

Оператор ветвления.

Простые и составные условия

Принятие решений в программе. Простые условия

Как известно, операторы (команды) ввода и вывода, оператор (команда) присваивания позволяют составлять *линейные* программы, в которых все команды выполняются последовательно, одна за другой.

Напомню структуру программы на языке Паскаль:

```
var
    <раздел описания переменных>;
begin
    <раздел команд (тело программы)>;
end.
```

Примечания.

1. После каждой строки программы ставится точкой с запятой (;).
Исключения: после begin и var не ставится ничего, а в конце программы после end ставится точка.
2. При вводе текста программы с клавиатуры не имеет значения, какие буквы вы используете: строчные или прописные.

Но в жизни, решая те либо иные задачи, часто приходится принимать решения в зависимости от создавшейся ситуации либо от некоторого условия (условий). Например, если сделаны уроки, можно отдохнуть; если холодно – необходимо теплее одеться; если в театральной кассе имеются билеты, можно пойти в театр на спектакль, иначе можно просто погулять в парке. Принимая решения, человек рассуждает, анализируя ситуацию.

В программировании можно также создавать программы, умеющие **выполнять выбор**. Для этого существуют команды, которые позволяют компьютеру принимать решения в зависимости от выполнения некоторого условия. Одной из таких команд является условный оператор языка программирования Pascal

```
if <условия>
then begin
    <набор операторов 1>;
end
else begin
    <набор операторов 2>;
end;
```

В переводе на русский язык данная форма записи означает: **если выполняются условия, то исполняется набор операторов 1, иначе исполняется набор операторов 2.**

Рассмотрим на примере, как работает оператор if.

Например, необходимо дать задание компьютеру *проверить, знают ли*

ученики таблицу умножения.

Задача 1. Проверка знаний таблицы умножения

```
var
  a: integer;
begin
  writeln('Сколько будет 3х5?');
  readln(a);           {запрашивается ответ, вводимый с клавиатуры,
                        {который записывается в переменную, а}

  if a = 15
  then begin           {оператор if проверяет и анализирует значение переменной а.
                        {Если а=15, то компьютер выводит сообщение «Верно»,
                        {в противном случае сообщает «Неверно»}

    writeln('Верно');
  end
  else begin
    writeln('Неверно');
  end;
end.
```

```
var
  a: integer;
begin
  writeln('Сколько будет 3х5?');
  readln(a);
  if a = 15
  then begin
    writeln('Верно');
  end
  else begin
    writeln('Неверно');
  end;
end.
```

Результат выполнения программы в случае ввода числа 15:

```
Окно вывода
Сколько будет 3х5?
15
Верно
```

При ином ответе, например, 25, появится сообщение:

```
Окно вывода
Сколько будет 3х5?
25
Неверно
```

Как видно из результатов, эта программа умеет «рассуждать», сравнивая число, которое вы ввели с клавиатуры с правильным ответом. И делает это она с помощью оператора if.

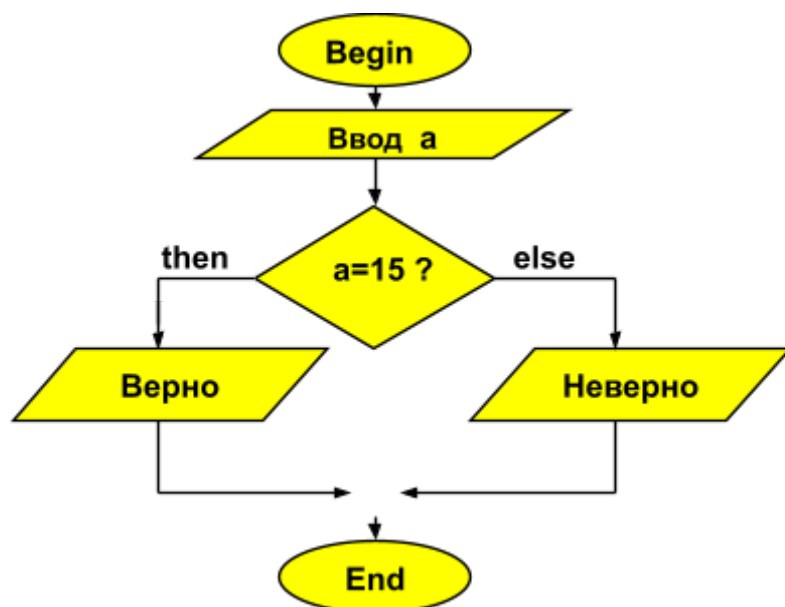
Оператор if анализирует некоторое условие, например, $a > b$ либо $a = 15$ либо $a > 0$. Действия, которые затем будут выполняться, зависят от того, выполняется либо не выполняется конкретное условие. Эти действия (операторы) называются *ветвями программы*.

Условие – это выражение, стоящее в операторе if..then сразу после слова if. В зависимости от этого условия (его истинности или ложности) компьютер выполняет ту либо иную ветвь программы.

Для обозначения оператора if на блок-схемах используются ромбы, называемые *блоками проверки условия*. Алгоритмическая конструкция ветвления представлена на рисунке.



А вот как будет выглядеть блок-схема алгоритма, реализованного в примере 1.



Простое условие - это математическое сравнение двух выражений по величине (сравнение двух величин).

Операции сравнения на языке программирования можно записать при помощи следующих знаков

Операции сравнения

Знак	Операция сравнения
=	Равно
<	Меньше
<=	Меньше либо равно
>	Больше
>=	Больше либо равно
<>	Не равно

Примеры простых условий:

$a > b$

$A \leq 0$

$A + 3 * c \geq 20$

В качестве оператора_1 и оператора_2 может быть любая из уже известных вам команд.

Рассмотрим еще одну задачу.

Задача 2. С клавиатуры вводятся 2 числа. Вывести на экран большее из них.

```
var
  a, b: integer;
begin
  writeln('Введите числа ');
  readln(a, b);
  if a > b
  then begin
    write(a);
  end
  else begin
    write(b);
  end;
end.
```

```
var
  a, b: integer;
begin
  writeln('Введите числа ');
  readln(a, b);
  if a > b
  then begin
    write(a);
  end
  else begin
    write(b);
  end;
end.
```

Окно вывода

Введите числа
25 89
89

Составные условия

При решении различных задач иногда возникает необходимость проверять выполнение двух и более условий. Такие условия называют *составными* (как например, $0 < a < 5$).

Для записи составных условий на языке программирования используют следующие *логические операции*:

- **and** – логическое «и»;
- **or** – логическое «или»;
- **not** – логическое отрицание.

Простые условия при этом обязательно заключаются в скобки, так как логические операции имеют более высокий приоритет, чем операции сравнения.

Примечания.

Правила выполнения логических операций

1. Составное условие, состоящее из двух простых условий, соединенных операцией **and**, верно (истинно) только тогда, когда верны оба простых условия.
2. Составное условие, состоящее из двух простых условий, соединенных

операцией **or**, верно тогда, когда верно хотя бы одно из простых условий.

3. Составное условие **not** верно только тогда, когда простое *условие* ложно.
4. Составное условие, состоящее из двух простых условий, соединенных операцией **xor**, верно тогда, когда верно только одно из условий.

Задача 3. Определите, принадлежит ли введенное с клавиатуры число промежутку [20;140).

```
var
  a: integer;

begin
  writeln('Введите число: ');
  readln(a);
  if (a >= 20) and (a < 140)
  then begin
    writeln('Принадлежит');
  end
  else begin
    writeln('Не принадлежит');
  end;
end.
```

```
var
  a: integer;
begin
  writeln('Введите число: ');
  readln(a);
  if (a >= 20) and (a < 140)
  then begin
    writeln('Принадлежит');
  end
  else begin
    writeln('Не принадлежит');
  end;
end.
```

Окно вывода

Введите число:
28
Принадлежит

Сокращенная форма условного оператора

Форма условного оператора

```
If <условия>
  then begin
    <набор операторов 1>;
  end
  else begin
    <набор операторов 2>;
  end;
```

называется **полной**.

В языке программирования Pascal существует также *сокращенная форма* условного оператора, которая применяется в тех случаях, когда какое-либо действие (группу действий) нужно выполнить только при выполнении заданного условия.

Сокращенная форма условного оператора имеет вид:

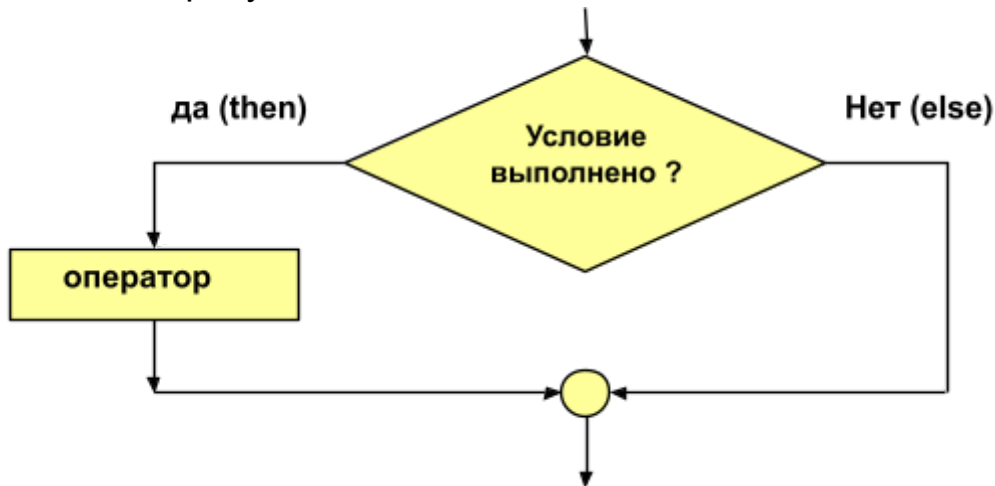
```
If <условия>
```

```

then begin
    <набор операторов 1>;
end;

```

Блок – схема сокращенной формы записи алгоритма с ветвлением представлена на рисунке.



Задача 4. С клавиатуры вводится число. Если оно четное, уменьшить его на 10.

```

var
  a: integer;
begin
  read(a);
  if a mod 2 = 0
  then begin
    a := a - 10;
  end;
  writeln(a);
end.

```

```

var
  a: integer;
begin
  read(a);
  if a mod 2 = 0
  then begin
    a := a - 10;
  end;
  writeln(a);
end.

```

Окно вывода

46
36

```

var
  a: integer;
begin
  read(a);
  if a mod 2 = 0
  then begin
    a := a - 10;
  end;
  writeln(a);
end.

```

Окно вывода

45
45

Составной оператор

При составлении программ на языке программирования часто бывает так, что в случае выполнения либо невыполнения некоторого условия в операторе `if` необходимо осуществить несколько действий. В этом случае последовательность действий (несколько операторов подряд) объединяют в одну группу, заключенную между словами `begin` и `end`.

Пример:

```

if x > 0
then begin
  x := x * 2;

```

```
write (x);  
end;
```

Такая группа называется **составным оператором** и рассматривается как единое целое.

Зарезервированные слова **Begin** и **End** часто называют открывающей и закрывающей **операторными скобками**.

Рассмотрим пример, демонстрирующий использование составного оператора.

Задача 5. С клавиатуры вводятся 2 числа. Если эти числа положительны, найти разность и частное этих чисел, в противном случае – сумму и произведение.

```
var  
  a, b, c, d: real;  
begin  
  writeln('Введите числа: ');  
  readln(a, b);  
  if (a > 0) and (b > 0)  
  then begin  
    c := a - b;  
    d := a / b;  
  end  
  else begin  
    c := a + b;  
    d := a * b;  
  end;  
  writeln(c);  
  writeln(d);  
end.
```

```
var  
  a, b, c, d: real;  
begin  
  writeln('Введите числа: ');  
  readln(a, b);  
  if (a > 0) and (b > 0)  
  then begin  
    c := a - b;  
    d := a / b;  
  end  
  else begin  
    c := a + b;  
    d := a * b;  
  end;  
  writeln(c);  
  writeln(d);  
end.
```

<

Окно вывода

Введите числа:
58 -25
33
-1450