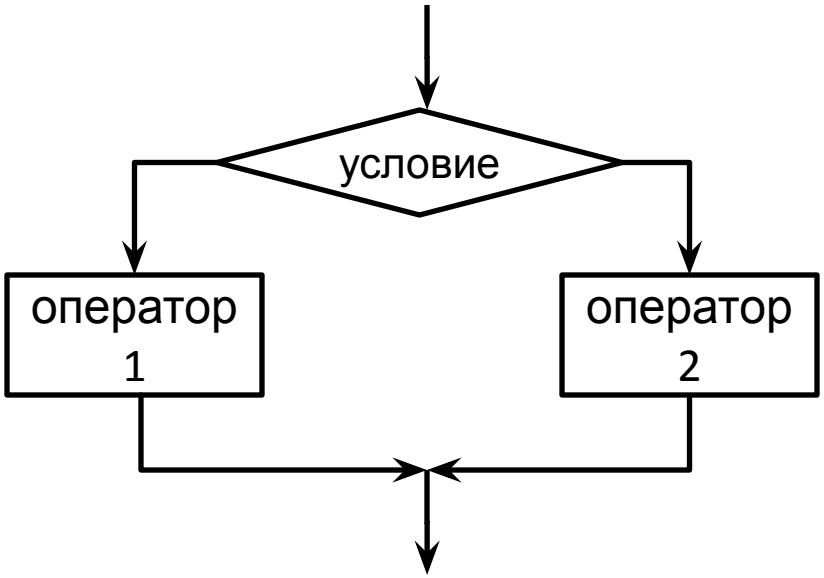
4. : Оператор составной (операторные скобки)

```
begin
   оператор1;
   оператор2;
   …
end;
```

Для создания из последовательности нескольких операторов одного – составного оператора (блока)

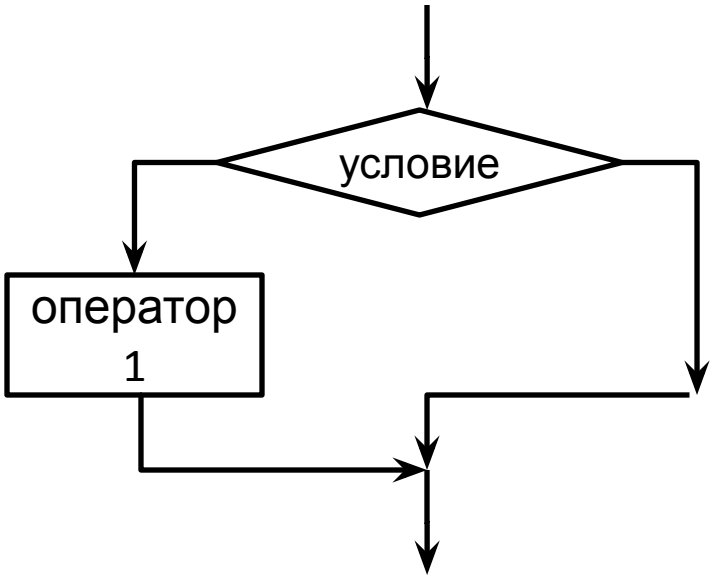
5. : Оператор условный

```
if условие then
  оператор1
[else
  оператор2];
```

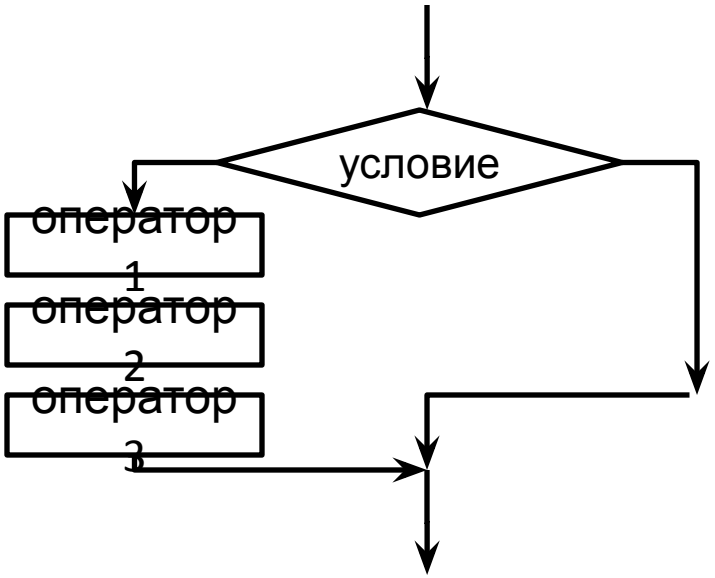


- **Исполняется оператор1, если условие истинно, в противном случае – оператор2**
- **Часть else может отсутствовать**
- **Условие – это выражение логического типа**
- **Оператор может быть составным begin..end**

```
if условие then
   оператор1;
```




```
if условие then
begin
   оператор1;
   оператор2;
   оператор3;
end;
```



Пример.

```
if x > 0 then
   if y > 0 then
      writeln('Первая четверть')
   else
      writeln('Четвертая четверть')
else
   if y > 0 then
      writeln('Вторая четверть')
   else
      writeln('Третья четверть');
```

6. : Оператор выбора

```
case выражение of
   зн1, зн2,… : оператор1;
   зн5, зн6,… : оператор2;
   …
   [else операторN;]
end;
```

- **Проверяются последовательно значения, при совпадении выполняется соответствующий оператор**
- **Часть else может отсутствовать**
- **выражение и значения должны быть одного (порядкового) типа**
- **значения должны быть все разными!**

Пример.

```
writeln('Введите номер месяца');
readln(m);
case m of
   12, 1, 2   : writeln(' Зима!');
   3, 4, 5    : writeln('Весна!');
   6, 7, 8    : writeln(' Лето!');
   9, 10, 11  : writeln('Осень!');
   else
      writeln('Такого не бывает!');
end;
```

7. : Оператор цикла с параметром

```
for параметр:=нз to кз do
  оператор; {тело цикла}
```

- параметр – переменная порядкового типа
- нз и кз вычисляются ДО начала исполнения цикла
- в теле цикла параметр изменять нельзя!
- вместо to может быть downto, тогда параметр уменьшается

Пример1.

```
readln(n);
s := 0;
f := 1;
for i:=1 to n do
begin
   p := p * n;
   s := s + p
end;
writeln(s);
```

Пример2.

```
n := 3;
s := 0;
for i:=1 to n do
begin
   s := s + i;
   if n mod 2 = 0 then
      n := n + 1;
end;
writeln(s);
```

Что выдаст программа?

Пример3.

```
s := 0;
for i:=1 to 3 do
   for j := 1 to i do
      s := s + i * j
writeln(s);
```

Что выдаст программа?

8. : Оператор цикла с предусловием

```
while условие do
  оператор; {тело цикла}
```

- **Пока условие *истинно*, исполняется тело цикла**
- **Если условие всегда истинно, то зацикливается!**
- **Тело может ни разу не быть исполнено**

Пример4.

{Найти все натуральные степени числа a, меньшие в}

```
readln(a, b);
p := a;
while p < b do
begin
   writeln(p);
   p := p * a;
end;
```

Пример5.
{Найти сумму цифр целого числа N}

```
readln(N);
s := 0;
while N <> 0 do
begin
   s := s + N mod 10;
   N := N div 10;
end;
```

Пример6.
{Найти максимальную цифру целого числа N}

```
readln(N);
max := 0;
while N <> 0 do
begin
   d := N mod 10;
   if d > max then
      max := d;
   N := N div 10;
end;
```

Пример7.

{Разложить натуральное N>2 на произведение простых чисел}

```
readln(N);
p:=2;
while N <> 1 do
begin
   while N mod p = 0 do
   begin
      write(p, '*');
      N := N div p;
   end;
   p := p + 1;
end;
```

9. : Оператор цикла с постусловием

```
repeat
   оператор1;   { тело цикла}
   оператор2;
   …
until условие;
```

- **Пока условие *ложно*, исполняется тело цикла**
- **Если условие всегда ложно, то зацикливается!**
- **Тело по крайней мере 1 раз будет исполнено**

Пример8.

{Дано натуральное число N. Найти наименьший полный квадрат, больший N}

```
readln(N);

p:=1;
repeat
   p := p + 1
until sqr(p) > N;

writeln(p);
```

10. : Оператор вызова

```
ИмяПроцедуры(ФактическиеПараметры);
```

Пример9.

```
uses Crt;
begin
   ClrScr;
   GotoXY(20, 20);
   write('Hello');
   GotoXY(40, 50);
   write('World!');
end.
```

11. : Оператор присоединения

```
with запись do
   оператор;
```

Используется для присоединения имени поля к имени записи

12. : Оператор безусловного перехода

goto метка;

Пример:

```
program MyProg1;
label
  lab1;
begin
  if 1 <= 2 then
    goto lab1;
  writeln('Hello, World!');
  lab1: writlen ('Hello, Boris!');
end.
```

НЕ ИСПОЛЬЗОВАТЬ