



Программирование ветвлений

Разветвляющийся алгоритм

Разветвляющимся называется такой алгоритм, в котором выбирается один из нескольких возможных вариантов вычислительного процесса. Каждый подобный путь называется ветвью алгоритма.

Признаком разветвляющегося алгоритма является наличие операций проверки условия.

Различают два вида условий - простые и составные.

Простые условия

Простым условием (отношением) называется выражение, составленное из двух арифметических выражений или двух текстовых величин (иначе их еще называют операндами), связанных одним из знаков:

$<$ - меньше, чем...

$>$ - больше, чем...

$<=$ - меньше, чем... или равно

$>=$ - больше, чем... или равно

$<>$ - не равно

$=$ - равно

Простые условия

Выражения, при подстановке в которые некоторых значений переменных, о нем можно сказать истинно (верно) оно или ложно (неверно) называются булевыми (логическими) выражениями.

Примечание. Название «булевы» произошло от имени математика Джорджа Буля, разработавшего в XIX веке булеву логику и алгебру логики.

Переменная, которая может принимать одно из двух значений: True (правда) или False (ложь), называется булевой (логической) переменной.

Пример

Например, простыми отношениями являются следующие:

- $x - y > 10$;
- $k \leq \text{sqr}(c) + \text{abs}(a + b)$;
- $9 < > 11$;
- 'мама' < > 'папа'.

В приведенных примерах первые два отношения включают в себя переменные, поэтому о верности этих отношений можно судить только при подстановке некоторых значений.

Составные условия

Логические операции, операции отношения и арифметические операции часто встречаются в одном выражении. При этом отношения, стоящие слева и справа от знака логической операции, должны быть заключены в скобки, поскольку логические операции имеют более высокий приоритет.

Составные условия

Принят следующий приоритет операций:

- **not**
- **and, *, /, div, mod**
- **or, +, -**
- **операции отношения**

Логическую операцию **and** еще называют логическим умножением, а логическую операцию **or** - логическим сложением.

Задание

- 1) Вычислите значения выражения:
 - а) $\text{sqr}(x) + \text{sqr}(y) \leq 4$ при $x=0.3$, $y=-1.6$;
 - б) $k \bmod 7 = k \text{ div } 5 - 1$ при $k=15$;
 - в) $t \text{ and } (p \bmod 3 = 0)$ при $t=\text{true}$, $p=101010$;
 - г) $(x * y \neq 0) \text{ and } (y > x)$ при $x=2$, $y=1$;
 - д) $(x * y \neq 0) \text{ or } (y > x)$ при $x=2$, $y=1$;
 - е) $a \text{ or } (\text{not } b)$ при $a=\text{False}$, $b=\text{True}$;

Задание

- 2) Записать на Паскале отношение, истинное при выполнении указанного условия и ложное в противном случае:
- а) целое k делится на 7;
 - б) точка (x, y) лежит вне круга радиуса R с центром в точке $(1,0)$;
 - в) натуральное число N является квадратом натурального числа;
 - г) $0 < x < 1$;
 - д) хотя бы одна из логических переменных a и b имеет значение True;
 - е) обе логические переменные a и b имеют значение True.

Задание

3) Записать на Паскале выражение, истинное при выполнении указанного условия и ложное в противном случае:

- а) x принадлежит отрезку $[0, 1]$;
- б) x лежит вне отрезка $[0, 1]$;
- в) x принадлежит отрезку $[2, 5]$ или $[-1, 1]$;
- г) x лежит вне отрезков $[2, 5]$ и $[-1, 1]$;
- д) каждое из чисел x, y, z положительно;
- ж) ни одно из чисел x, y, z не является положительным.

Полная форма конструкции условного оператора

```
if <логическое выражение>  
  then  
    <оператор 1>  
  else  
    <оператор 2>
```

Условный оператор работает по следующему алгоритму

Сначала вычисляется значение логического выражения, расположенного за служебным словом IF. Если его результат **ИСТИНА**, выполняется <оператор 1>, расположенный после слова THEN, а действия после ELSE пропускаются; если результат **ЛОЖЬ**, то, наоборот, действия после слова THEN пропускаются, а после ELSE выполняется <оператор 2>.

Составной оператор

Если в качестве оператора должна выполняться серия операторов, то они заключаются в операторные скобки `begin-end`.

Конструкция `Begin ... End` называется составным оператором.

Полная форма конструкции условного оператора

```
if <логическое выражение>  
  then  
    begin  
      оператор 1;  
      оператор 2;  
      ...  
    end  
  else  
    begin  
      оператор 1;  
      оператор 2;  
      ...  
    end;  
end;
```

Общие правила употребления точки с запятой

- Каждое описание переменной и определение константы заканчиваются точкой с запятой.
- Каждый оператор в теле программы завершается точкой с запятой, если сразу за ним не следуют зарезервированные слова `End`, `Else`, `Until`.
- После определенных зарезервированных слов, таких, как `Then`, `Else`, `Var`, `Const`, `Begin`, никогда не ставится точка с запятой.

Пример

Вывести на экран большее из двух данных чисел.

Program Example1;

Var x, y: integer; {вводимые числа}

Begin

write('Введите 2 числа '); {вводим два целых числа через пробел}

readln(x,y);

if x > y

then

writeln ('Большее число ', x) {если x больше y, то выводим x}

else

writeln ('Большее число ', y) {иначе выводим y}

End.

Сокращенную (неполную) форму записи условного оператора

```
if <логическое выражение>  
  then  
    <оператор>
```

Тогда если выражение, расположенное за служебным словом IF, в результате дает истину, выполняются действия после слова THEN, в противном случае эти действия пропускаются.

Пример

Составить программу, которая, если введенное число отрицательное, меняет его на противоположное.

```
Program Chisla;
```

```
  Var x: integer; {вводимое число}
```

```
  Begin
```

```
    write('Введите число '); {вводим целое число}
```

```
    readln(x);
```

```
    if x < 0
```

```
      then
```

```
        x := -x;
```

```
    writeln ('x= ', x)
```

```
  End.
```