**Операторы цикла**

С.А. Григорьев

Для реализации циклических алгоритмов, т.е. алгоритмов, содержащих многократно повторяющиеся одинаковые операции, применяются специальные операторы цикла. В Паскале есть три вида циклов: FOR, WHILE и REPEAT. Оператор цикла FOR записывается в виде:

FOR переменная:=начальное значение TO конечное значение DO

оператор/блок

или

FOR переменная:=начальное значение DOWNTO конечное значение DO

оператор/блок.

Здесь переменная - любая переменная порядкового типа, называемая в таком контексте переменной цикла, начальное значение и конечное значение - выражения того же типа (исключение, как всегда делается для разнотипных целочисленных переменных). Цикл FOR выполняется следующим образом: переменной цикла присваивается начальное значение, после чего выполняется тело цикла (оператор или блок, стоящий после DO). Два этих действия вместе составляют один шаг цикла. Затем переменной цикла присваивается следующее (в цикле FOR ... TO) или предыдущее (в цикле FOR ... DOWNTO) значение (вспомним функции Succ и Pred) и выполняется следующий шаг цикла. Так происходит до тех пор, пока значение переменной цикла не станет больше (FOR...TO) или меньше (FOR...DOWNTO) конечного значения. Цикл FOR может не выполниться ни разу, если начальное значение больше конечного в цикле FOR...TO или меньше конечного в цикле FOR...DOWNTO. Запишем два примера использования цикла FOR : вычислим сумму квадратов натуральных чисел от 1 до N.

 VAR i : Word;

 CONST s : Real = 0; N = 22;

 BEGIN FOR i:=1 TO N DO s:=s+SQR(i); WRITELN('сумма=',s); END.

и выведем на экран символы с номерами от 32 до 255

 VAR c : Char;

 BEGIN FOR c:=' ' TO #255 DO WRITE(c); WRITELN; END.

Второй тип цикла - цикл WHILE - записывается в виде:

WHILE логическое выражение DO оператор/блок

Здесь логическое выражение - любое выражение типа Boolean. Цикл выполняется следующим образом : вычисляется логическое выражение и, если оно истинно, выполняется тело цикла, в противном случае цикл заканчивается. Очевидно, что цикл WHILE может как не выполниться ни разу, так и выполняться бесконечное количество раз (в последнем случае говорят, что программа зациклилась). Запишем две предыдущие задачи, используя цикл WHILE :

 CONST i : Word = 1; s : Real = 0; N = 22;

 BEGIN WHILE i<=N DO BEGIN s:=s+SQR(i); INC(i); END;

 WRITELN('сумма=',s);

 END.

 VAR c : Char;

 BEGIN c:=Pred(' ');

 WHILE c<#255 DO BEGIN c:=Succ(c); WRITE(c); END;

 WRITELN;

 END.

В качестве упражнения, подумайте, почему программа

 VAR c : Char;

 BEGIN c:=' ';

 WHILE c<=#255 DO BEGIN WRITE(c); c:=Succ(c); END;

 WRITELN;

 END.

оказывается зацикленной.

Третий тип цикла - REPEAT - записывается в виде:

REPEAT операторы UNTIL логическое выражение;

Если тело цикла REPEAT содержит больше одного оператора, нет необходимости использовать блок, поскольку сами ключевые слова REPEAT и UNTIL являются в данном случае логическими скобками. Перед UNTIL можно не ставить ";". Цикл REPEAT выполняется так : сначала выполняется тело цикла, затем вычисляется логическое выражение, и если оно истинно, цикл заканчивается. Таким образом, цикл REPEAT всегда выполняется хотя бы один раз и так же, как и WHILE, подвержен зацикливанию. Запишем наши примеры циклом REPEAT :

 CONST i : Word = 1; Real = 0; N = 22;

 BEGIN REPEAT s:=s+SQR(i); INC(i) UNTIL i>N;

 WRITELN('сумма=',s);

 END.

 VAR c : Char;

 BEGIN c:=Pred(' ');

 REPEAT c:=Succ(c); WRITE(c) UNTIL c=#255;

 WRITELN;

 END.

Из приведенных примеров очевидно, что любой циклический алгоритм можно записать любым видом цикла, все они взаимозаменяемы и выбираются программистом в соответствии с его вкусами, однако можно порекомендовать в тех случаях, когда количество шагов цикла известно заранее, использовать цикл FOR.

В последней версии языка Паскаль появились процедуры BREAK и CONTINUE, аналогичные операторам break и continue языка С. Процедура BREAK приводит к немедленному окончанию цикла, в котором она вызвана. Вызов процедуры CONTINUE приводит к немедленному переходу к следующему шагу цикла. Запишем наши примеры, используя BREAK :

 CONST i : Word = 1; s : Real = 0; N = 22;

 BEGIN WHILE TRUE DO BEGIN

 s:=s+SQR(i); INC(i); IF i>N THEN BREAK; END;

 WRITELN('сумма=',s);

 END.

 VAR c : Char;

 BEGIN c:=Pred(' ');

 REPEAT c:=Succ(c); WRITE(c); IF c=#255 THEN BREAK UNTIL FALSE;

 WRITELN;

 END.

Чтобы привести осмысленный пример использования процедуры CONTINUE, изменим условие второй задачи следующим образом: вывести на экран все символы с 32-го по 255-й, не являющиеся русскими буквами.

 VAR c : Char;

 BEGIN FOR c:=' ' TO #255 DO BEGIN

 IF (c>='А')AND(c<='Я')OR(c>='а')AND(c<='п')OR

 (c>='р')AND(c<='я') THEN CONTINUE;

 WRITE(c);

 END;

 WRITELN;

 END.

Впрочем, последнюю задачу, очевидно, можно решить проще:

 VAR c : Char;

 BEGIN FOR c:=' ' TO #255 DO BEGIN

 IF NOT((c>='А')AND(c<='Я')OR(c>='а')AND(c<='п')OR

 (c>='р')AND(c<='я')) THEN WRITE(c);

 WRITELN;

 END.