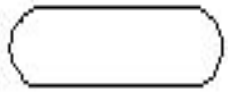


# Операторы языка Pascal

1. Схемы алгоритмов
2. Подпрограммы ввода-вывода
3. Составной оператор
4. Условный оператор
5. Оператор выбора **case**
6. Операторы цикла
7. Оператор цикла **for**
8. Оператор цикла **while**
9. Оператор цикла **repeat..until**
10. Операторы передачи управления

# 1. Схемы алгоритмов



Терминатор (вход, выход)



Ввод, вывод данных



Операция



Граница цикла



Решение



Предопределенный процесс

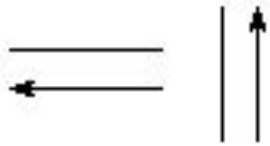


Комментарий



Соединитель

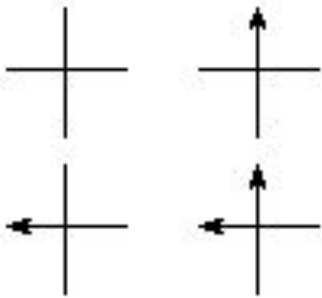
# 1. Схемы алгоритмов



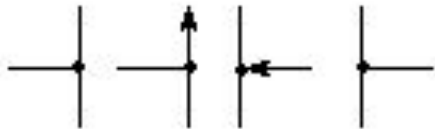
Линии потока



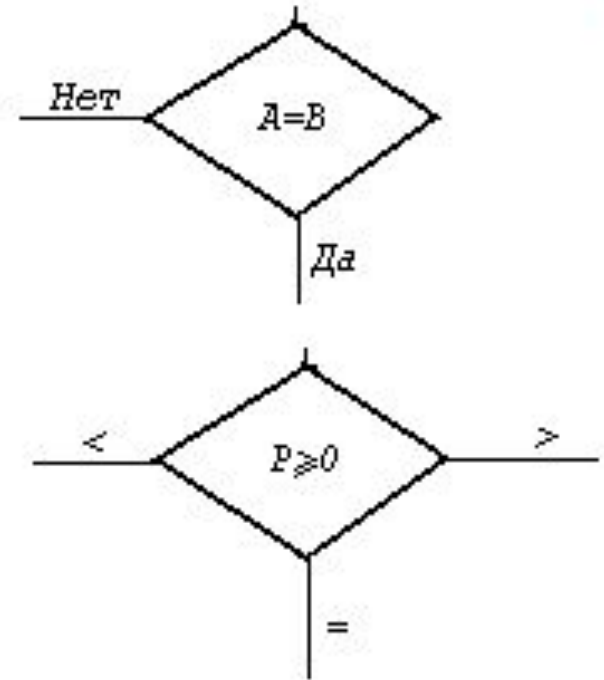
Излом линии



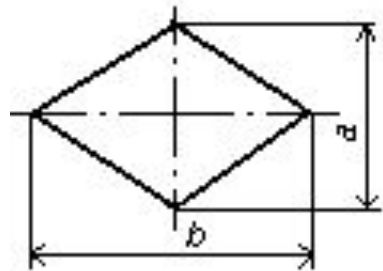
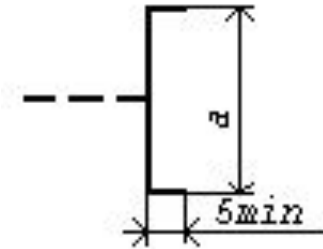
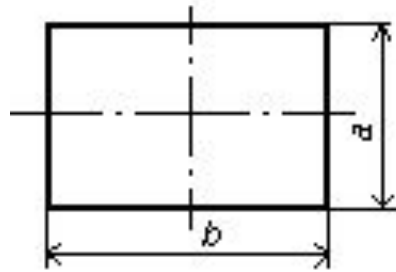
Пересечение линий потока



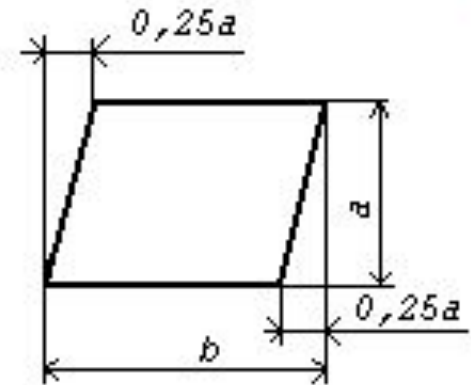
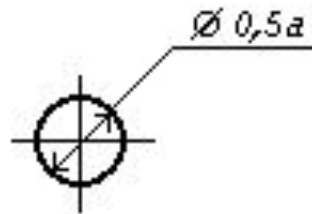
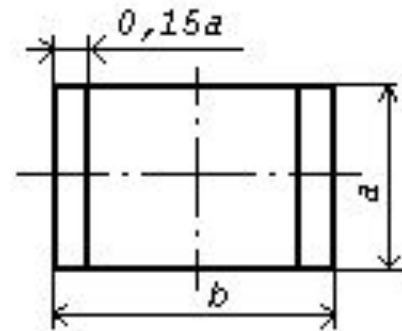
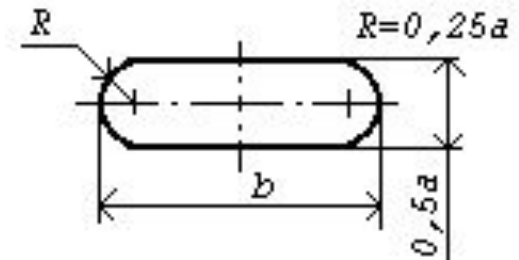
Слияние линий потока, место слияний потока обозначено точкой



# 1. Схемы алгоритмов

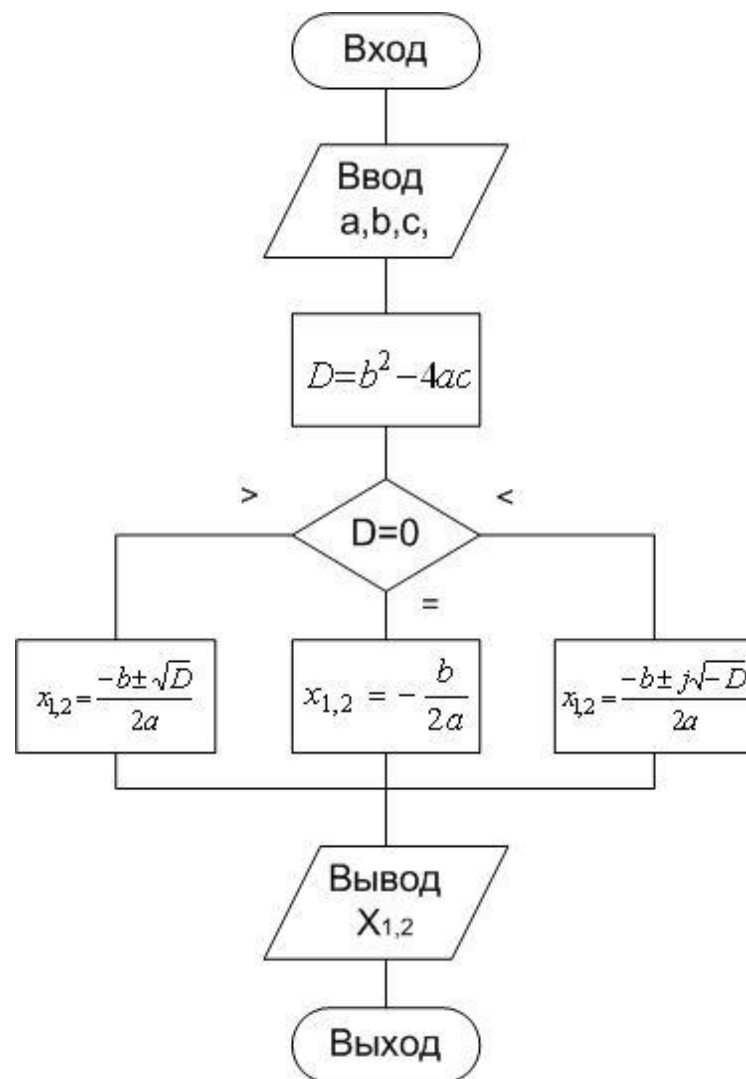
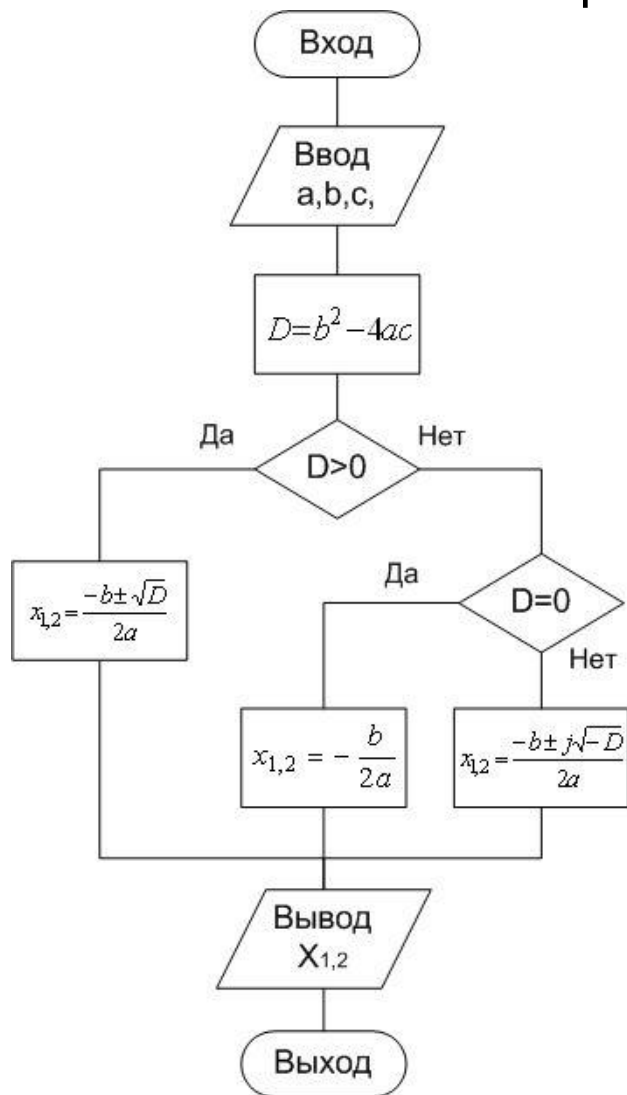


$a = 10, 15, 20 \text{ мм.}$   
 $b = 1.5a$



# 1. Схемы алгоритмов

## Схема алгоритма решения квадратного уравнения



# Операторы языка Pascal

1. Схемы алгоритмов
2. **Подпрограммы ввода-вывода**
3. Составной оператор
4. Условный оператор
5. Оператор выбора **case**
6. Операторы цикла
7. Оператор цикла **for**
8. Оператор цикла **while**
9. Оператор цикла **repeat..until**
10. Операторы передачи управления

## 2. Подпрограммы ввода-вывода

Ввод данных с клавиатуры:

```
Read(<Список ввода>);
```

```
Readln(<Список ввода>);
```

```
Readln;
```

Вывод данных на экран:

```
Write(<Список вывода>);
```

```
Writeln(<Список вывода>);
```

```
Writeln;
```

## 2. Подпрограммы ввода-вывода

**var**

a: integer;

b: real;

**begin**

readln(a,b);

writeln(a);

writeln(b);

**end.**

Вариант ввода 1:

1 10.12

1

10.12

Вариант ввода 2:

11

12.1

11

12.1



## 2. Подпрограммы ввода-вывода

### Форматированный вывод

Символ «:» задает формат вывода числа

```
Write(x:5); // под вывод числа отводится 5 символов,  
           // не хватает - расширяется
```

```
Write(x:5:2); // под число – 5 символов,  
            // под дробную часть – 2 символа
```

## 2. Подпрограммы ввода-вывода

```
var x: real;  
    y : integer;  
begin  
x:=12345.1537; y := 10;  
write(y); write(x); writeln;  
writeln(y,x); writeln(y,'-',x);  
writeln('Переменная y=',y,', переменная x=',x);  
writeln(y,x:12);  
writeln(x:2:3); writeln(x:3:4);  
writeln(x:15:2);  
end.
```

### Результат:

1012345.1537

1012345.1537

10-12345.1537

Переменная y=10, переменная x=12345.1537

10 12345.1537

12345.154

12345.1537

12345.15

# Операторы языка Pascal

1. Схемы алгоритмов
2. Подпрограммы ввода-вывода
3. **Составной оператор**
4. Условный оператор
5. Оператор выбора **case**
6. Операторы цикла
7. Оператор цикла **for**
8. Оператор цикла **while**
9. Оператор цикла **repeat..until**
10. Операторы передачи управления

# 3. Составной оператор

**Составной оператор** – это последовательность произвольных операторов программы, заключенная в операторные скобки – зарезервированные слова **BEGIN ... END**.

Составные операторы обрабатываются, как один оператор.

Это имеет решающее значение там, где синтаксис Паскаля допускает использование только одного оператора.

**begin**

.....

**begin**

    Z := X;

    X := Y;

**end;**

**end;**

# Операторы языка Pascal

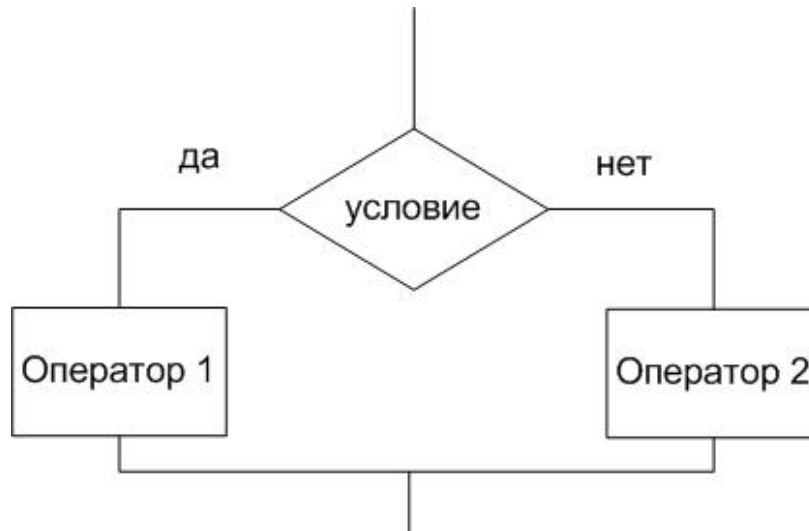
1. Схемы алгоритмов
2. Подпрограммы ввода-вывода
3. Составной оператор
4. **Условный оператор**
5. Оператор выбора **case**
6. Операторы цикла
7. Оператор цикла **for**
8. Оператор цикла **while**
9. Оператор цикла **repeat..until**
10. Операторы передачи управления

## 4. Условный оператор

Структура *полной* формы условного оператора:

**IF <условие> THEN <оператор1> ELSE <оператор2>**

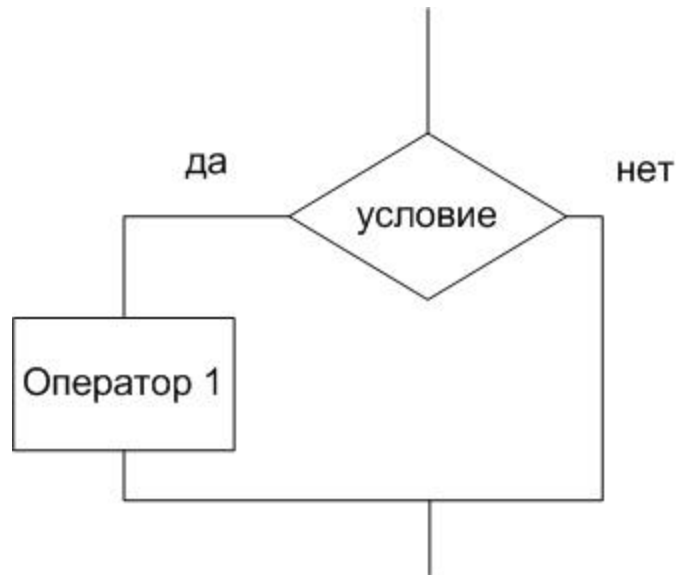
где IF, THEN, ELSE – зарезервированные слова (если, то, иначе);  
<условие> – произвольное выражение логического типа;  
<оператор1>, <оператор2> – любые операторы



## 4. Условный оператор

Структура *краткой* формы условного оператора:

**IF <условие> THEN <оператор1>**



## 4. Условный оператор

```
var
  x, y, max: real;
begin
  ... {ввод значений x, y, max}
  if x>max then //Полная форма условного оператора
    y := max
  else
    y := x;
  ...
end.
```

```
var
  x, y, max: real;
begin
  ... {ввод значений x, y, max}
  y := x;
  if y>max then //Краткая форма условного оператора
    y := max;
  ...
end.
```



## 4. Условный оператор

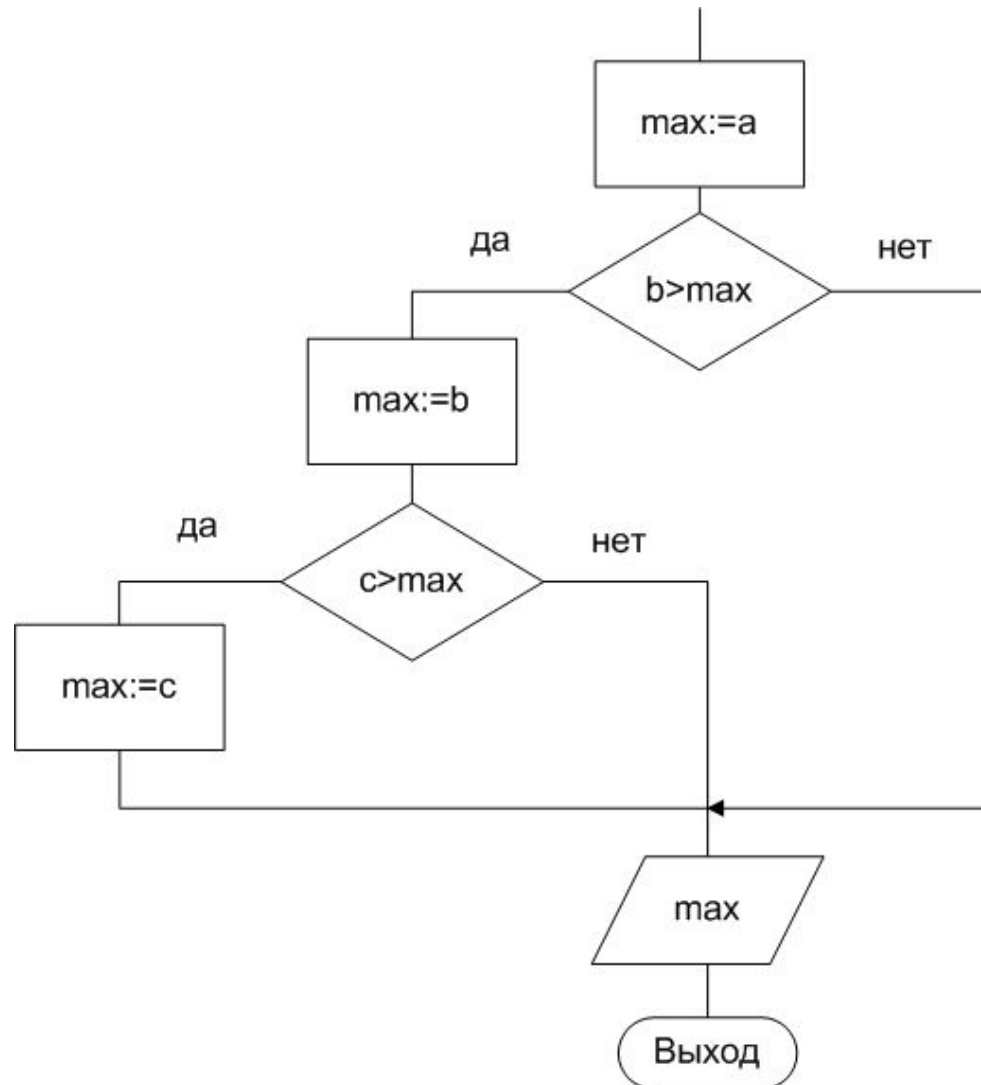
Пример. Отыскание наибольшего из трех чисел

```
var
  a, b, c, max: integer;
begin
  ...
  if a>b then
    if b>c then
      max := a
    else
      if a>c then
        max := a
      else
        max := c
    else {b>a}
      if b>c then
        max := b
      else
        max := c;
  writeln('Max = ', max);
end.
```

```
var
  a, b, c, max: integer;
begin
  ...
  max := a;
  if b>max then
    max := b;
  if c>max then
    max := c;
  writeln('Max = ', max);
end.
```

## 4. Условный оператор

Схема алгоритма к задаче поиска наибольшего из трех чисел



## 4. Условный оператор

**Пример: Вывести на экран сообщение попадает ли значение введенной переменной в отрезок [10; 20]**

```
var
  a:integer;
begin
  writeln('Введите значение переменной');
  readln(a);
  if a >= 10 and a<=20 then
    writeln('Значение "a" попадает в отрезок [10; 20]')
  else
    writeln('Значение "a" не попадает в отрезок [10; 20]');
end.
```

```
var
  a:integer;
begin
  writeln('Введите значение переменной');
  readln(a);
  if (a >= 10) and (a<=20) then
    writeln('Значение "a" попадает в отрезок [10; 20]')
  else
    writeln('Значение "a" не попадает в отрезок [10; 20]');
end.
```

**ОШИБКА:**  
Операция '>=' не применима к типам **integer** и **boolean**

**Верно**

# Операторы языка Pascal

1. Схемы алгоритмов
2. Подпрограммы ввода-вывода
3. Составной оператор
4. Условный оператор
5. **Оператор выбора case**
6. Операторы цикла
7. Оператор цикла **for**
8. Оператор цикла **while**
9. Оператор цикла **repeat..until**
10. Операторы передачи управления

# 5. Оператор выбора CASE

Структура:

```
case <ключ выбора> of  
    <список выбора 1>: <оператор1>;  
    ...  
    <список выбора N>: <операторN>;  
else <оператор0>  
end;
```

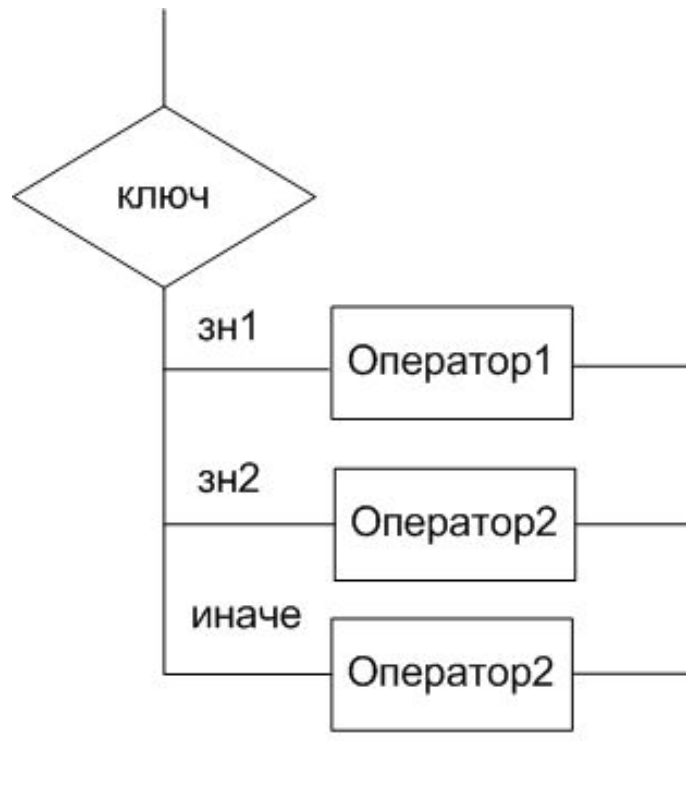
Здесь CASE, OF, ELSE, END – зарезервированные слова (случай, из, иначе, конец);

< список выбора>: <оператор>;(< список выбора > – константа того же типа, что и <ключ выбора>);

<оператор> – произвольный оператор Турбо Паскаля

# 5. Оператор выбора CASE

Схема алгоритма оператора выбора CASE



# 5. Оператор выбора CASE

```
var
  x,y,z: real;
  operation: char; {знак операции}
  stop: boolean; {признак ошибочной операции и останов}
begin
  stop := false;
  repeat
    writeln;      {пустая строка-разделитель}
    write('x, y ='); readln(x,y);
    write('операция:');
    readln(operation);
    case operation of
      '+': z := x+y;
      '-': z := x-y;
      '/': z := x/y;
      '*': z := x*y;
    else
      stop := true;
    end;
    if not stop then
      writeln('результат = ', z);
  until stop;
end.
```

# Операторы языка Pascal

1. Схемы алгоритмов
2. Подпрограммы ввода-вывода
3. Составной оператор
4. Условный оператор
5. Оператор выбора case
6. **Операторы цикла**
7. Оператор цикла for
8. Оператор цикла **while**
9. Оператор цикла **repeat..until**
10. Операторы передачи управления



## 6. Операторы цикла

**Цикл** – повторение одних и тех же действий

**Тело цикла** – последовательность действий, которая повторяется в цикле

**Шаг цикла** – один проход цикла

**Параметры цикла** – переменные, которые изменяются внутри цикла и влияют на его окончания

### **ВАЖНО:**

– цикл всегда должен закончиться, содержимое тела должно обязательно влиять на его окончание

– условие выхода из цикла должно состоять из корректных выражений и значений, определенных еще до первого выполнения тела цикла

# Операторы языка Pascal

1. Схемы алгоритмов
2. Подпрограммы ввода-вывода
3. Составной оператор
4. Условный оператор
5. Оператор выбора case
6. Операторы цикла
7. **Оператор цикла for**
8. Оператор цикла **while**
9. Оператор цикла **repeat..until**
10. Операторы передачи управления

# 7. Оператор цикла for

## Структура

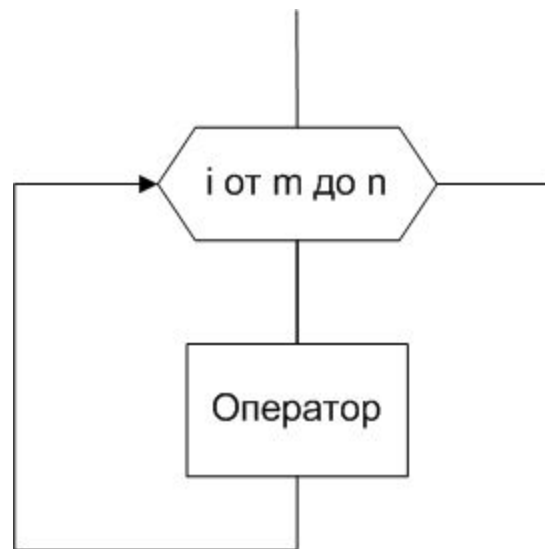
**FOR <парам. цикла> := <нач. знач.> TO <кон. знач.> DO <оператор>;**  
**FOR <парам. цикла> := <кон. знач.> DOWNTO <нач. знач.> DO <оператор>;**

Здесь FOR, TO, DO – зарезервированные слова (для, до, выполнить)

<парам. цикла> – переменная перечисляемого типа;

<нач. знач.> и <кон. знач.> – выражения переменная перечисляемого типа;

<оператор> – произвольный оператор Паскаля



## 7. Оператор цикла for

Пример. Расчёт факториала числа N

```
var
  i, N, fact: integer;
begin
  write('Введите N:');
  readln(N);
  fact := 1;
  for i := 1 to N do
    fact := fact*i;
  writeln('N!=', fact);
end.
```

```
var
  i, N, fact: integer;
begin
  write('Введите N:');
  readln(N);
  fact := 1;
  for i := N downto 1 do
    fact := fact*i;
  writeln('N!=', fact);
end.
```

# 7. Оператор цикла for

```
var
  i: integer;
begin
  writeln('начало цикла');
  for i := 1 to 3 do writeln('счетчик ',i);
  writeln('цикл завершился');
end.
```

начало цикла  
счетчик 1  
счетчик 2  
счетчик 3  
цикл завершился

```
var
  i: integer;
begin
  writeln('начало цикла');
  for i := 10 downto 8 do writeln('счетчик ',i);
  writeln('цикл завершился');
end.
```

начало цикла  
счетчик 10  
счетчик 9  
счетчик 8  
цикл завершился

# 7. Оператор цикла for

```
var
  i: integer;
begin
  writeln('начало цикла');
  for i := 5 to 2 do writeln('счетчик ',i);
  writeln('цикл завершился');
end.
```

начало цикла  
цикл завершился

```
var
  i: integer;
begin
  writeln('начало цикла');
  for i := 10 downto 11 do writeln('счетчик ',i);
  writeln('цикл завершился');
end.
```

начало цикла  
цикл завершился

## 7. Оператор цикла for

```
var
  i,x: integer;
begin
  writeln('начало цикла');
  x := 1;
  for i := 1 to 5 do
    begin
      x := x + 2;
      writeln('счетчик ',i, ' x=',x);
    end;
  writeln('цикл завершился');
end.
```

начало цикла  
счетчик 1 x=3  
счетчик 2 x=5  
счетчик 3 x=7  
счетчик 4 x=9  
счетчик 5 x=11  
цикл завершился

# 7. Оператор цикла for

```
var  
  i: integer;  
begin  
  writeln('начало цикла');  
  for i := 1 to 5 do ;  
    writeln('счетчик ',i);  
  writeln('цикл завершился');  
end.
```

начало цикла  
счетчик 5  
цикл завершился



# Операторы языка Pascal

1. Схемы алгоритмов
2. Подпрограммы ввода-вывода
3. Составной оператор
4. Условный оператор
5. Оператор выбора case
6. Операторы цикла
7. Оператор цикла for
8. **Оператор цикла while**
9. Оператор цикла **repeat..until**
10. Операторы передачи управления

# 8. Оператор цикла while

Структура

**WHILE <условие> DO <оператор>**

Здесь WHILE, DO – зарезервированные слова (пока [выполняется условие], делать)

<условие> – произвольное выражение логического типа;

<оператор> – произвольный оператор.



## 8. Оператор цикла while

```
var
  run: boolean;
  x : integer;
begin
  run := true;
  x := 0;
  while run do
    begin
      writeln('в цикле x=',x);
      x := x + 1;
      if x>=3 then run := false;
    end;
  writeln('после цикла x=',x);
end.
```

в цикле x=0  
в цикле x=1  
в цикле x=2  
после цикла x=3

## 8. Оператор цикла while

Пример. Отделение корня на отрезке методом половинного деления.

```
program mdp;
function f(x: real): real;
begin
  f:=exp(x*ln(5))-6*x-3;
end;
var
  a, b, e, c, x: real;
begin
  a:=1; b:=2;
  write ('задайте точность e=');
  read(e);
  c:=(a+b)/2;
  while abs(b-a)>e do
  begin
    if f(a)*f(c)<0 then b:=c else a:=c;
    c:=(a+b)/2;
  end;
  x:=(a+b)/2;
  writeln ('x=',x:3:3,' f(x)=',f(x):4:4);
end.
```

# Операторы языка Pascal

1. Схемы алгоритмов
2. Подпрограммы ввода-вывода
3. Составной оператор
4. Условный оператор
5. Оператор выбора case
6. Операторы цикла
7. Оператор цикла for
8. Оператор цикла while
9. **Оператор цикла repeat..until**
10. Операторы передачи управления

# 9. Оператор цикла repeat..until

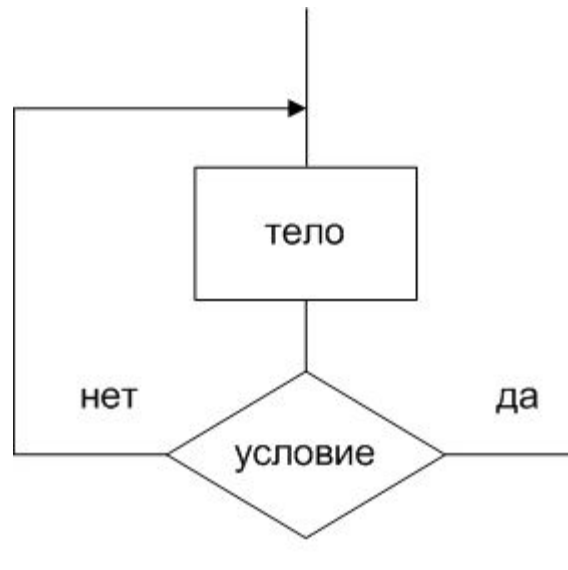
## Структура

**REPEAT <тело цикла> UNTIL <условие>**

Здесь REPEAT, UNTIL – зарезервированные слова (повторять до тех пор, пока не будет выполнено условие)

<тело цикла> – произвольная последовательность операторов.

<условие> – произвольное выражение логического типа



## 9. Оператор цикла repeat..until

Пример. Вывод ASCII-кода символа.

```
const
  ENTER = 13;
var
  ch: char;
begin
repeat
  readln(ch);
  writeln('Код символа "', ch, '" – ', ord(ch) );
until ord(ch) = ENTER
end.
```

```
1
Код символа '1' – 49

Код символа ' ' – 32
r
Код символа 'r' – 114
R
Код символа 'R' – 82

Код символа '
' – 13
```

## 9. Оператор цикла repeat..until

Рассчитать значение функции  $f(x)=2x+3$  на отрезке  $[a;b]$  с шагом  $h$

При помощи цикла *for*

**var**

i,N: integer;

a,b,h,x,y : real;

**begin**

writeln('введите параметры отрезка [a,b], h');

write('a=');readln(a);

write('b=');readln(b);

write('h=');readln(h);

x := a;

N := trunc((b-a)/h)+1;

**for i := 1 to N do**

**begin**

y := 2\*x + 3;

writeln('y(',x,')=',y);

x := x + h;

**end;**

**end.**



## 9. Оператор цикла repeat..until

Рассчитать значение функции  $f(x)=2x+3$  на отрезке  $[a;b]$  с шагом  $h$

При помощи цикла *while*

**var**

i,N: integer;

a,b,h,x,y : real;

**begin**

writeln('введите параметры отрезка [a,b], h');

write('a=');readln(a);

write('b=');readln(b);

write('h=');readln(h);

x := a;

**while not (x > b) do**

**begin**

y := 2\*x + 3;

writeln('y('x,')='y);

x := x + h;

**end;**

**end.**

## 9. Оператор цикла repeat..until

Рассчитать значение функции  $f(x)=2x+3$  на отрезке  $[a;b]$  с шагом  $h$

При помощи цикла *repeat..until*

**var**

i,N: integer;

a,b,h,x,y : real;

**begin**

writeln('введите параметры отрезка [a,b], h');

write('a=');readln(a);

write('b=');readln(b);

write('h=');readln(h);

x := a;

**repeat**

y := 2\*x + 3;

writeln('y(',x,')=',y);

x := x + h;

**until x > b**

**end.**

# 9. Оператор цикла *repeat..until*

## При помощи цикла *for*

введите параметры отрезка [a,b], h

a=-3

b=4

h=2

y(-3)=-3

y(-1)=1

y(1)=5

y(3)=9

## При помощи цикла *while*

введите параметры отрезка [a,b], h

a=-3

b=4

h=2

y(-3)=-3

y(-1)=1

y(1)=5

y(3)=9

## При помощи цикла *repeat..until*

введите параметры отрезка [a,b], h

a=-3

b=4

h=2

y(-3)=-3

y(-1)=1

y(1)=5

y(3)=9

# Операторы языка Pascal

1. Схемы алгоритмов
2. Подпрограммы ввода-вывода
3. Составной оператор
4. Условный оператор
5. Оператор выбора case
6. Операторы цикла
7. Оператор цикла for
8. Оператор цикла while
9. Оператор цикла repeat..until
10. **Операторы передачи управления**

# 10. Операторы передачи управления

Операторы передачи управления принудительно изменяют порядок выполнения команд:

- **BREAK** - осуществляет немедленный выход из циклов `repeat`, `while`, `for` и управление передается оператору, находящемуся непосредственно за циклом,
- **CONTINUE** – начинает новую итерацию цикла, даже если предыдущая не была завершена,
- **EXIT** – осуществляет выход из подпрограммы,
- **HALT** – прекращает выполнение программы.

# 10. Операторы передачи управления

## **begin**

```
writeln('строка 1');  
writeln('строка 2');  
halt;  
writeln('строка 3');  
writeln('строка 4');
```

**end.**

строка 1  
строка 2

```
var i : integer;
```

## **begin**

```
writeln('До цикла');
```

```
for i := 1 to 5 do
```

### **begin**

```
  writeln('итерация ',i);
```

```
  if i=2 then break;
```

```
end;
```

```
writeln('После цикла');
```

**end.**

До цикла  
итерация 1  
итерация 2  
После цикла

# 10. Операторы передачи управления

```
var i, j : integer;  
begin  
  writeln('До цикла');  
  for i := 1 to 3 do  
    for j := 1 to 3 do  
      begin  
        writeln('итерация i=',i,' j=',j);  
        if i=2 then break;  
      end;  
  writeln('После цикла');  
end.
```

До цикла  
итерация i=1 j=1  
итерация i=1 j=2  
итерация i=1 j=3  
итерация i=2 j=1  
итерация i=3 j=1  
итерация i=3 j=2  
итерация i=3 j=3  
После цикла