



Pascal

Pascal

Паскаль был разработан швейцарским ученым Никлаусом Виртом. Паскаль считается важнейшим инструментом для обучения методам структурного программирования и с 1983 г. введен в учебные курсы в школах для учащихся, которые специализируются в области информатики.

В дальнейшем язык Паскаль совершенствовался и приобрел новые свойства, отличные от авторского варианта.

- ❖ Язык Паскаль относительно прост в изучении, довольно ясен и логичен и, будучи первым изучаемым языком программирования, приучает к хорошему стилю.

- *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:

^ # \$

Структура программы

{1. заголовок программы}

program Имя_Программы;

{2. раздел указания используемых модулей}

uses Список_Используемых_Модулей;

{3. Раздел описаний}

label Описания_меток;

const Описания_Констант;

type Описания_Типов;

var Описания_Переменных;

procedure Описания_Процедур_и_функций;

function

exports Описания_Экспортируемых_Имен;

{4. Раздел операторов}

begin

 Операторы

end.

Упрощенная структура программы

{1. заголовок программы}

program Имя_Программы;

{2. раздел указания используемых модулей}

uses Список_Используемых_Модулей;

{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

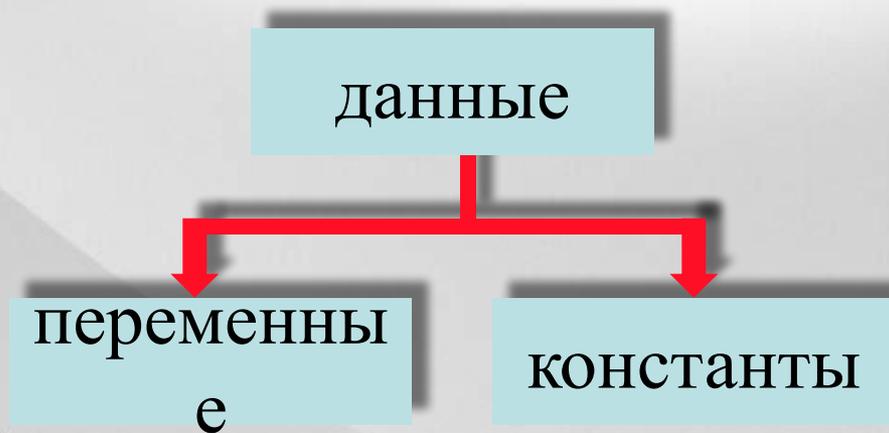
```
program perimetr;           {заголовок программы}
uses crt;                 {crt – необходим для очистки экрана}
var a,b:integer;         {объявление переменных}
    P:integer;
Begin                     {начало программы}
    clrscr;               {очистка экрана}
    a:=12; {присваиваем переменной 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 устанавливаются 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, поэтому имена **NaMe** и **name** одинаковы.

- ***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 | Integer | -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 | Real | $2.9e^{-39} .. 1.7e^{+38}$ | 11 | 6 |
| Вещественное число | Single | $1.5e^{-45} .. 3.4e^{+38}$ | 7 | 4 |
| Вещественное число двойной точности | Double | $5.0e^{-324} .. 1.7e^{+308}$ | 15 | 8 |
| Вещественное число повышенной точности | Extended | $3.4e^{-4932} .. 1.1e^{+4932}$ | 19 | 10 |
| Целое число в формате вещественного | Comp | $-9.2e^{+18} .. 9.2e^{+18}$ | 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

Булевскому типу данных соответствует идентификатор **Boolean**. Переменные булевского типа имеют размер **1 байт** и могут содержать значения **TRUE** или **FALSE**.

Значению FALSE соответствует 0, любое число отличное от нуля считается TRUE.

```
Var кнопка, flag : boolean;
```

```
Begin
```

```
    кнопка:=true;
```

Символьному типу соответствует стандартный идентификатор **Char**.

Переменные и константы символьного типа могут принимать значения из множества символов кода ASCII.

Объявление символьных переменных

```
Var simvol, bukva, z : char;
```

Строковому типу соответствует стандартный идентификатор **String**.

Var

S : String;

{строка от 0 до 255 символов}

S2: String[5];

{стока из 5-ти символов}

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'

- ✓ Символьный тип, а также целые и булевские типы относят к, так называемым, **порядковым** типам.
- ✓ Множество допустимых значений любого порядкового типа представляет собой упорядоченную последовательность, каждый элемент которой имеет свой порядковый номер (начиная с 0).

Функция Ord

возвращает порядковый номер этого значения в описании типа.

`Ord(2)=2, Ord('0')=48`

```
WriteLn(ord('e'));
```

```
WriteLn(ord('9'));
```

Таблица кодировки ASCII

- **ASCII** (*American Standard Code for Information Interchange*; произносится «áски») — компьютерная кодировка для представления латинского алфавита, арабских цифр, некоторых знаков пунктуации и управляющих символов.
- Ее ввел американский институт стандартизации ANSI.

| Код | Символ | Код | Символ | Код | Символ | Код | Символ |
|------------|---------------|------------|---------------|------------|---------------|------------|---------------|
| 32 | пробел | 56 | 8 | 80 | P | 104 | h |
| 33 | ! | 57 | 9 | 81 | Q | 105 | i |
| 34 | " | 58 | : | 82 | R | 106 | j |
| 35 | # | 59 | ; | 83 | S | 107 | k |
| 36 | \$ | 60 | < | 84 | T | 108 | l |
| 37 | % | 61 | = | 85 | U | 109 | m |
| 38 | & | 62 | > | 86 | V | 110 | n |
| 39 | ' | 63 | ? | 87 | W | 111 | o |
| 40 | (| 64 | @ | 88 | X | 112 | p |
| 41 |) | 65 | A | 89 | Y | 113 | q |
| 42 | * | 66 | B | 90 | Z | 114 | r |

| | | | | | | | |
|-----------|----------|-----------|----------|------------|----------|------------|----------|
| 43 | + | 67 | C | 91 | [| 115 | s |
| 44 | , | 68 | D | 92 | \ | 116 | t |
| 45 | - | 69 | E | 93 |] | 117 | u |
| 46 | . | 70 | F | 94 | ^ | 118 | v |
| 47 | / | 71 | G | 95 | _ | 119 | w |
| 48 | 0 | 72 | H | 96 | ` | 120 | x |
| 49 | 1 | 73 | I | 97 | a | 121 | y |
| 50 | 2 | 74 | J | 98 | b | 122 | z |
| 51 | 3 | 75 | K | 99 | c | 123 | { |
| 52 | 4 | 76 | L | 100 | d | 124 | |
| 53 | 5 | 77 | M | 101 | e | 125 | } |
| 54 | 6 | 78 | N | 102 | f | 126 | ~ |
| 55 | 7 | 79 | O | 103 | g | 127 | |

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;  
Uses crt;  
Var a,b,summa:integer;  
Begin  
  clrscr;  
  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*