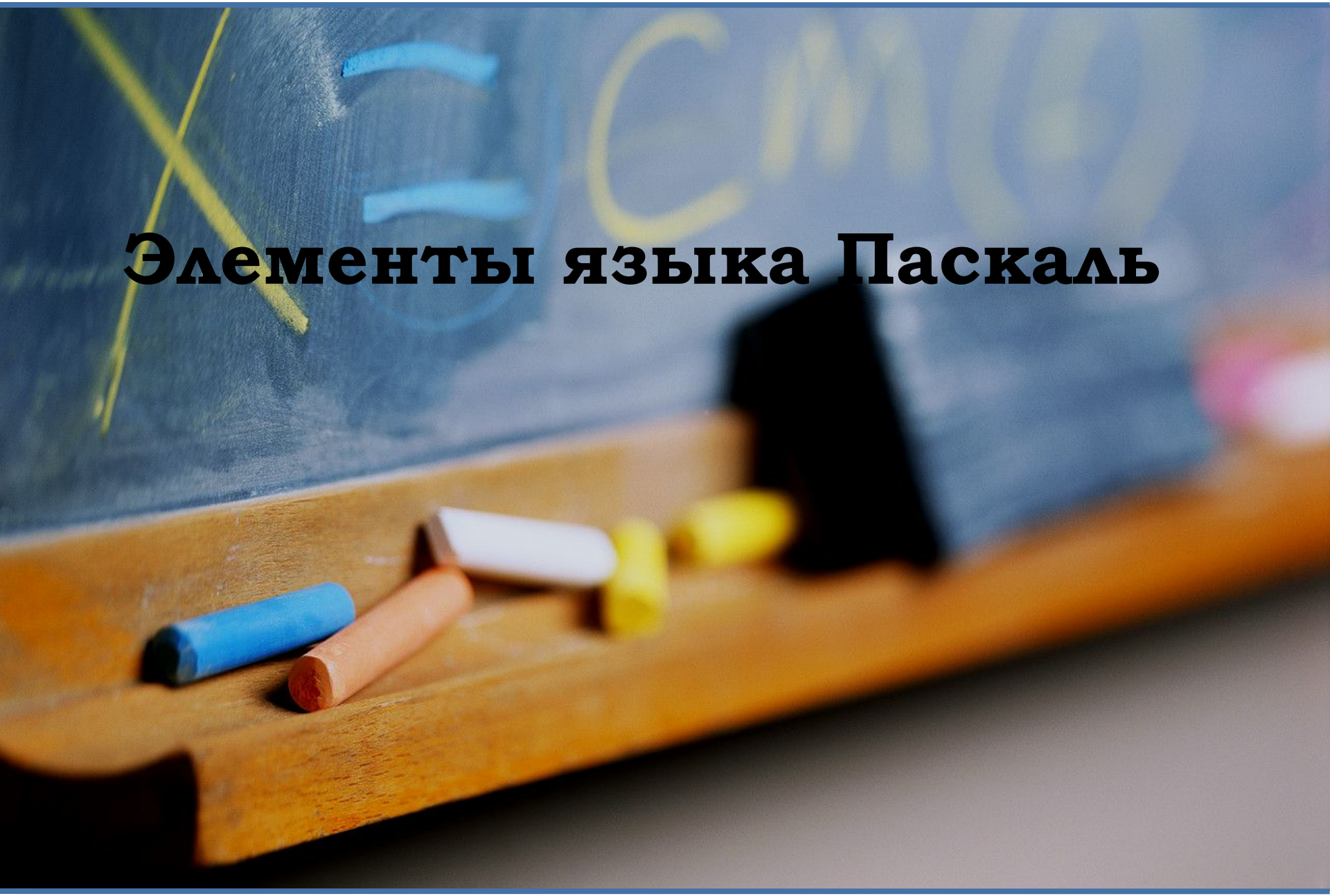


Элементы языка Паскаль

Язык Паскаль

Язык Паскаль - язык профессионального программирования, который назван в честь французского математика и философа Блеза Паскаля (1623-1662) и разработан в 1968-1971 гг. Никлаусом Виртом.

Первоначально был разработан для обучения, но вскоре стал использоваться для разработки программных средств в профессиональном программировании.

Структура Паскаль - программ

Блок типа PROGRAM	латинских букв и цифр. Его присутствие не обязательно, но рекомендуется записывать для быстрого распознавания нужной программы среди других •раздел описания модулей (uses);
Программный блок	•раздел описания констант (const); •раздел описания типов данных (type); •раздел описания переменных (var);

Общая структура программы на языке Паскаль

Program ИМЯ..; {заголовок программы}

Uses ...; {раздел описания модулей}

Var ..; {раздел объявления переменных}

...

Begin {начало исполнительной части программы}

... {последовательность
... операторов}

End. {конец программы}

Пример программы на языке Паскаль

```
Program  z1;
{ Программа для вычисления площади прямоугольника
    по заданным сторонам}
Var
     a,b,s : integer;
Begin
     writeln (‘Введите стороны A и B’);
     read (a, b);                          { ввод чисел a и b }
     s:=a*b;                               { s – площадь }
     write (‘S=‘,s,’ кв.см.’)
End.
```

Идентификаторы

Имена, даваемые программным объектам (константам, типам, переменным, функциям и процедурам, да и всей программе целиком) называются **идентификаторами**.

Должно удовлетворять следующим требованиям:

- длина имени не должна превышать 63 символа,
- первым символом не может быть цифра,
- переменная не может содержать пробел;
- имя не должно совпадать с зарезервированным (служебным) словом;
- прописные и строчные буквы воспринимаются одинаково.

Примеры зарезервированных слов

and	goto	set	
array	implementation	shl	
begin	in	shr	
case	interface	string	
const	label	then	
div	mod	text	
do	nil	to	
downto	not	type	
else	of	unit	
end	or	until	
file	pointer	uses	
far	procedure	var	
for	program	while	
forward	record	with	
function	repeat	xor	

Переменные и типы данных

Переменная – это программный объект, значение которого может изменяться в процессе работы программы.

Тип данных – это характеристика диапазона значений, которые могут принимать переменные, относящиеся к этому типу данных.

Все используемые в программе переменные должны быть описаны в специальном разделе **var** по следующему шаблону:

var
<имя_переменной_1> [, <имя_переменной_2, _>] : <имя_типа_1>;
<имя_переменной_3> [, <имя_переменной_4, _>] : <имя_типа_2>;

О типах данных

Для временного хранения информации в операторах памяти машины в языке Паскаль используются константы и переменные. Они могут быть различных типов:

Простые:
- integer – целых чисел;
- real – действительных чисел;
- char – символьный тип;
- string – строковый;
- boolean – логический;

Сложные:
- record – комбинированный;
- set – множественный

и другие.

Приведем пример описания переменных:

```
Var
    a : integer;
    b,c : real;
    m : boolean;
```

Константы

Константа – это объект, значение которого известно еще до начала работы программы.

В языке Pascal существует три вида констант:

- неименованные константы (цифры и числа, символы и строки, множества);
- именованные нетипизированные константы;
- именованные типизированные константы.

Неименованные константы

Неименованные константы не имеют имен, и потому их не нужно описывать.

Примерами использования неименованных констант могут послужить следующие операторы:

```
I := -10;
r := 12.075 + x;
c := 'z';
s := 'abc' + st;
s5 := [1,3,5] * s6;
b := true;
```

Нетипизированные константы

Именованные константы, как следует из их названия, должны иметь имя.

Эти имена необходимо сообщить компилятору, то есть описать в специальном разделе **const**.

Если не указывать тип константы, то по ее внешнему виду компилятор сам определит, к какому (базовому) типу ее отнести.

Нетипизированные константы

Вот несколько примеров описания нетипизированных именованных констант:

```
const
  n = -10;
  m = 1000000000;
  mmm = n*100;
  x = 2.5;
  c = 'z';
  s = 'string';
  b = true;
```

Типизированные константы

Типизированные именованные константы представляют собой ***переменные(!)*** с начальным значением, которое к моменту старта программы уже известно.
Типизированные константы нельзя использовать для определения других констант, типов данных и переменных.
Их значения можно изменять в процессе работы программы.

Описание типизированных констант производится по следующему шаблону:

```
const
<имя_константы> : <тип_константы> = <начальное_значение>;
```

Типизированные константы

Пример описания типизированных констант

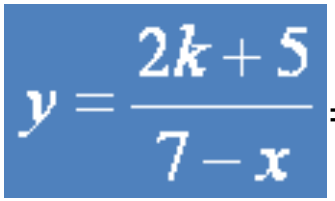
```
const
    n: integer = -10;
    x: real = 2.5;
    c: char = 'z';
    b: boolean = true;
```

Операции и выражения. Арифметические операции.

Операции общей арифметики
(арифметические оперции)

\+ сложение
\- вычитание
* умножение
/ деление

Пример арифметического выражения :



=(2*k+5)/(7-x);

Арифметические операции

Операции целочисленной арифметики применимы, как легко догадаться, только к целым типам.

a **div** b – деление a на b нацело

Пример:

```
x:=13;
y:=5;
z:=x div y;
```

В результате переменная **z** получит значение 2.

Арифметические операции

a **mod** b – взятие остатка при делении a на b нацело.

Пример:

```
x:=13;
y:=5;
z:=x mod y;
```

В результате переменная **z** получит значение 3.

Логические операции

Операции сравнения

= равно

<> неравно

> больше

< меньше

<= меньше либо равно

>= больше либо равно

Применимы ко всем базовым типам.

Результатом является значение **истина** (true) или **ложь** (false)

Логические операции

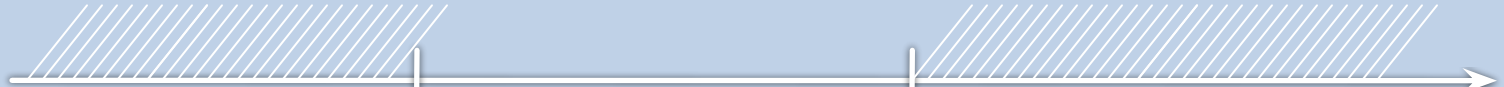
Пример операций сравнения:

5>=4	true
7<=7	true
(2+7)<3	false
true=false	false
a>b	зависит от значений a и b

Логические операции

and	логическое «и» (конъюнкция)
or	логическое «или» (дизъюнкция)
not	логическое «не» (инверсия)
xor	логическое «или исключающее»

(x>2) and (x<5)



(x<2) xor (x>5)

Порядок вычислений

Приоритеты операций языка Pascal

	Операции	**Приоритет**
Унарные операции	not	Первый(высший)
Операции, эквивалентные умножению	*, /, div, mod, and	Второй
Операции, эквивалентные сложению	+,–, or, xor	Третий
Операции сравнения	=, <>, >, <, <=, >=	Четвертый

Стандартные математические функции

Функция	*Описание*
abs(x)	Абсолютное значение (модуль) числа
arctan(x)	Арктангенс (в радианах)
cos(x)	Косинус (в радианах)
exp(x)	Экспонента (e^x)
frac(x)	Взятие дробной части числа
int(x)	Взятие целой части числа
ln(x)	Натуральный логарифм (по основанию e)
odd(x)	Проверка нечетности числа
pi	Значение числа π
round(x)	Округление к ближайшему целому
trunc(x)	Округление "вниз" - к ближайшему меньшему целому
sin(x)	Синус (в радианах)
sqr(x)	Возведение в квадрат
sqrt(x)	Извлечение квадратного корня

Операторы ввода-вывода

Вызов: read(v1,v2,...,vn) процедура	
Параметры: var v1,v2,...,vn:integer или real или char или string	
Действие:	Если vi имеет тип - integer или real, считывается одно число соответствующего формата и значение его присваивается переменной vi.Знаки пробела или перевода строки перед числом игнорируются. - char,считывается один символ и присваивается переменной vi. - string, при длине n строковой переменной vi считывается максимум n символов.
Вызов:	readln(v1,v2,...,vn) процедура
Параметры:	как для read
Действие:	как для read с последующим переходом на начало новой строки.

Операторы ввода-вывода

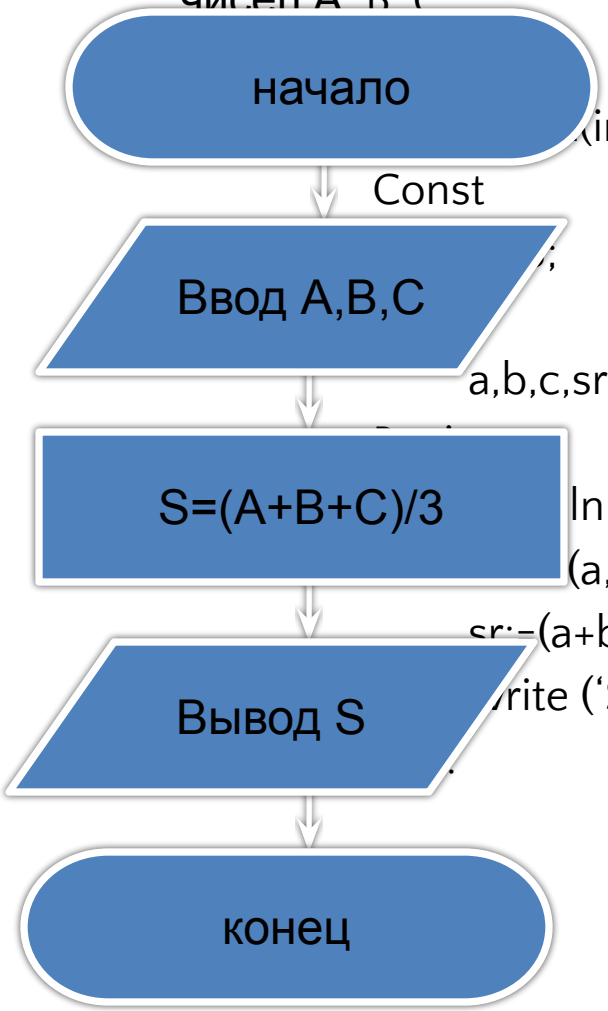
Вызов:	write (p1,p2,...,pn) процедура
Параметры:	p1,p2,...,pn:integer или real или boolean или char или string
Действие:	Выдается на экран значение pi в стандартной форме
Вызов:	writeln(p1,p2,...,pn) процедура
Параметры:	как для write
Действие:	Как для write. В заключение выполняется переход на начало новой строки (writeln = write line)

Формат, отличный от стандартного, можно выбрать следующим образом:

pi:d	d - выражение типа integer, задающее ширину поля данных, в которое должно быть записано значение pi (i=1,...,n) с выравниванием по правому краю.
pi:d:s	pi (i=1,...,n) имеет тип real. d используется также, как это было только что описано выше. s - выражение типа integer, задающее число знаков после десятичной точки (но тогда без экспоненты!).

Пример реализации линейного алгоритма на языке Pascal

Задача . Составить алгоритм нахождения среднего арифметического трех чисел A, B, C




```
(input, output);
Const
;
a,b,c,sr :real;
ln (‘Введите A, B и C’);
(a, b, c);
sr:=(a+b+c)/N;
rite (‘Sr=‘,sr)
```