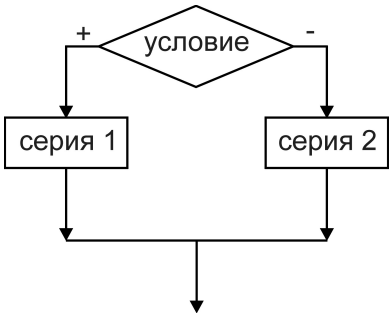
1.1 Оператор условного перехода

Условный оператор позволяет проверить некоторое условие (логическое выражение, boolean) и в зависимости от результатов проверки выполнить то или иное действие. Таким образом, условный оператор - это средство ветвления вычислительного процесса.

Для записи команды ветвления используется **полная команда ветвления команда *if...then...else;***.

если условие	*if* условие
то оператор1	*then* оператор1
иначе оператор2	*else* оператор 2;
все	



Где IF, THEN, ELSE - зарезервированные слова (если, то, иначе); <условие> - произвольное выражение логического типа; <оператор1>, <оператор2> - любые операторы языка Паскаль.

Условный оператор работает по следующему алгоритму. Вначале вычисляется условное выражение <условие>. Если результат есть TRUE (истина), то выполняется <оператор1>, а <оператор2> пропускается; если результат есть FALSE (ложь), наоборот, <оператор1> пропускается, а выполняется <оператор2>.

Пример: *if x>=0 then y:=2*x else y:=3 - x ;.*

Перед служебным словом *else* знак «;» не ставится.

Можно сказать, что слова *then* и *else* действуют только на один оператор, т. е. до первого знака «;». Чтобы действие этих слов распространялось на несколько операторов, используют операторные скобки или составной оператор:

begin ..... end (перед *end* знак «;» может не ставиться).

Пример:

```
 if x>0 then
                 begin y:=1; z:=7 end
        else     begin y:=2; z:=9 end;.
```

В этом случае команда ветвления заканчивается последним словом *end* и после этого слова ставится знак «;».

Пример:

1)
```
if x>0 then  y:=1
          else begin y:=2; z:=9 end;
```
2)
```
if x>0 then begin y:=1; z=7 end
          else y:=2;
```

В этих примерах только после одного служебного слова используются операторные скобки.

В записи *if x>0 then y:=1 else y:=2;z:=7* значение 7 присвоится переменной ***z*** и при *x*>0 и при *x*<=0, так как оператор *z*:=7; уже не относится к оператору ветвления.

После служебного слова *if* без скобок записывается только простое условие. При записи составных условий обязательно каждое простое условие заключается в круглые скобки и отделяется пробелом от служебных слов:

if (x>1) and (x<7) then ...

При записи команды ветвления возникает проблема «болтающегося» *else* при оценке нескольких условий. Поскольку любой из операторов <оператор1> и <оператор2> может быть любого типа, в том числе и условным, а в то же время не каждый из «вложенных» условных операторов может иметь часть ELSE <оператор2>, то возникает неоднозначность трактовки условий.

Если не ввести дополнительного соглашения, то в записи

if условие *then if* условие *then* оператор 1 *else* оператор 2;

не понятно, к какому условному оператору, внутреннему или внешнему, относится *else*.

Эта неоднозначность в Паскале решается следующимобразом: любая встретившаяся часть ELSE соответствует ближайшей к ней «сверху» части THEN условного оператора.

Но если в *then* входит оператор *if*, то его лучше заключить его в операторную скобку *begin...end*, даже если *if* единственный оператор в *then*-части.

```
if условие then
  begin
     if условие
        then оператор 1
        else оператор 3;
  end
     else оператор 2;
```

Сокращенная форма оператора ветвления имеет вид:
***if* условие *then* оператор;**

или

```
if условие                    если
  then                      то
     begin                  оператор1
        оператор 1;             оператор2
        оператор 2;             оператор3
        оператор 3;
     end;                   все
```

Пример 1. Решим уравнение вида $ax = в$.

Для решения этого, на первый взгляд, простого уравнения, необходимо рассмотреть следующие случаи:

1.Если a не равно нулю, то решение уравнения определяется через деление числа $в$ на число a.

2.Если a равно нулю и $в$ равно нулю, то уравнение принимает вид $0x = 0$, а это значит, что решением его может быть любое число.

3.Если a равно нулю, а $в$ не равно нулю, то уравнение $0x = в$ не имеет решения.

Вариант программы 1.

```
program urav;
        var a,b,x:real;
BEGIN
        writeln('введите значения a и  в');
        readln(a,b);
        if a<>0 then begin
                                x:=b/a;
                                writeln('Ответ ',x:5:2);
                        end
                else
                                if (b=0) then writeln('любое число')
                                        else writeln('нет решений');
END.
```

Вариант праграмы 2.

```
program urav;
        var a,b,x:real;
BEGIN
        writeln('введите значения а и в');
        readln(a,b);
        if a<>0 then begin
                                x:=b/a;
                                writeln('Ответ ',x:5:2);
                        end;
        if (a=0) and (b=0) then writeln('любое число');
        if (a=0) and (b<>0) then writeln('нет решений');
END.
```

1.2 Оператор выбора варианта

Оператор выбора варианта является одним из обобщений условного оператора. Он дает возможность выполнить один или несколько операторов в зависимости от значения варианта.

Его вид:

– полная форма:

case вариант ***of***
<список меток>:оператор 1;
<список меток>:оператор 2;
.....
<список меток>:оператор *n*
else оператор n+1;
***end*;**

– сокращенная форма:

case вариант ***of***
<список меток>:оператор 1;
<список меток>:оператор 2;
.....
<список меток>:оператор *n*;
***end*;**

case, *of*, *end* (выбор, из, конец) – служебные слова.

Вариант – выражение любого скалярного типа, кроме вещественного.

Оператор – любой оператор языка Паскаль.

Список меток – это список разделенных запятыми значений выражения или одно его значение. Эти константы должны иметь тот же тип, что и выражение, и называются метками выбора. Эта метка не обязательно целое число (может быть и символ), и она не описывается в разделе меток *label*, на нее нельзя ссылаться в операторе *goto*. Метка отмечает только один оператор, для отметки нескольких операторов используются операторные скобки begin.. end.

Оператор выбора исполняет тот оператор, одна из меток которого равна текущему значению условия. По окончании выполнения выборного оператора управление передается на конец команды case.

```
program abs
var
  x,g,y:real;
  nomer:integer;
BEGIN
```

```
  y:=0; x:=2.7; g:=-12.4; readln (nomer);;
  case nomer of
      2: y:=g;
      4: y:=g*x;
      6: y:=g*sqrt(abs(x));
      8: y:=g*sqr(sin(x)+12)
end;
  writeln ('y=',  y:7:2)
END.
```

Задача. Сколько дней в каждом месяце?

```
program bbb;
type
  mes=(январь, февраль, март, апрель, май, июнь, июль, август,
   сентябрь, октябрь, ноябрь, декабрь);
var  mm: mes;     g: 1900..2000;  p: 28..31;
BEGIN
  case mm of
      май, январь, март, июль, август, октябрь, декабрь: p:=31;
      апрель, июнь, сентябрь, ноябрь: p:=30;
      февраль: if (g mod 4 = 0) and(g mod 100<>0) or (g mod 400=0)
                then p:=29
                else p:=28;
  end;
  writeln('число дней ', p);
END.
```

1.3 Оператор безусловного перехода

Он имеет вид *goto* <метка>;

Метка – целое число без знака определения в разделе *label*. Оператор *goto* производит передачу управления к оператору, помеченному указанной меткой. Применение операторов безусловного перехода в языке Паскаль является необязательным и нежелательным, т. к. присутствие этого оператора в программе нарушает структурную целостность и наглядность. Такую программу становится трудно читать, отлаживать, модифицировать.

```
program rrr;
label 5;
var a,b:real;
BEGIN
  5: raedln(a);
  if a<0 then goto 5 ;
  b:=a*a;
  writeln(b)
END
```