В.Н. Алексеев, Н.А. Никитина

Ишимский педагогический институт им. П.П. Ершова (филиал) ТюмГУ, г. Ишим

УДК 681.3

ИССЛЕДОВАНИЕ АЛГОРИТМА РАБОТЫ ЖЕСТКОГО ЦИКЛА В СИСТЕМЕ ПРОГРАММИРОВАНИЯ PASCAL ABC

Аннотация. В учебниках, пособиях и справках систем программирования часто приведены не полные, а иногда и ошибочные, сведения об алгоритме работы оператора цикла с параметром. В данной статье приведено описание различных нюансов работы алгоритма, реализующего оператор цикла с управляющим параметром в системе программирования Pascal ABC.

Ключевые слова: Оператор, алгоритм, управляющий параметр, оператор цикла с параметром, тело цикла.

В статье [1] описан ряд простых экспериментов по исследованию алгоритма работы оператора цикла с параметром (счетчиком) в системе Turbo Pascal. Это исследование, равно как и предпринятое в данной статье, имеет определенный практический интерес для программистов. Основой такого интереса являются ошибочные описания алгоритмов действий компьютера при исполнении оператора цикла со счетчиком. Примеры не адекватных описаний можно найти в различных учебниках и учебных пособиях по программированию. Пользуясь возможностью свободного оформления текстов Pascal-программ, будем далее формировать их фрагменты в линейной форме (с целью экономии места).

Рассмотрим [2, с. 26-27]. В указанном источнике утверждается, что первый вариант оператора for j:=expression1 to expression2 do statement эквивалентен следующей последовательности операторов: j:=expression1; k:=expression2; while j <= k do begin statement; inc(j) end;.

В учебном пособии [3, с. 30] фактически утверждается, что такой оператор цикла с управляющим параметром следует представлять в виде

следующей эквивалентной последовательности: temp1:=expression1; temp2:=expression2; j:=temp1; while j <= temp2 do begin statement; inc(j) end;.

На самом деле, как показано в [1], такой оператор нужно заменять следующей последовательностью операторов: temp1:=expression1; temp2:=expression2; if temp1 <= temp2 then begin j:=temp1; statement; while j <> temp2 do begin inc(j); statement; end; end;. Причем в этой конструкции, как и в предыдущем абзаце, temp1 и temp2 недоступны для непосредственного воздействия программиста.

В связи с этим очевидна несостоятельность притязаний на адекватное описание алгоритма работы жесткого (с параметром) цикла во многих учебниках и пособиях по языку Pascal различных версий. Например, в упомянутых выше источниках [2] и [3] не там указан момент инициализации управляющего параметра начальным значением (особенно катастрофично это в учебнике С.А. Немнюгина), не там находится "точка" изменения значения управляющего параметра перед очередной итерацией. Но самым "мифологическим" является общее утверждение авторов этих книг о том, что управляющий параметр имеет неопределенное значение по окончании работы цикла. Источник этого заблуждения (кстати, легко вскрываемого дизассемблированием) – неверное указание условия продолжения итераций (то есть и условия завершения цикла). На самом деле, как видно из правильного описания, после завершения работы управляющий параметр имеет в качестве текущего значения либо значение равное значению temp2 (если тело цикла исполнялось хотя бы один раз), либо результат инициализации, предшествующей выполнению цикла (0 при отсутствии таковой).

Рассмотрим версию языка Pascal, используемую в системе программирования Pascal ABC версия 3.0.1.26 (Freeware версия). Хотя сопровождение этой версии больше не ведется, но она бесплатна, доступна в любой школе, обладает простым интерфейсом, и вполне годится для стартового изучения языка. Всюду далее Pascal ABC будет обозначать именно эту версию. Причем такой выбор не принципиален, поскольку в других версиях (включая

сетевые), анализ работы оператора жесткого цикла можно провести аналогичным способом.

В системе помощи Pascal ABC о жестком цикле с параметром написано: «… Значение параметра цикла после завершения цикла считается неопределенным. … Изменение переменной-параметра цикла внутри цикла является логической ошибкой. …» (Содержание \ Справочник по языку \ Операторы \ Оператор цикла for). Вновь мы видим утверждение о неопределенности значения управляющей переменной цикла после его завершения (чем же эта переменная отличается от других, у которых все определено?). Также вызывает сомнение и второе отмеченное утверждение. Если нет явного запрета на изменение значения управляющего параметра в теле цикла, то программист должен знать к чему может привести "насильственная" попытка такого изменения.

Для проведения экспериментов, аналогичных описанным в [1], мы сконструируем "ловушку" для анализа нажатий клавиш. Оформим ее в виде процедуры, после завершения которой нужно "заглянуть" в буфер клавиатуры. Ресурсы модуля Crt позволяют легко собрать такую процедуру. Это функции без аргументов KeyPressed и ReadKey. Первая возвращает значения логического типа (false, если буфер клавиатуры пуст, и true в противном случае), вторая возвращает в точку вызова значения типа Char, извлекая информацию о последней нажатой клавише из буфера клавиатуры. Вот текст этой процедуры. Procedure Trap; Var Ch:Char; Begin While KeyPressed do Ch:=ReadKey; Repeat until KeyPressed End;

Эту процедуру мы "встроим" в тело цикла и при нажатии определенной клавиши, скажем 'i' (input) обеспечим возможность ввода значения управляющего параметра с клавиатуры. При нажатии любой другой клавиши будет продолжено "естественное" исполнение очередной итерации.

Приведем фрагмент диагностирующего теста – операторный блок (дополнение до работоспособной программы – задача читателя).

BEGIN

```
 j:=5; // j:=7; j:=9;
 For j:=3*j-10 to 4*j-17 do begin
   Write(j:5);
   Trap; Key:=ReadKey;
   If Key='i' then begin
     WriteLn; Write('j > ');ReadLn(j)
   end
 end;
 WriteLn; WriteLn('final j = ',j)
END.
```

Открывая по очереди указанные стартовые значения переменной цикла и проводя изменения значения этой переменной, приходим к следующим выводам. Значения стартового и финишного выражений, определяющих диапазон изменения параметра цикла, вычисляются один раз и больше не пересчитываются. Выход значения параметра за стартовое значение не контролируется (исполнение далее продолжается по направлению к финальному значению). При вводе значения, превышающего значение финишного выражения, исполнение цикла прекращается с введенным значением параметра. Инициализация управляющего параметра стартовым значением происходит всегда.

Анализ этих результатов позволяет описать работу оператора for j:=expression1 to expression2 do statement следующим образом. Указанный оператор эквивалентен следующему фрагменту: temp1:=expression1; temp2:=expression2; j:=temp1; If temp1<=temp2 then begin statement; While j<temp2 do begin Inc(j); statement end; end;

В системе Pascal ABC вход в тело цикла по метке запрещен (ошибка выявляется на этапе компиляции). А вот выход оператором goto из тела цикла разрешен, как "вперед" (на операторы после оператора цикла), так и "назад". При этом управляющий параметр имеет то значение, при котором управление получил оператор безусловного перехода.

В языке Object Pascal (система Delphi) указанный оператор эквивалентен фрагменту temp1:=expression1; temp2:=expression2; j:=temp1; While j<=temp2 do begin statement; Inc(j) end; Причем в теле цикла statement запрещено "принудительное" изменение управляющего параметра. Также отметим, что в этой системе дискретные типы "зациклены", как и в Turbo Pascal, а в Pascal ABC это свойство дискретных типов отсутствует [см. 1].

Итак, оператор жесткого цикла, не являясь базовой алгоритмической структурой, совершенно по-разному реализован коллективами разработчиков в различных оболочках. В связи с этим программист должен тщательно проанализировать и понять алгоритм его работы в используемом языке для различных ситуаций.

СПИСОК ЛИТЕРАТУРЫ

1. Алексеев В.Н. «Загадки» цикла с параметром в системах программирования QBasic и Turbo Pascal // Информатика: прил. к газете «Первое сентября». – 2004. - № 41(473). – С. 18-24.

2. Немнюгин С.А. Turbo Pascal. СПб.: Питер, 2000, 496 с

3. Епанешников А.М., Епанешников В.А. Программирование в среде Turbo Pascal 7.0. М.: Диалог-МИФИ, 1993, 288 с.