Программирование на языке Turbo Pascal

Выход

Содержание:

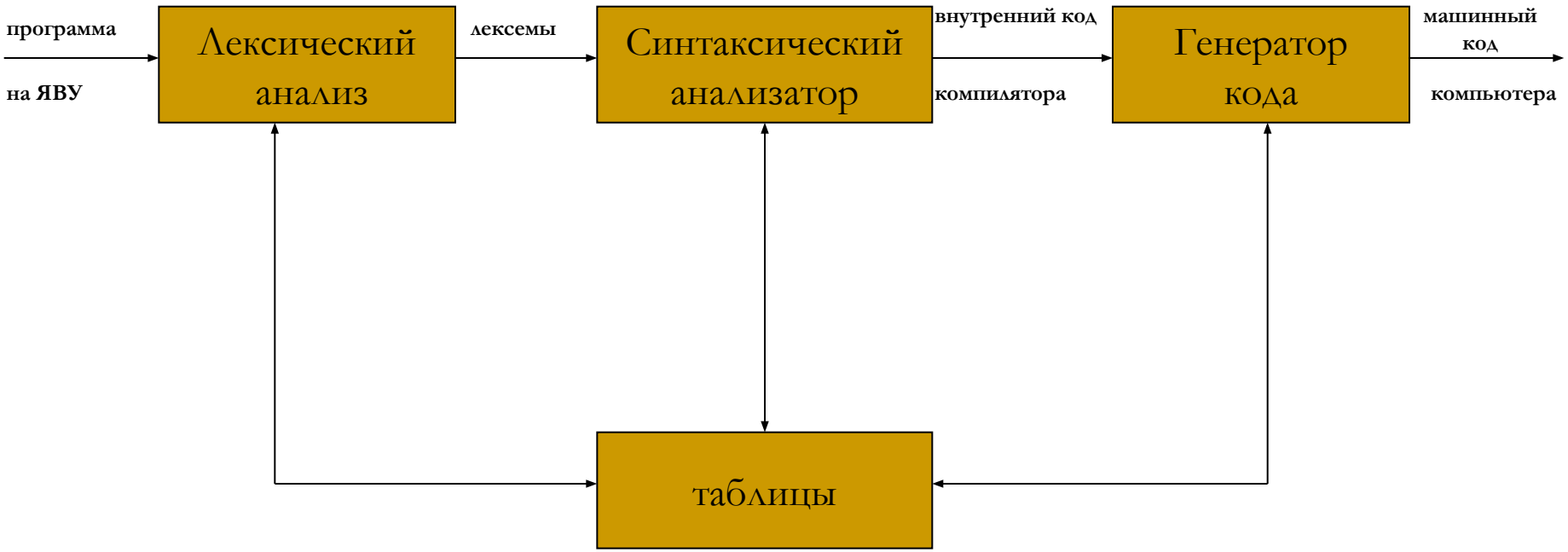
Выход

Тема: Программирование на языке Turbo Pascal

Перевод программ, написанных на языках программирования высокого уровня, к числу которых относиться и язык **Turbo Pascal**, входящий в состав профессионального пакета разработки программ **Borland Pascal with Objects 7.0** перевод на язык машинных кодов, выполняемых компьютером, осуществляется специальными программами, которые называются ***трансляторами***.

Трансляторы

По способу работы трансляторы делятся на ***компиляторы*** и ***интерпретаторы***



Упрощенная модель компилятора

Набор символов

Прописные и строчные буквы латинского алфавита, а также символ подчеркивания, который использует наравне с буквами.

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

(коды ASCII от 65 до 90)

a b c d e f g h i j k l m n o p q r s t u v w x y z

(коды ASCII от 97 до 122)

_ (код ASCII 95)

Арабские десятичные цифры.

0 1 2 3 4 5 6 7 8 9
(коды ASCII от 48 до 57)

Специальные символы (в скобках указан код ASCII).

#	(35)
$	(36)
‘	(39)
(	(40)
)	(41)
*	(42)
+	(43)
,	(44)

-	(45)
.	(46)
/	(47)
:	(58)
;	(59)
<	(60)
=	(61)
>	(62)

@	(64)
[	(91)
]	(93)
^	(94)
{	(123)
}	(125)

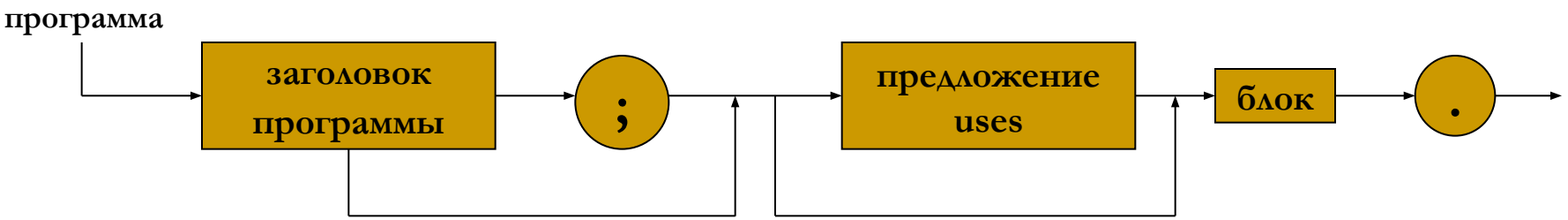
Символ пробела (код ASCII 32).

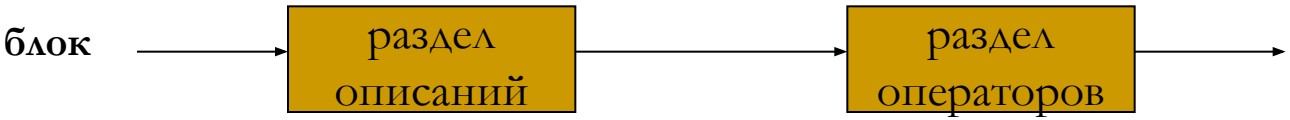
Управляющие символы (коды ASCII от 0 до 31).

Содержание:

Выход

Синтаксические диаграммы общей структуры программы на языке **Turbo Pascal** имеют такой вид:





Выход

Структура программы

{ I. Заголовок программы}

program Имя _ Программы;

{II. Раздел указания используемых модулей}

uses Список _ Используемых _ Модулей;

{III. Раздел описаний}

label Описания _ Меток;

const Описание _ Констант;

type Описание _ Типов;

var Описание _ Переменных;

procedure

function Описание _ Процедур _ и _ Функций

exports Описание _ Экспортируемых _ Имен;

{IV. Раздел операторов (Операторный блок)}

begin

Операторы

end.

Содержание:

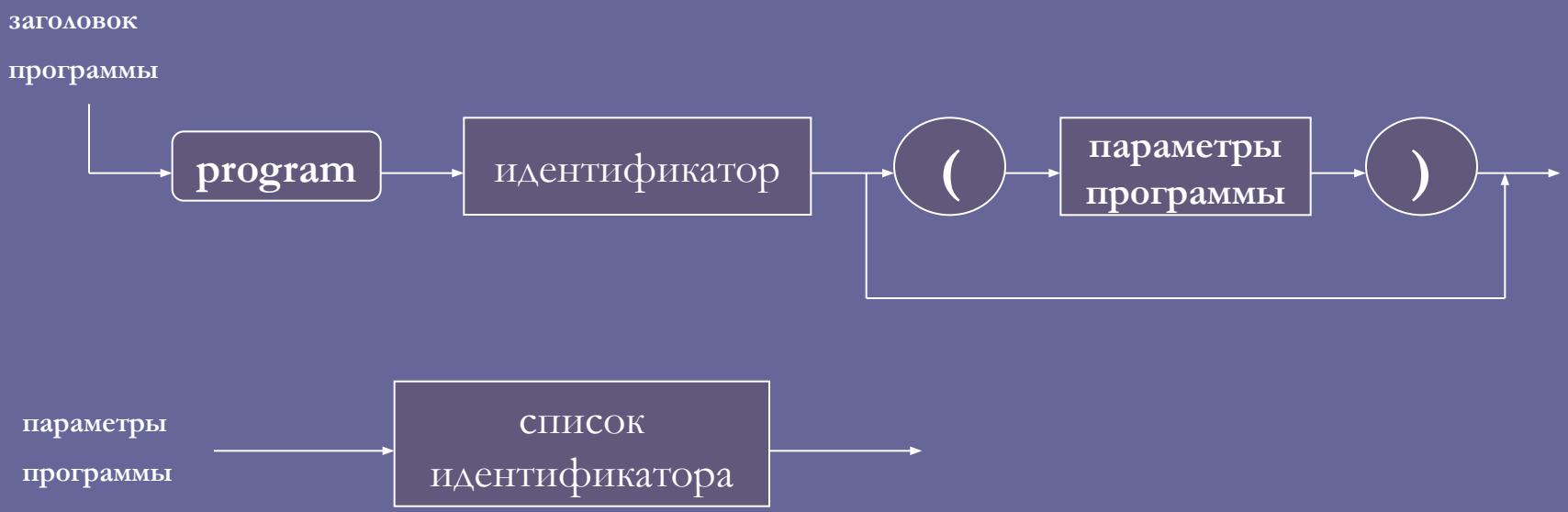
Выход

I. Заголовок программы

Program имя _ программы;

{имя программы одно целое слово}

Прежде чем работать с переменными необходимо описать до начало программ. Примеры:



Выход

Примеры:

```
program   Simple;
program   Print (Output);
program   GetPut (Input, Output);
program   Complex (Input, Output, MyFile);
```

Выход

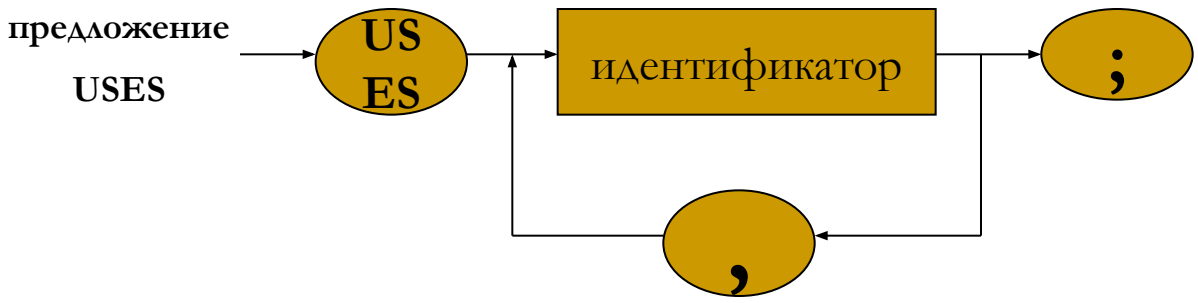

II. Раздел указаний используемой модули

Раздел указаний используемых модулей начинается с зарезервированного слова ***USES***.

USES список _ используемых _ моделей;
USES – использование.

Она описывается в случаях, если в программе используются константы, типы, переменные, процедуры или функции, определенные в стандартных модулях **Turbo Pascal**, кроме модуля **System**, или в модулях, созданных пользователем. Примеры:

Синтаксис предложения ***USES*** имеет следующий вид:



Выход

Примеры:

uses Crt, Graph;

uses Crt, Graph, Mylib, Stack;

Предложение ***uses*** в каждой отдельной программе может быть описано *только однократно* и должно располагаться *непосредственно после заголовка программы*.

III. Раздел описаний

label – описание _ меток;

const – описание _ констант (постоянная величина);

B=7 – меняться не может

B=B+2 – нельзя изменять значение **const** на протяжении всей программы.

type – описание типов (тип пользовательский, комбинированный и стандартный).

var – описание _ переменных (чисел); **(var x, y byte);**

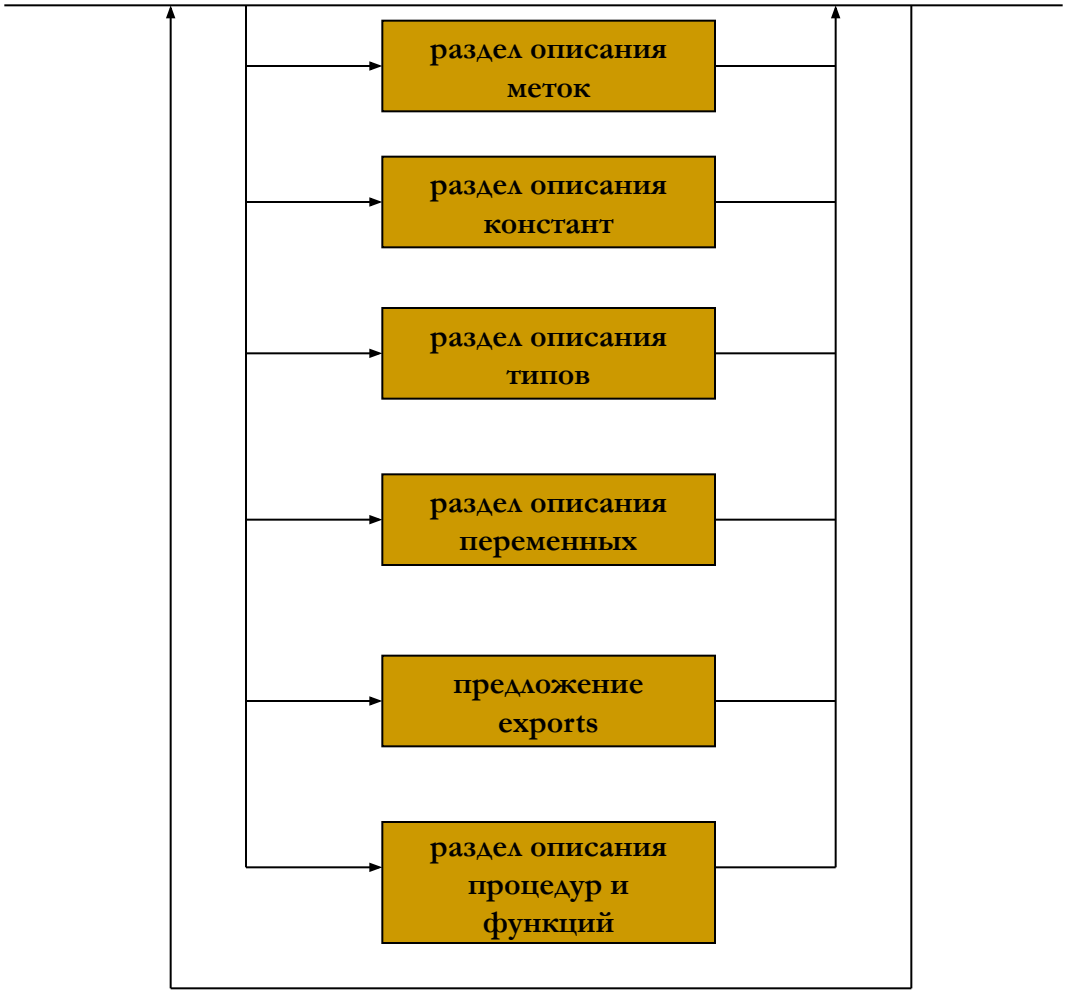
Переменные – идентификатор, который может принимать любые значения из диапазона, заданного каким – либо типом.

export – описание _ экспортных имен;

Пример:

Примеры:

раздел описаний

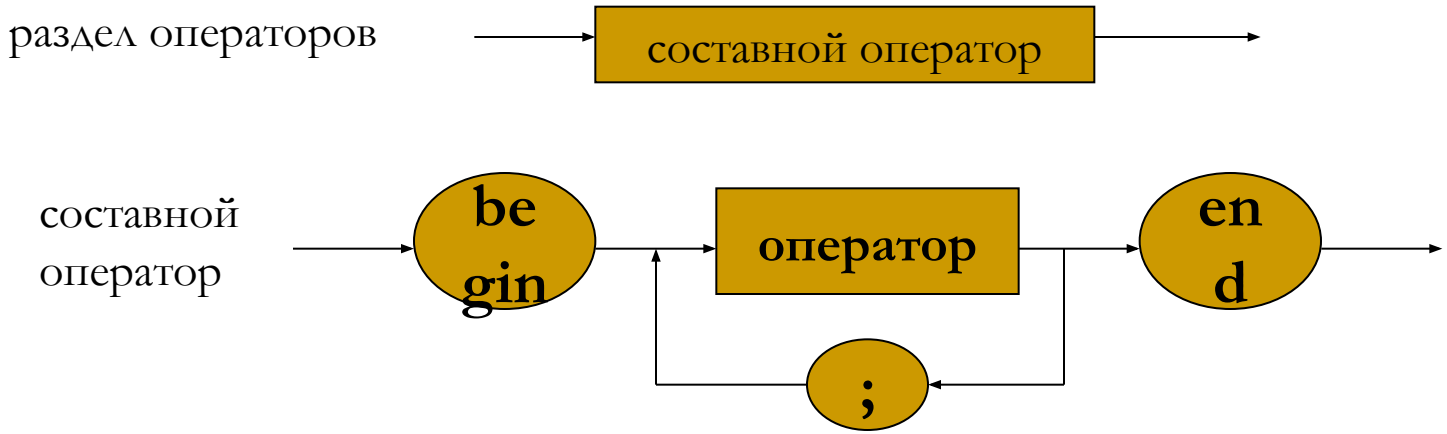


Выход

IV. Раздел операторов (операционный блок)

```
begin
    Writeln ('Hello, Word')
end.
```

Синтаксис раздела операторов имеет вид:



Выход

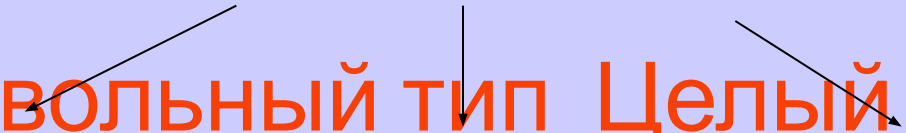
Содержание:

Выход

Стандартные типы данных:

- группу целых типов (Shorting, Integer, Longint, Byte, Word);
- группу вещественных типов (Single, Real, Double, Extended, Comp);
- группу булевских типов (Boolean, ByteBool, WordBool, LongBool);
- символьный тип (Char);
- строковые типы (String, Pchar);
- указательный тип (Pointer);
- текстовый тип (Text).

Порядковый тип



Символьный тип Целый тип

Булевский тип

Выход

Порядковые типы характеризуются *4*-мя свойствами*:*

- Множество документальных значений*;*
- Любой порядковый номер представляет собой упорядочную последовательность, каждый элемент который имеет свой порядковый номер*;*
- Порядковый номер представляется целым числом *1*-ое значение любого порядкого типа имеет порядковый номер *0*, следующие значение имеет порядковый номер *1* и т.д. Исключительно составляют порядковые типы*: Shortint, Integer, Longint.*

Выход

К любому значению порядкового типа можно приметить стандартную функцию **SUCC**, порядкового номера последнего значения **PRED (0)-255, PRED (9)-10**.

Если эта функция применяется последнему допустимому значению любого порядкового типа, кроме булевских, то возвращая к порядковым номером первого значение **SUCC (255)-0**.

Группа целых типов

Название типа	Идентификатор	Диапазон представления чисел	Размер памяти
Короткое целое со знаком	Shortint	-128…127	1 байт
Целые со знаком	Integer	-32768…32767	2 байта
Длинные целое со знаком	Logint	-2147483648…2147483647	4 байта
Короткое целое без знака	Byte	0…255	1 байт
Целые без знака	Word	0…65535	2 байта

Выход

Группы вещественных типов

Название типа	Идентификатор	Диапазон, представления чисел	Значащие цифры Мантиссы	Размер памяти
Вещественно одинарной точности	Single	$-1,5*10^{-45}$ до $3,4*10^{38}$	7..8	4 байта
Вещественное	Real	$2,9*10^{-39}$ до $1,7*10^{38}$	11..12	6 байт
Вещественное двойной точности	Double	от $5,0*10^{-324}$ до $1,7*10^{308}$	15..16	8 байт
Вещественное повышенной точности	Extended	$3,4*10^{-4932}$ до $1,1*10^{4932}$	19..20	10 байт
Целое в формате вещественного	Comp	от $-2^{63}+1$ до $2^{63}-1$ или $-9,2*10^{18}$ до $9,2*10^{18}$	19..20	8 байт

Выход

Группы булевских типов

Идентификатор типа	Значение False соответствует	Значение True соответствует	Размер памяти
1. Boolean	Число 0	Любое число отличное от 0	1 байт
2. Bytebool	Число 0		1 байт
3. Wordbool	Число 0 в обоих байтах		2 байта
4. Longbool	Число 0 в во всех байтах		4 байта

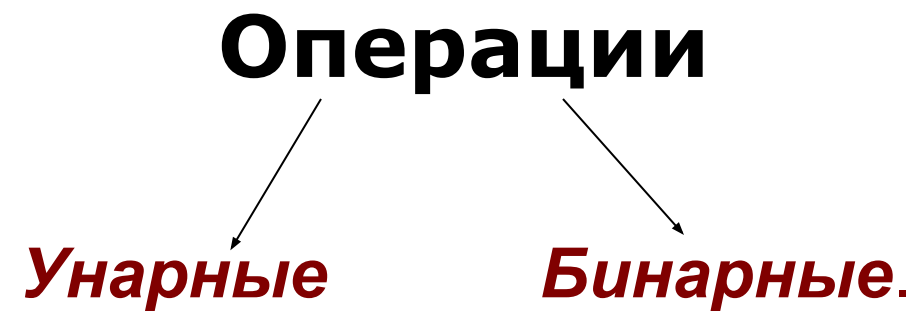
Fool – нет чисел больше 0

Try – есть числа меньше 0

Не дает конкретного значения

Выход

Операции. Приоритеты операции.



Примеры.

Выражение	Результат
-7	-7
-(-9)	9
not False	True

Примеры.

Выражение	Результат
5+7	12
(4-2)*5+10	20
True or False	True

Приоритеты операции

Приоритет	Операции	Категории операции
Первый (высший)	+ - not @	Унарные операции
Второй	* / div mod and shr shl	Бинарные операции типа умножения
Третий	+ - or xor	Бинарные операции типа сложения
Четвертый (низкий)	= <> >= <= in	Бинарные операции отношения

Классификация операций

По характеру выполняемых действий операции можно разделить на следующие группы:

1. Арифметические операции:
   - унарные: +, -
   - бинарные: +, -, *, /, div, mod.
2. Операции отношения

=, <>, <, >, <=,>=

3. Булевские (логические) операции

not, and, or, xor

4. Поразрядные логические и сдвиговые операции

not, and, or, xor, shl, shr

5. Строковая операция (конкатенация)

+

6. Операции над множествами

+, -, *, in, <=, >=

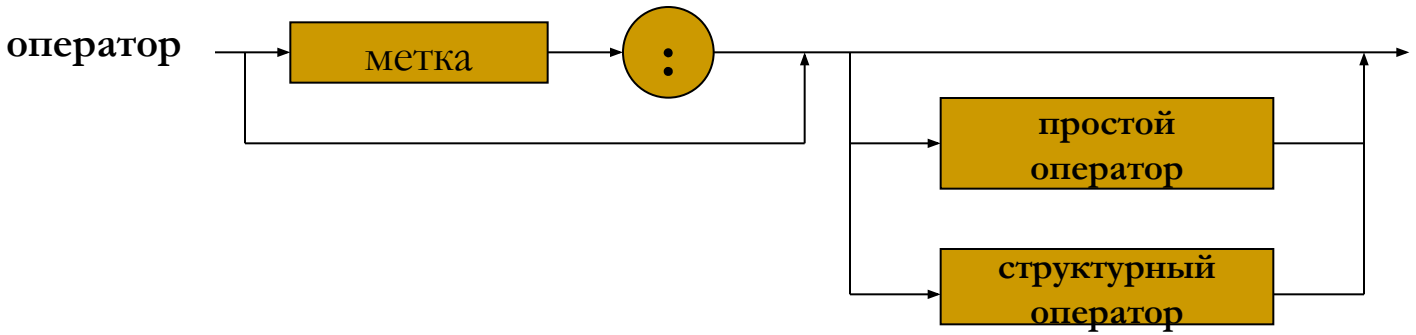
7. Операция взятия адреса

@

Содержание:

Выход

Операторы предназначены для описания действий, которые будут выполнены при реализации алгоритма.



Операторы:

- Простые операторы;
- Структурные операторы;

Выход

Простые операторы – это такие операторы, которые не содержат в себе других операторов.

Простые операторы:

- **Оператор присваивания** *(выполнение операторов присваивания приводит к вычислению значения, определяемого выражением, и присваиванию этого значения переменной, идентифицируемой именем, стоящим слева от символа присваивания);*

- **Оператор процедуры** *(Выполнение оператора процедуры проводит к активизации действий, описанных в ее теле) ;*

- **Оператор перехода** – оператор состоит из ключевого слова **goto** *(Выполнение оператора **goto** приводит к передаче управления на оператор, перед которым стоит указанная в операторе **goto** метка).*

Структурные операторы включают в себя другие операторы и управляют последовательностью их выполнения.

В Turbo Pascal *структурными операторами* являются:

1. составной оператор;
2. условные оператор;
   - оператор альтернативы if;
   - оператор выбора case;
3. операторы цикла:
   - оператор цикла с предусловием while;
   - оператор цикла с постусловием repeat;
   - оператор цикла со счетчиком for;
4. оператор для записей with.

Условный оператор *if*

Оператор if

Неполная форма

Полная форма

Условный оператор if может быть записан в полной и неполной форме. *Примеры:*

Выход

Примеры:

Неполная форма:

```
if Выражение
then Оператор
```

Полная форма:

```
if Выражение
then Оператор 1
else Оператор 2
```



При выполнение условного оператора сначала вычисляется Выражение, результат которого может принимать только **булевский тип**, а затем, в зависимости от значения результата (True, False), выполняется или оператор 1, стоящий после ключевого слова then (если результат равен True) или оператор 2, стоящий после ключевого слова else (если результат равен False).

После ключевых слов **then** и **else**, может стоять всего лишь один оператор.

Количество операторов в ветви		Обобщенные формы оператора if
Then	**Else**	
Один	Один	If Выражение then оператор else оператор
Несколько	Один	If Выражение then оператор оператор; оператор; … оператор; end else оператор.
Один	Несколько	If Выражение then оператор else оператор оператор; оператор; … оператор end.

Выход

Количество операторов в ветви		Обобщенные формы оператора if
Then	**Else**	
Несколько	Несколько	if Выражение then begin оператор; оператор; … end else begin оператор; оператор; … оператор; end.

Выход

Операции ввода и вывода на экран с клавиатуры

Оператор ввода

ReadIn (x) [,x2, x3, …] – сначала идет имя процедуры ReadIn, зетам имена переменных, которым будут присвоены значения. Их может быть несколько либо одно. После ввода ReadIn каждой переменной требует переход на новую строку, иначе часть введенной строки отбрасывается.

Оператор вывода

WriteIn ([‘Текст пояснения ‘,] x1 [, x2, x3, …]) – сначала идет имя процедуры WriteIn,зетам текст, имена переменных, арифметическое выражение. Их может быть несколько либо одно имя. WriteIn переводит курсор на новую строку в отличии он Write. Арифметическое выражение вычисляется, а затем выводится. При выводе вещественных чисел необходимо формирование, иначе они будут представлены в форме с плавающей запятой. WriteIn (real 2:5:2). Здесь переменная real 2, выводиться на экран ограниченная по ширине в 5 символов, а также с 2 знаками после запятой.

Выход

Оператор цикла с предусловием while.

Оператор While является универсальной управляемой конструкцией. С помощью него можно записать любое циклическое действие. Оператор While позволяет многократно выполнять одни и те же действия в зависимости от некоторого условия.

Если тело цикла состоит из одного оператора	Если тело цикла включает более одного оператора
while Условие do Оператор	*While* Условие do *begin* Оператор; Оператор; . . . Оператор; *end.*

Оператор цикла с постусловием repeat.

Оператор цикла с постоянным условием, состоит из ключевых слов repeat, после которого замыкает слово until, после которого указывается условие выполнения цикла.

repeat	**повторить**
Операторы	**данные операторы, пока не выполняют условие**
until	

Цикл с предусловием while.	Цикл с постусловием repeat.
1. До начала цикла должны быть сделаны начальные установки цикла для корректного входа в него.	
2. В теле цикла, должны присутствовать операторы изменяющиеся переменным условием так, чтобы цикл через некоторое число повторение завершилось (ограничивается ресурсами), (не ограничивается понятие бесконечно и сколько угодно).	
3. Пока условие истинна.	3. Цикл работает, пока условие ложное.
4. Цикл завершается, когда условие становится ложным.	4. Цикл завершается, когда условие становится истинным.
5. Цикл может не выполнится не разу, если исходное значение условия при входе в цикл ложное.	5. Цикл обязательно выполняется минимум один раз.
6. Если в теле цикла требуется более одного оператора, то необходимо использовать со ставкой оператор	6. Не зависимо от количества операторов или цикла использования составление оператора (дополнительно begin и end не требуется).

Содержание:

Выход

Массив

Массив – это структура данных, которая представляет собой однородную, фиксированную по размеру и конфигурации совокупность элементов простой или составной структуры, упорядоченных по номерам. **Пример:**

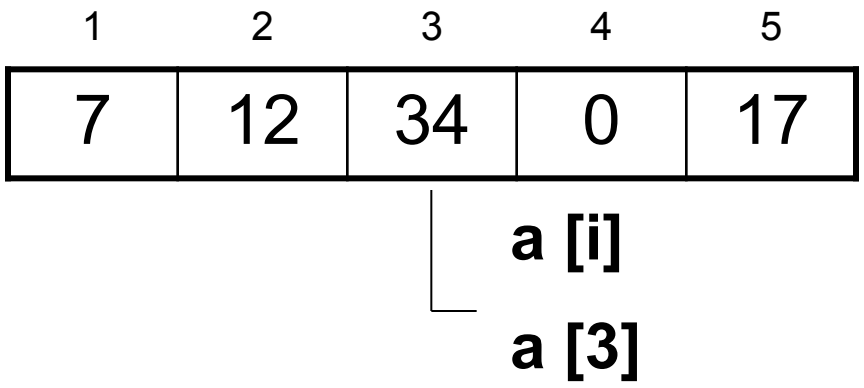
Выход

Пример:

Массив – однотипный [1..10] *A – integer.*

a: array

a



Элемент массива характеризуется порядковым номером и конкретного значения

Три основных типа массива:

1. Одномерный. Иначе его называют **вектором**. Все данные записываются последовательно друг за другом, каждый элемент имеет только одну координату. Пример:

2. Двухмерный. Это структура имеющих две координаты (строки и столбцы). Пример:

3. Трехмерный. Каждый элемент определяется 3 – мя координатами (ширина, высота, глубина). Пример:

Выход


Пример:

for I =1 to 10 do

readln (a [i]) - заполнение одномерного массива.

for I =1 to 10 do

write (a [i]:8) - вывод одномерного массива на экран.

Выход

Пример:

Если записать в столбик, то writeln –

2 – х мерный массив.

```
for l = 1 to 10 do
for j = 1 to 2 do
readln (a [l, j]).
```

Выход

Пример:

Двух мерный массив (матрица)

Матрица – эта главная диагональ, элементы главной диагонали имеют одинаковые индексы.

Выход

Пример:

3 – х мерный массив:

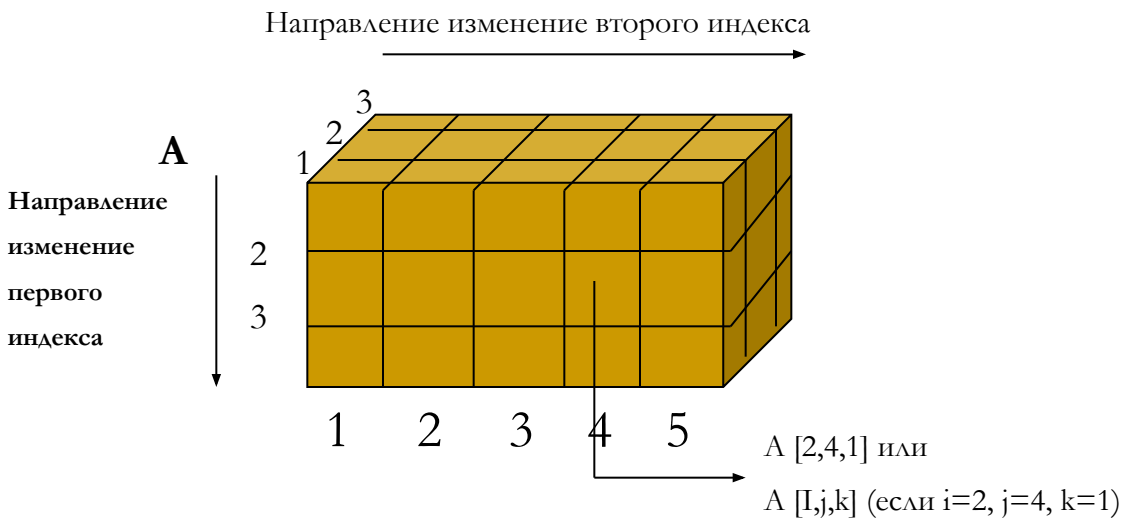
for I = 1 to 10 do

for j = 1 to 2 do

for k = 1 to 3 do

readln (a [I, j, k]).

a [I, j, k] – пространственно напоминает параллелограмм.



Выход