

---

# ЯЗЫК ПРОГРАММИРОВАНИЯ PASCAL

- 
- *Как и естественные языки, каждый язык программирования имеет свой стиль и свои правила.*
  - **Синтаксис** языка программирования – это набор правил, которые определяют способы построения правильных программ из символов алфавита.

# АЛФАВИТ ЯЗЫКА ПАСКАЛЬ

---

26 латинских строчных и 26 латинских прописных букв:

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  
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

подчеркивание \_

10 цифр:

0 1 2 3 4 5 6 7 8 9

знаки операций:

+ - \* / = <> < > <= >= := @

ограничители (разделители):

. , ' ( ) [ ] ( . ) { } ( \* \* ) .. : ;

спецификаторы:

^ # \$

# УПРОЩЕННАЯ СТРУКТУРА ПРОГРАММЫ

*{1. заголовок программы}*  
program Имя\_Программы;

*{3. Раздел описаний}*  
const Описания\_Констант;  
var Описания\_Переменных;

*{4. Раздел операторов}*  
begin  
    Операторы программы  
end.

- 
- Команды языка программирования называются операторами
  - Разделителем операторов в Паскале является ; (точка с запятой)

***КОММЕНТАРИЙ*** ПРЕДСТАВЛЯЕТ СОБОЙ  
ФРАГМЕНТ ТЕКСТА ПРОГРАММЫ,  
ОГРАНИЧЕННЫЙ СИМВОЛАМИ `{}`.  
КОММЕНТАРИИ В ПРОГРАММЕ  
ВЫПОЛНЯЮТ ИНФОРМАЦИОННУЮ  
ФУНКЦИЮ.

*{Моя первая программа }*

program first;

begin

    writeln('Hello, World!')

end.

# ПРИМЕР: ПЕРИМЕТР ПРЯМОУГОЛЬНИКА

```
program perimetr;           {заголовок программы}
var a,b:integer;           {объявление переменных}
    P:integer;
Begin                       {начало программы}
    a:=12;   {присваиваем переменной a значение 12}
    b:=7;    {присваиваем переменной b значение 7}
    P:=2*(a+b); {значение выражения присваиваем P}
    write('P = ',P); {выводим на экран значение P}
end.                  {конец программы}
```

## ***ПРОГРАММА В СВОЕЙ РАБОТЕ ИМЕЕТ ДЕЛО С ДАННЫМИ.***

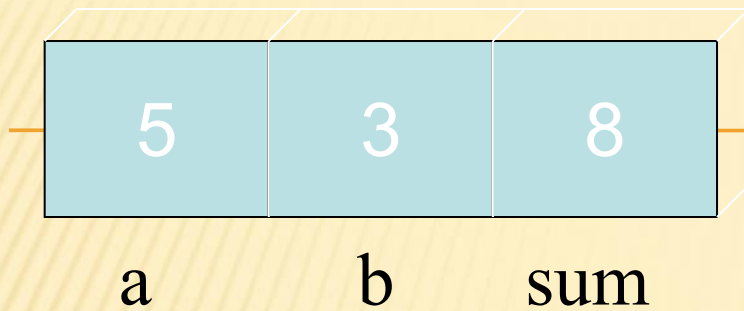
НЕКОТОРЫЕ ДАННЫЕ УСТАНОВЛИВАЮТСЯ ЕЩЕ ДО ТОГО, КАК ПРОГРАММА НАЧНЕТ ВЫПОЛНЯТЬСЯ, А ПОСЛЕ ЕЕ ЗАПУСКА СОХРАНЯЮТ СВОИ ЗНАЧЕНИЯ НЕИЗМЕННЫМИ НА ВСЕМ ПРОТЯЖЕНИИ РАБОТЫ ПРОГРАММЫ. ЭТО ***КОНСТАНТЫ.***

ДРУГИЕ ДАННЫЕ МОГУТ ИЗМЕНЯТЬСЯ ВО ВРЕМЯ ВЫПОЛНЕНИЯ ПРОГРАММЫ. ОНИ НАЗЫВАЮТСЯ ***ПЕРЕМЕННЫМИ.***



- ✓ Различие между *переменной* и *константой* довольно очевидно: во время выполнения программы значение переменной может быть изменено, а значение константы нет.





Под переменной мы будем понимать ячейку («коробку»), куда компьютер может записывать («складывать») данные.

Под ячейкой мы в действительности подразумеваем «кусочек памяти» в котором хранится информация.

Чтобы воспользоваться информацией, хранящейся в ячейке, нужно, чтобы каждая ячейка имела свое *имя* или, как часто говорят, - *идентификатор*.

# ИДЕНТИФИКАТОРЫ.

---

- ✓ *Именами (идентификаторами)* называют элементы языка - константы, метки, типы, переменные, процедуры, функции, модули, объекты.
- ✓ Идентификатором является последовательность букв, цифр и знаков подчеркивания, которая начинается с буквы или символа подчеркивания и не содержит пробелов.

- ✓ Имя может содержать произвольное количество символов, но значащими являются 63 символа.
- ✓ Не разрешается в языке ПАСКАЛЬ использовать в качестве имен *служебные слова* и *стандартные имена*, которыми названы стандартные константы, типы, процедуры, функции и файлы.
- ✓ Примеры имен языка ПАСКАЛЬ:  
A b12 r1m SIGMA gamma I80\_86

---

В ПАСКАЛЕ РАЗНИЦА МЕЖДУ СТРОЧНЫМИ И  
ПРОПИСНЫМИ БУКВАМИ ИГНОРИРУЕТСЯ

- 
- ▣ *Служебное слово* – это слово, которое в языке ПАСКАЛЬ имеет определенное смысловое значение, которое не может быть изменено. Иногда его называют **ключевым словом**.

# СЛУЖЕБНЫЕ (ЗАРЕЗЕРВИРОВАННЫЕ) СЛОВА:

ABSOLUTE EXPORTS LIBRARY SET  
ASSEMBLER EXTERNAL MOD SHL  
AND FAR NAME SHR  
ARRAY FILE NIL STRING  
ASM FOR NEAR THEN  
ASSEMBLER FORWARD NOT TO  
BEGIN FUNCTION OBJECT TYPE  
CASE GOTO OF UNIT  
CONST IF OR UNTIL  
CONSTRUCTOR IMPLEMENTATION PACKED USES  
DESTRUCTOR IN PRIVATE VAR  
DIV INDEX PROCEDURE VIRTUAL  
DO INHERITED PROGRAM WHILE  
DOWNTO INLINE PUBLIC WITH  
ELSE INTERFACE RECORD XOR  
END INTERRUPT REPEAT  
EXPORT LABEL RESIDENT

---

*Пробелы* нельзя использовать внутри  
сдвоенных символов и зарезервированных  
слов.

*Существуют имена которые называются  
стандартными.*

sin cos real true

В отличие от служебных слов смысл  
стандартных имен *может быть*  
*переопределен* программистом.



# Объявление переменных

---

- ✓ Это указание компилятору, сколько памяти необходимо зарезервировать для переменных нашей программы.
- ✓ В откомпилированной программе для всех переменных отведено место в памяти, и всем переменным присвоены нулевые значения.
- Все переменные, используемые в программе необходимо объявить в разделе описания переменных после зарезервированного слова `var`.

---

ДАННЫЕ БЫВАЮТ РАЗЛИЧНЫХ  
ТИПОВ...

# ТИПЫ ДАННЫХ

---

Имена *стандартных* типов являются predetermined идентификаторами и действуют в любой точке программы. Они описаны в стандартном модуле System, который по умолчанию подключается в список используемых модулей.

*Пользовательские* типы – это дополнительные типы (простые и структурированные) описанные пользователем.

# СТАНДАРТНЫЕ ТИПЫ ДАННЫХ

1. группа целых типов (Shortint, Integer, Longint, Byte, Word);
2. группа действительных типов (Single, Real, Double, Extended, Comp);
3. логические (булевские типы) (Boolean, ByteBool, WordBool, LongBool);
4. символьный (Char);
5. строковый (String);
6. указательный (Pointer);
7. текстовый тип (Text).

# ГРУППА ЦЕЛЫХ ТИПОВ

Название типа	Тип	диапазон значений	требуемая память
Короткое целое со знаком	Shortint	-128 .. 127	1 байт
Целое со знаком	<b>Integer</b>	-32768 .. 32767	2 байта
Длинное целое со знаком	Longint	-2147483648 .. 2147483647	4 байта
Короткое целое без знака	Byte	0 .. 255	1 байт
Целое без знака	Word	0 .. 65535	2 байта

# Объявление переменных целого типа

```
Var  b : byte;  
     summa, count : integer;
```

## *Испозование*

```
summa:= -365;
```

**Числа** в языке ПАСКАЛЬ обычно записываются в десятичной системе счисления.

---

Положительный знак числа может быть опущен.

**Целые** числа записываются в форме без десятичной точки, например:

**217   -45   8954   +483**

---

# *Группа вещественных типов*

определяет те данные, которые реализуются подмножеством действительных чисел.

```
Var A: real;
```

```
begin
```

```
  ...
```

```
  A:=0.65;
```

```
  ...
```

```
End.
```



Название типа	Тип	Диапазон значений	Количество цифр мантиссы	Размер (байт)
Вещественное число одинарной точности	<b>Real</b>	$2.9e^{-39} .. 1.7e^{+38}$	11	6
Вещественное число	<b>Single</b>	$1.5e^{-45} .. 3.4e^{+38}$	7	4
Вещественное число двойной точности	<b>Double</b>	$5.0e^{-324} .. 1.7e^{+308}$	15	8
Вещественное число повышенной точности	<b>Extended</b>	$3.4e^{-4932} .. 1.1e^{+4932}$	19	10
Целое число в формате вещественного	<b>Comp</b>	$-9.2e^{+18} .. 9.2e^{+18}$	19	8

***Действительные числа*** записываются в форме с десятичной **точкой** или в форме с использованием десятичного порядка, который изображается буквой **E**:

**28.6      0.65      -0.018      4.0**

**5E12      -1.72E9      73.1E-16**

*Строка символов* представляет собой последовательность символов из набора символов кода ASCII, заключенную в одиночные кавычки.

*Строки* в языке ПАСКАЛЬ - это последовательность символов, записанная между апострофами. Если в строке в качестве содержательного символа необходимо употребить сам апостроф, то следует записать два апострофа.

Примеры строк:

'СТРОКА' 'STRING' 'АД''ЮТАНТ'

# СТАНДАРТНЫЙ ВВОД И ВЫВОД

- ✓ Осуществляется встроенными процедурами
- ✓ *Read*(список переменных); – чтение значений, введенных с клавиатуры
- ✓ *Readln*(список переменных); - ln в конце означает переход на новую строку
- ✓ *Write*(список переменных); - вывод на экран
- ✓ *Writeln*(список переменных);

# ПРИМЕР

---

```
Program primer;  
Var a,b,summa:integer;  
Begin  
    write('Введите 2 целых числа: ');  
    readln(a,b);  
    Summa:=a+b;  
    write('Сумма = ',summa);  
End.
```

- 
- ✓ Вводить можно переменные *целых, вещественных, символьного и строкового* типов.
  - ✓ Допускается вывод значений *целых, вещественных, символьного, строкового и булевских* типов.

# ОПЕРАЦИИ И ВЫРАЖЕНИЯ

---

## Выражение

- в программировании служит для определения действий.
- Выражения состоят из операций и операндов.
- По количеству операндов выражения делятся на унарные (один операнд с символом операции:  $-5$ ,  $-(-6)$ , `not False`) и бинарные (два операнда, между которыми ставится символ операции:  $5+7$ ,  $(4-2)*5+10$ , *True or False* - истина или ложь).

# КЛАССИФИКАЦИЯ ОПЕРАЦИЙ

---

1. арифметические операции:
  - унарные: +, -
  - бинарные: +, -, \*, /, div, mod
2. операции отношения: =, <>, <, >, <=, >=
3. булевские (логические) операции:  
not (логическое отрицание), and (лог. И), or (лог. ИЛИ), xor (исключающее ИЛИ)
4. строковая операция (конкатенация) +



---

div – частное от деления

$$7 \text{ div } 3 = 2$$

*A:=10 div 3;*

mod – остаток от деления

$$7 \text{ mod } 3 = 1$$

*A:=25 mod 10;*

---

## ***ОПЕРАТОРЫ***

**ПРЕДНАЗНАЧЕНЫ ДЛЯ ОПИСАНИЯ  
ДЕЙСТВИЙ, КОТОРЫЕ БУДУТ  
ВЫПОЛНЯТЬСЯ ПРИ РЕАЛИЗАЦИИ  
АЛГОРИТМА.**

# ***ПРОСТЫЕ ОПЕРАТОРЫ НЕ СОДЕРЖАТ В СЕБЕ ДРУГИХ ОПЕРАТОРОВ.***

---

1. Оператор *присваивания* :=
2. Оператор *процедуры* состоит из имени, за которым в круглых скобках может располагаться список фактических параметров. *Swap(x, y).*
3. Оператор *перехода* *GoTo*

# ~~СТРУКТУРНЫЕ ОПЕРАТОРЫ~~ ВКЛЮЧАЮТ В СЕБЯ ДРУГИЕ ОПЕРАТОРЫ.

1. Составной оператор
2. Условные операторы (*if, case*)
3. Операторы цикла (*while, repeat, for*)
4. Оператор присоединения *with*