Операторы языка

Лекция 3

Простые операторы не содержат внутри себя других операторов

1. Простые операторы

- 1.1 Оператор присваивания
- 1.2 Оператор безусловного перехода
- 1.3 Оператор вызова процедуры
- 1.4 Пустой оператор.

- **2. Структурные операторы**
- 2.1. Составной оператор
- 2.2. Условные операторы
  - 2.2.1.Оператор условия if
  - 2.2.2.Оператор выбора case
- 2.3.Операторы повтора.
  - 2.3.1.Оператор while.
  - 2.3.2.Оператор повтора repeat
  - 2.3.3.Оператор повтора for
- 2.4.Вложенные операторы цикла

Оператор присваивания (:=)

- предписывает выполнить выражение, заданное в его правой части, и присвоить результат переменной, идентификатор которой расположен в левой части. Переменная и выражение должны быть совместимы по типу.
- A:=B+C;
- P:=25;
- C:=C+Pi;

содержание

Оператор безусловного перехода (go to)

- Означает "перейти к..." и применяется в случаях, когда после выполнения некоторого оператора надо выполнить не следующий по порядку, а какой-либо другой, отмеченный меткой оператор.
- Label 999;
- Var
- ...
- **go to 999;**

содержание

Оператор вызова процедуры

- служит для активизации предварительно определенной пользователем, или стандартной, процедуры.
- **ClrScr;**
- {Вызов стандартной процедуры очистки экрана}
- **InitWotrk(True);**
- {Вызов пользовательской процедуры}

содержание

Пустой оператор

- не содержит никаких символов и не выполняет никаких действий.
- A:=B;
- R:=2;
- ;
- K:=7.2;

содержание

Составной оператор

- представляет собой группу из произвольного числа операторов, отделенных друг от друга точкой с запятой, и ограниченную операторными скобками **begin** и **end**. *Синтаксическую диаграмму составного оператора можно представить в виде следующей схемы:*

содержание

Условные операторы

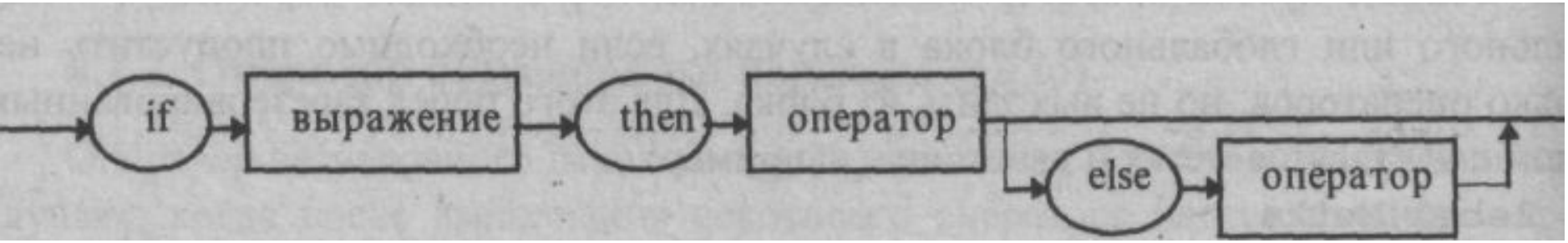
- предназначены для выбора к исполнению одного из воздействий (операторов) в зависимости от некоторого условия (при этом одно из действий может быть пустым, т.е. отсутствовать).

1.Оператор условия 1. Оператор условия if
2.Оператор выбора 2.Оператор выбора case

Оператор условия if

- **if <условие> then <оператор 1>**
  else <оператор 2> или
- **if <условие> then <оператор>**

Синтаксическая диаграмма оператора условия **if** выглядит таким образом:



Оператор выбора CASE

- CASE *<выражение>* OF
- *Константа 1: оператор 1;*
- *Константа 2: оператор 2;*
- ….
- *Константа N: оператор N;*
- END

Здесь **CASE** (в случае), **OF** (из), **END** (конец) – служебные слова.

содержание

Операторы повтора.

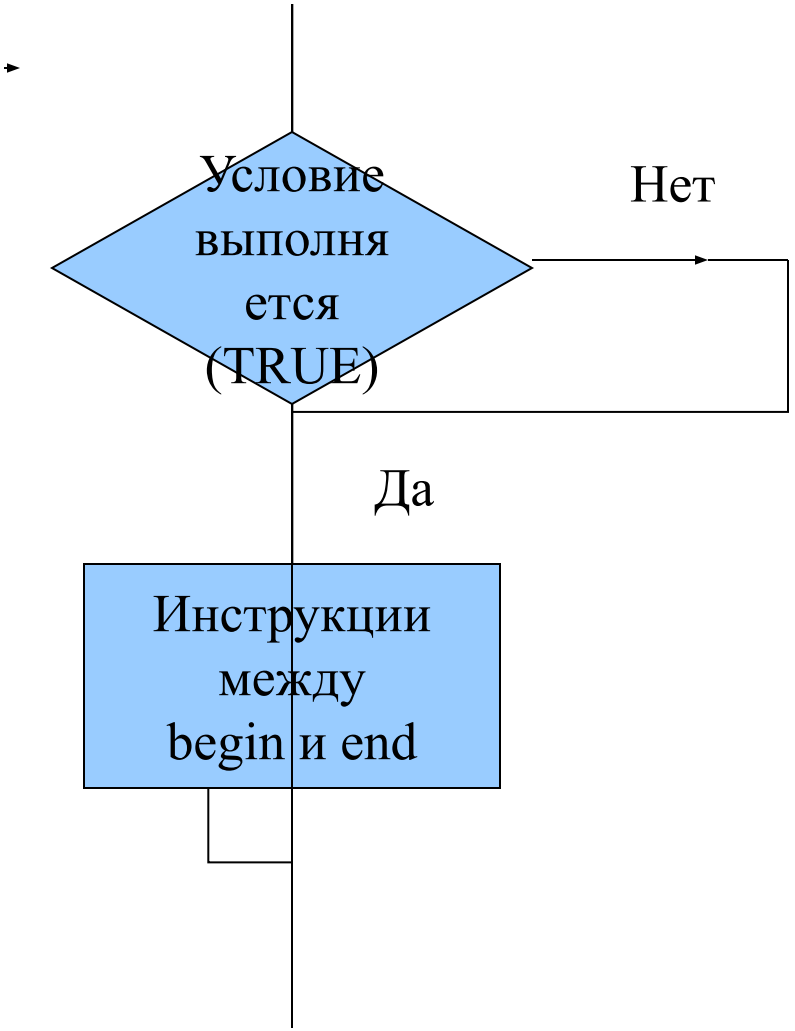
- повторяющиеся действия называются циклами и реализуются в программе с использованием инструкций циклов.
- **While**
- **repeat** **repeat**
- **for**

while

- Оператор **while** (пока) часто называют оператором цикла с предварительным условием (с предусловием). Используется в тех случаях, когда заранее неизвестно число повторений цикла.
- Форма записи оператора цикла с предусловием имеет вид:

WHILE <логическое выражение> **DO**
BEGIN
<операторы циклической части программы>
END

Здесь **WHILE** (пока) и **DO** (выполнить) – служебные слова.


Условие
выполня
ется
(TRUE)
Нет
Да
Инструкции
между
begin и end

содержание

Оператор цикла с последующим условием.

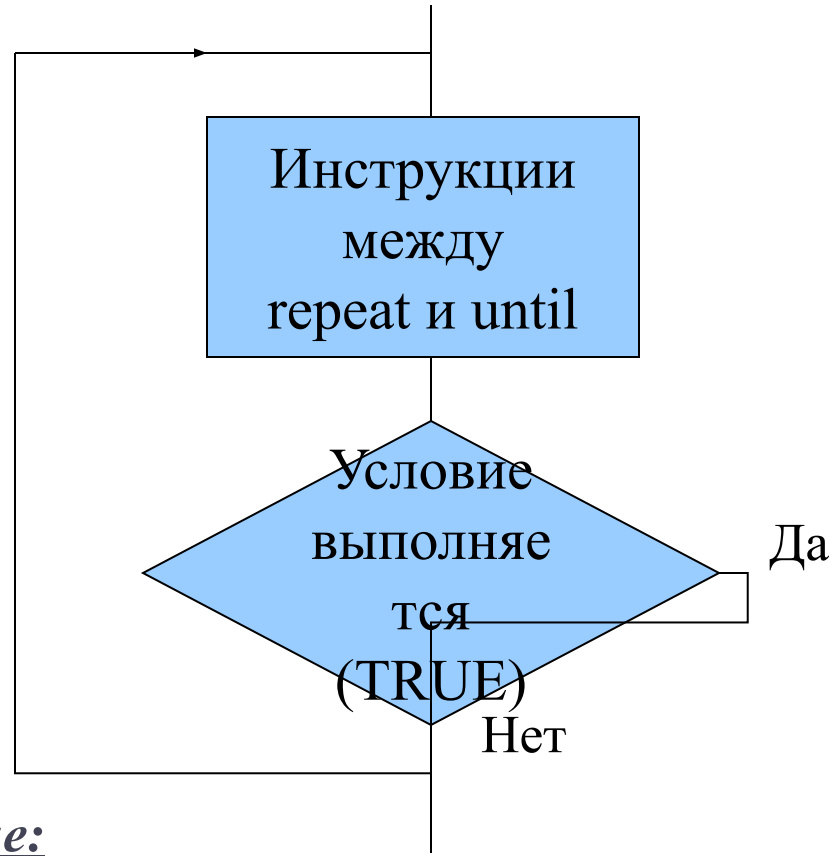
- Цикл с постусловием, как правило, используется в тех случаях, когда заранее неизвестно число повторений цикла. Оператор цикла имеет вид:

REPEAT

<операторы циклической части программы>

UNTIL <логическое выражение>

Здесь **REPEAT** (повторить) и **UNTIL** (до тех пор) – служебные слова.



Обратите внимание:

- последовательность инструкций между *repeat* и *until* всегда будет выполнена хотя бы один раз;
- для того, чтобы цикл завершился, необходимо, чтобы последовательность операторов между *repeat* и *until* изменяла значения переменных, входящих в выражение условие.

содержание

Оператор цикла с параметром.

- В случаях, когда число повторений может быть заранее известно, для организации циклической обработки информации применяется оператор повтора **for**.
- Шаг +1: **FOR i:=n TO m DO**

BEGIN

<Операторы циклической части программы>

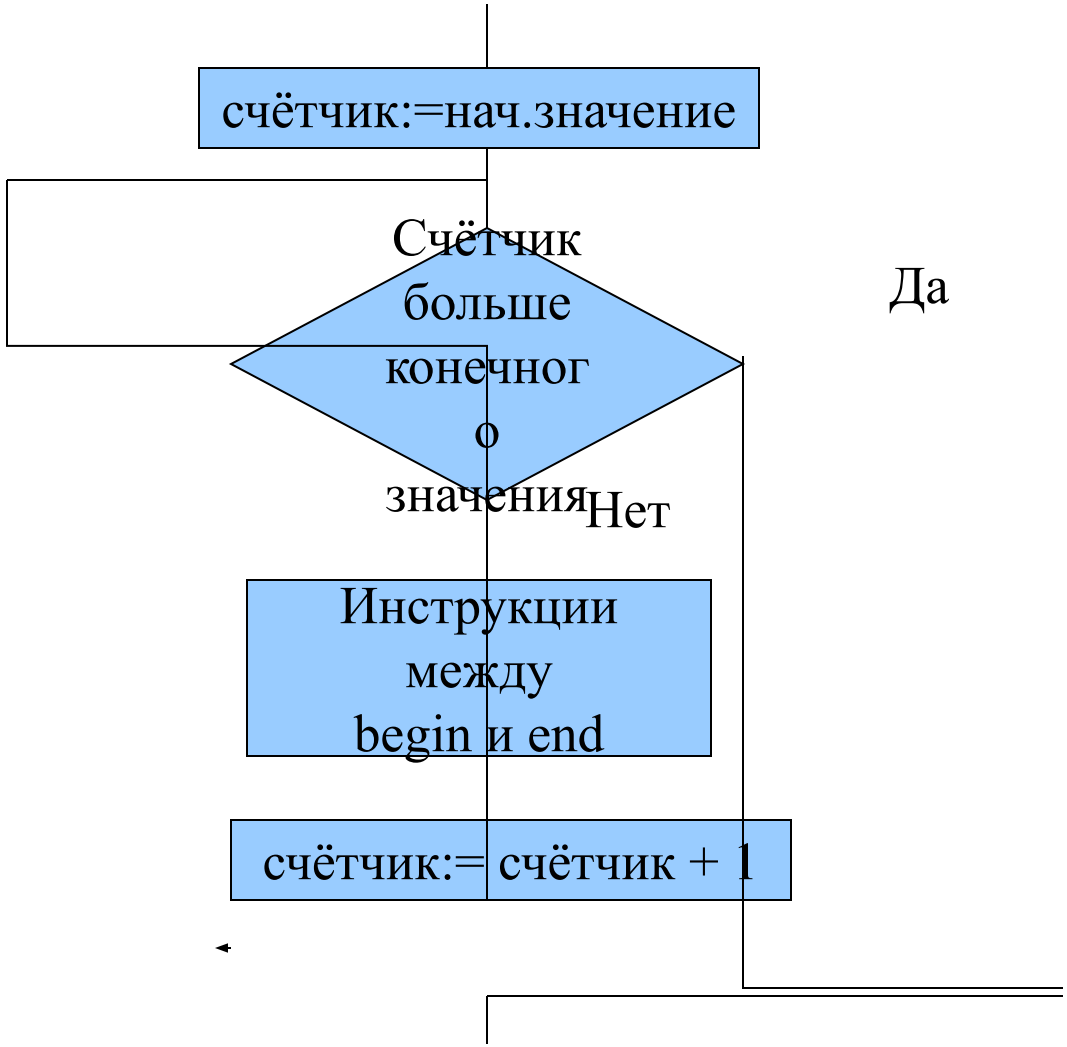
END

- Шаг -1: **FOR i:=n DOWNTO m DO**

BEGIN

<Операторы циклической части программы>

END



Обратите внимание, что в случае, если начальное значение счётчика превышает конечное значение счётчика, то последовательность операторов между *begin* и *end* ни разу не будет выполнена.

содержание

Вложенные циклы

- Если в теле цикла присутствует циклическая структура, то такие циклы называются вложенными**.** Цикл, содержащий в себе цикл, является внешним, а цикл, содержащийся внутри другого цикла, является внутренним. Внешний и внутренний циклы могут быть трех видов: цикл с предусловием **while**, цикл с постусловием **repeat** или циклами с параметрами **for**.

задача вывода на экран таблицы умножения, предполагает при решении использование вложенных циклов.

```
Program Tabl_Umn;
var
i, j: byte;
begin
 For i:=1 to 10 do   {внешний цикл}
      For j:=1 to 10 do   {внутренний цикл}
      writeln(i,'*',j,'=',i*j);    { тело внутреннего цикла}
end.
```

При записи операторов необходимо помнить, что:

- точка с запятой не ставиться в разделах описаний после зарезервированных слов **unit, uses, label, type, const, var** и ставиться после завершения каждого описания;
- слова **begin** и **end** являются операторными скобками, а не операторами, поэтому точка с запятой не ставиться после слова **begin** и перед **end**;
- точка с запятой является разграничителем операторов, ее отсутствие между операторами вызывает ошибку компиляции;
- в операторах цикла точка с запятой не ставиться после **while, repeat, do** и перед **until**;
- в условных операторах точка с запятой не ставиться после **then** и перед **else**.