«Не волнуйтесь, если что-то не работает. Если бы все работало, вас бы уволили»

Mosher's Law of Software Engineering

Лекция 2: Введение в Pascal ABC

Разработал: Павлов А.Н.

Сокращения

ОС – операционная система

ЯП – язык программирования

Содержание лекции

1. Язык Pascal и система программирования Pascal ABC
2. Алфавит языка и структура программы
3. Идентификаторы и зарезервированные слова
4. Константы, переменные, метки и типы
5. Типы данных, стандартные функции и выражения
6. Операнды и операции. Оператор присваивания
7. Комментарии к программе
8. Ввод/вывод данных
9. Первая программа

Язык Pascal и система программирования Pascal ABC

История языка Pascal

Язык был создан швейцарским ученым Никлаусом Виртом в 1968-69 годах.

Pascal – один из наиболее известных ЯП, используется для обучения программированию в старших классах и на первых курсах вузов, является базой для ряда других языков.

Особенностями языка являются строгая типизация и наличие средств процедурного программирования. Синтаксис языка интуитивно понятен даже при первом знакомстве.

В 1985 г. был создан диалект Object Pascal, поддерживающий объектно-ориентированное программирование.

Язык назван в честь французского математика, физика и философа Блеза Паскаля.

Система программирования

Для написания программы в принципе можно использовать обычный текстовый редактор (Блокнот), затем с помощью компилятора перевести ее в машинный код, т.е. получить исполняемую программу. Но проще и удобней использовать специально разработанную систему программирования.

Система программирования – это система для разработки программ на конкретном ЯП.

Популярные системы программирования:

ЯП	Система программирования
Pascal	Turbo Pascal, Borland Pascal, Borland Delphi
Basic	Quick Basic, Turbo Basic, Visual Basic
C	Turbo C

Система программирования

Современные системы программирования обычно предоставляют пользователям мощные и удобные средства разработки программ.

В них входят:

- **компилятор или интерпретатор**
- **интегрированная среда разработки**
- **средства создания и редактирования текстов программ**
- **обширные библиотеки стандартных программ и функций**
- **средства отладки, помогающие устранять ошибки в программе**
- **«дружественная» к пользователю диалоговая среда;**
- **многооконный режим работы**
- **мощные графические библиотеки**
- **встроенный ассемблер**
- **встроенная справочная служба**

Система программирования Pascal ABC

Раньше для программирования на ЯП Pascal широко использовалась система программирования Turbo (Borland) Pascal. Существовали различные версии – от Turbo Pascal 1.0 (1983) до Borland Pascal 7.1 (1994).

Система обладала многими достоинствами (удобной средой разработки, высокая скорость компиляции и выполнения программ, возможность использования вставок на языке ассемблера), но имела также ряд недостатков, основной из которых – работа в устаревшей ОС MS DOS.

Поэтому в 2003 году в Южном Федеральном Университете (г. Ростов-на-Дону) была создана учебная среда программирования PascalABC.NET. Система представляет собой интегрированную оболочку со встроенным интерпретатором языка Паскаль. Она стала удачной заменой системе Turbo Pascal.

Система программирования PascalABC

Система PascalABC является мультипарадигменной, т.е. в ней можно программировать в процедурном, объектно-ориентированном и функциональном стилях.

В состав системы входят следующие стандартные модули:

- GraphABC – растровая графика
- ABCObjects – векторная графика
- FormsABC – создание простых оконных приложений
- Arrays – работа с одно- и двумерных динамическими массивами
- Collections – содержит упрощенные классы коллекций
- исполнители Робот и Чертёжник (школьная информатика)

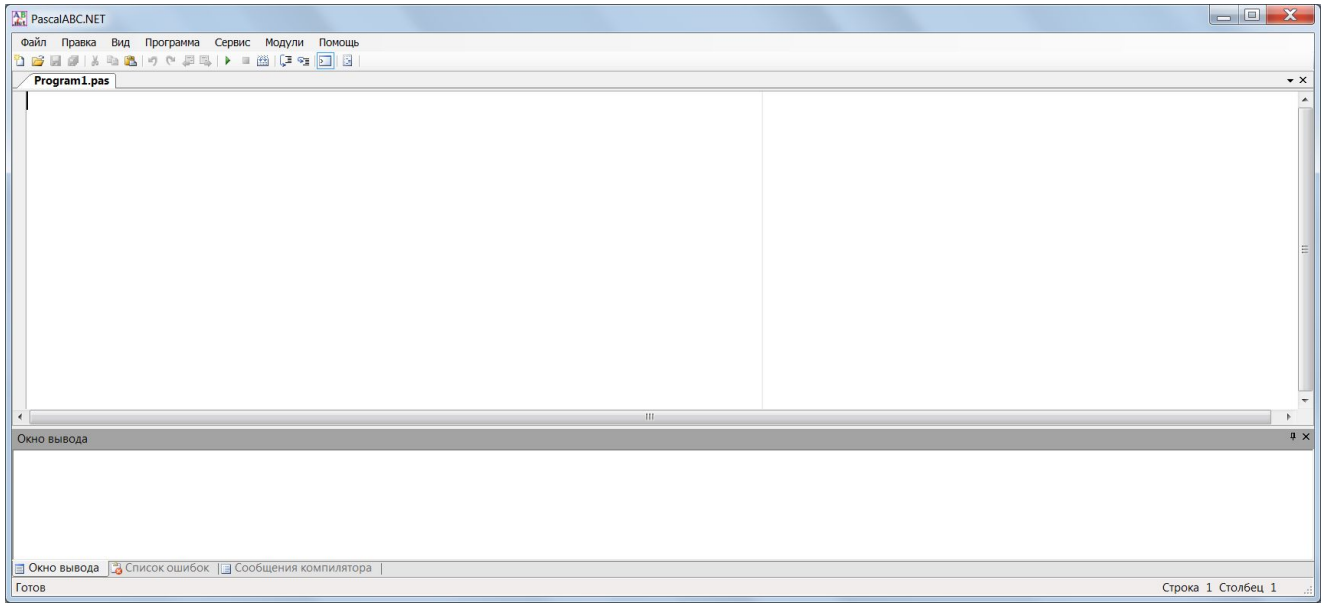
Установка и запуск системы PascalABC

Скачать программу PascalABC можно, перейдя по ссылке:

http://pascalabc.net/ssyilki-dlya-skachivaniya

После установки ярлык программы появляется на Рабочем столе и в меню «Пуск».

Дважды кликнув по этому ярлыку, запускаем программу:



Алфавит языка и структура программы

Алфавит языка Pascal

Алфавит ЯП – это совокупность всех допустимых символов, которые можно использовать в этом языке.

Алфавит языка Pascal включает в себя следующие символы:

- прописные и строчные буквы латинского алфавита от A до z, а также символ подчеркивания _, который тоже считается буквой.
  Прописные и строчные буквы равнозначны!
- арабские цифры 0 1 2 3 4 5 6 7 8 9
- специальные одиночные знаки + - * / = < > . , : ; ^ $ # @
- специальные парные знаки ( ) [ ] { } ‘ ‘
- составные знаки <= >= <> .. (* *) (..)

Также используются буквы русского алфавита, но только при вводе/выводе текста, заключенного в апострофы (‘ ‘) или в комментариях к программе.

Структура программы на языке Pascal

```
program имя программы;{заголовок программы}

uses   {раздел подключения модулей}

Label  {раздел описания меток}
Const  {раздел описания констант}
Type   {раздел описания типов}
Var    {раздел описания переменных}
Function …;   {раздел описания функций}
Procedure …;  {раздел описания процедур}

BEGIN
…
END.
```

Блок описаний

Раздел операторов

Структура программы на языке Pascal

Первая строка – это **заголовок программы**, не обязателен.

Далее следует **раздел подключения модулей**, где указываются через запятую подключаемые к программе модули. Также может отсутствовать.

Далее идет **блок описаний**, состоящий из пяти разделов:

1. описание меток
2. описание констант
3. описание типов
4. описание переменных
5. описание процедур и функций

В этом списке только раздел описания переменных является обязательным, остальные могут отсутствовать.

Структура программы на языке Pascal

Далее со служебного слова **BEGIN** начинается раздел операторов, которые отделяются друг от друга точкой с запятой «;».

Конструкция **begin** … **end** называется операторными скобками, операторы, находящиеся внутри этой конструкции, считаются одним составным оператором.

Вся программа завершается словом END с точкой.

Идентификаторы и служебные слова

Идентификаторы

Идентификаторы – это имена переменных, констант, меток, типов, модулей, процедур и функций.

Имена задает разработчик программы.

На идентификаторы накладываются ограничения:

- нельзя использовать служебные слова
- имя должно начинаться с буквы и может содержать латинские буквы, цифры и знаки подчеркивания

Пример:

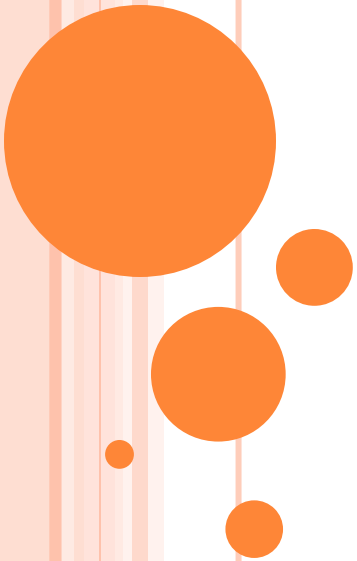
`a1, b_2, k123, _d` – идентификаторы

`1a, и2, @ru, integer, var` – не идентификаторы

Желательно выбирать имена, несущие смысловую нагрузку, например, **`result`**, **`summa`**, **`price`**. Это делает программу проще для понимания.

Служебные слова языка Pascal

```
and
array as
auto
begin
case
class
const
constructor
destructor
div
do
downto
else
end
event
except
file
finalization
finally
for
```



Константы, переменные, метки и типы

Константы

Константа (постоянная) – это величина, значение которой не изменяется при исполнении программы.

Раздел описания констант начинается со служебного слова **const**, после которого следуют элементы описания:

```
имя константы = значение;
```

или

```
имя константы : тип = значение;
```

Пример:

```
const
  Pi = 3.14;
  Number = 10;
  Name = 'Victor';
  Cifra = ['0'..'9'];
  Mass: array [1..5] of integer = (1,3,5,7,9);
```

Компьютер "знает", чему равны константы **e** и **π**.

Переменные

Переменная – одно из главных понятий в программировании.

Переменная – это величина, значение которой меняется при исполнении программы.

После объявления переменных программа выделяет определенное (в зависимости от типа данных) количество ячеек в памяти для хранения этих переменных.

Имена переменных присваиваются этим ячейкам, в которых затем хранятся значения переменных.

Храниться они могут или до конца выполнения программы или до тех пор, пока переменной не присвоится другое значение.

Имя переменной не изменяется до конца программы, а значение переменной может меняться.

В ЯП Pascal обязательное объявление переменных защищает программы от ошибок и повышает их надежность.

Переменные

Раздел описания переменных начинается со служебного слова **var**, после которого следует описание:

Пример:

```
var
  a,b: integer;
  c,d: real;
  m,n: byte;
  s,s1: string;
  ch,ch1: char;
  f: boolean;
```

Переменные

Переменные могут описываться как в начале программы, так и непосредственно внутри любого блока **begin** … **end**. Внутриблочные описания переменных имеют тот же вид, что и в разделе описаний.

Пример:

```
begin
  var a1,a2,a3: integer;
end.
```

Кроме того, переменные – параметры цикла могут описываться в заголовке оператора **for**.

Метки

Метки используются в инструкциях безусловного перехода **goto** для передачи управления помеченным операторам.

Перед употреблением метки должны быть описаны в начале программы в блоке описаний. Раздел описания меток начинается с зарезервированного слова **Label**, после которого следует список меток, перечисляемых через запятую.

В качестве меток могут использоваться идентификаторы и положительные целые числа:

Пример:

```
label a1,l2,777777;
```

Типы

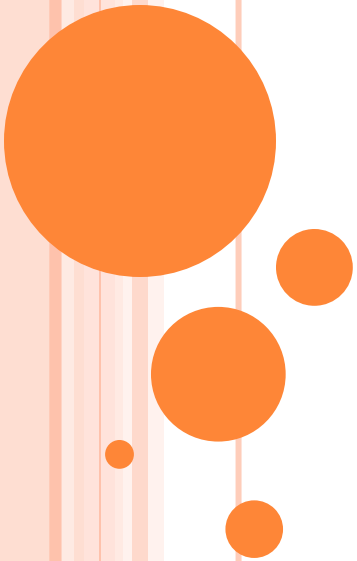
Раздел описания типов начинается со служебного слова **type**, после которого следуют строки вида

```
имя типа = тип;
```

Пример:

```
type
  arr10 = array [1..10] of integer;
  myint = integer;
  pinteger = ^integer;
  IntFunc = function(x: integer): integer;
```

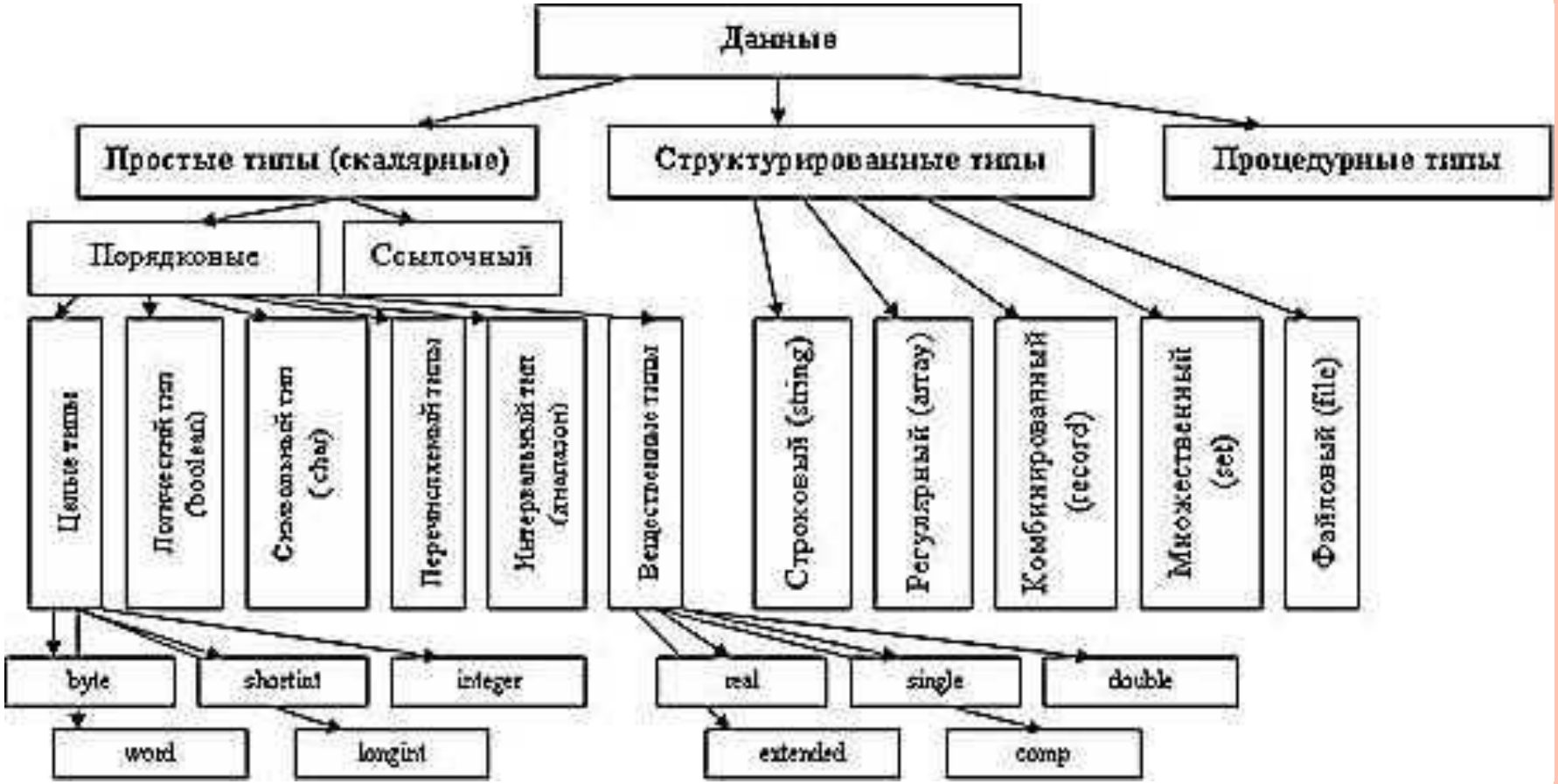
Обычно описание используется для составных типов (статические массивы, процедурные переменные, записи, классы) чтобы дать имя сложному типу.



Типы данных, стандартные функции и выражения

Стандартные типы данных

Тип данных определяет возможные значения констант, переменных, форму представления, а также возможные операции над данными этого типа.



Типы данных

- **простые**
  - целые
  - логические
  - символьные
  - перечислимые
  - диапазонные
  - вещественные

  } порядковые
- **структурированные**
  - массивы
  - записи
  - множества
  - файлы
- **строковые**
- **указатели**
- **процедурные типы**
- **классы**

Целый тип

Это целые положительные (и отрицательные) числа.

Имя типа	Значение	Размер, байт	Тип
BYTE	0..255	1	Числовой беззнаковый целый
word	0..65535	2	Числовой беззнаковый целый
integer	-2 147 483 648.. 2 147 483 647	4	Числовой знаковый целый

Целый тип

Над целыми числами можно выполнять следующие арифметические операции:

- сложение **+**
- вычитание **–**
- умножение *****
- деление с отбрасыванием дробной части **DIV**
- получение целого остатка от деления **MOD**

В результате также получается *целое* число.

Целый тип

Пример:

Даны переменные **A**, **B** и **N** целого типа **integer**.

```
A = 25; B = 2; N = 17;
```

Операция	Результат
A + 50	75
B – A	-23
B * N	-34
A DIV B	12
A MOD B	1

Вещественный тип

Данные вещественного (действительного) типа могут быть представлены в двух формах:

1. с фиксированной точкой
2. с плавающей точкой.

Пример:

Числа с фиксированной точкой: `35.62`; `-12.05`.

Числа с плавающей точкой:

Математическая запись	Запись на языке Pascal
$2 \cdot 10^{-4}$	`2E-4`
$0{,}32 \cdot 10^{4}$	`0.32E+4`
$-12{,}75 \cdot 10^{11}$	`-12,75E11`

Вещественный тип

Имя типа	Значение	Размер, байт	Количество значащих цифр
real (double)	$-1.8 \cdot 10^{308} .. 1.8 \cdot 10^{308}$	8	16
single	$-3.4 \cdot 10^{38} .. 3.4 \cdot 10^{38}$	4	8
decimal	-79228162514264337593543950335 .. 79228162514264337593543950335	16	29

Логический тип

Данные типа **boolean** занимают 1 байт, могут принимать одно из двух значений:

- **True** (истина, 1)
- **False** (ложь, 0)

Над логическими данными выполняются следующие операции:

- **OR** – логическое сложение «или»
- **AND** – логическое умножение «и»
- **NOT** – логическое отрицание «не»

Операции **OR** и **AND** являются бинарными, т.е. выполняются над двумя величинами, операция **NOT** – унарная, над одной величиной.

Логический тип

Результаты операций над логическими данными

A	B	NOT A	A OR B	A AND B
True	True	False	True	True
True	False	False	True	False
False	True	True	True	False
False	False	True	False	False

Каждая логическая операция имеет ранг старшинства.

Самой старшей является операция отрицания. Далее в порядке убывания старшинства следуют умножение и сложение.

Символьный тип

Символьная (литерная) величина **CHAR** – это любой символ языка, заключенный в апострофы.

Символьная величина занимает 1 байт.

Пример: **'B'**, **':'**, **'+'**, **'5'** – символьные величины.

Для задания апострофа вводят **''''**. При этом внешние апострофы не входят в символьную величину, а являются ее признаком.

Все символы языка Pascal упорядочены, к ним можно применять операции сравнения **<**, **>**, **=**, **<=**, **>=**.

Пример: **'A' < 'B'= True**, т.к. сравниваются их порядковые номера, а они равны 66 и 67 соответственно.

Строковый тип

Строки имеют тип **string**, состоят из набора последовательно расположенных символов **char** и используются для представления текста.

По умолчанию под переменную типа **string** отводится 256 байт, при этом в нулевом байте хранится длина строки. Т.е. строки состоят не более чем из 255 символов.
Можно явно указать количество символов в строке.

Пример описания строковых данных:

```
s: string;  //Наибольшая длина строки 255 символов
s: string[50]; //Наибольшая длина строки 50 символов
```

К отдельному символу строки можно обратиться по его номеру.

Перечислимый тип

Задается последовательным перечислением всех значений, которые может принимать переменная этого типа.

Пример описания данных перечислимого типа:

```
Month: (May, June, July, August);
```

Типы данных

Все рассмотренные типы данных, кроме вещественного, являются порядковыми, т.е. упорядоченными.

Для обращения к отдельному элементу порядкового типа следует указать его индекс.

Индекс – это порядковый номер элемента в последовательности.

Остальные типы данных будут рассмотрены позже

Стандартные функции

Правила записи стандартных функций:

1. Имя функции записывается латинскими буквами
2. Аргумент функции записывается в круглых скобках после имени функции
3. Аргументом функции может быть константа, переменная или арифметическое выражение

Основные стандартные функции

Функция	Назначение	Тип аргумента	Тип функции
ABS(X)	Вычисление модуля X	real integer	real integer
SQR(X)	Вычисление квадрата X	real integer	real integer
SIN(X)	Вычисление синуса X	real integer	real
COS(X)	Вычисление косинуса X	real integer	real
ARCTAN(X)	Вычисление арктангенса X	real integer	real
EXP(X)	Вычисление экспоненты X	real integer	real
EXP10(X)	10^X	real integer	real
LN(X)	ln X	real integer	real
LOG10(X)	Десятичный логарифм lg X	real integer	real

Основные стандартные функции

Функция	Назначение	Тип аргумента	Тип функции
SQRT(X)	Квадратный корень из X	real integer	real
TRUNC(X)	Целая часть от X	real integer	integer
ROUND(X)	Округление X до ближайшего целого	real integer	integer
ODD(X)	TRUE, если X – нечетное FALSE, если X – четное	integer	boolean
POWER(X,Y)	X в степени Y	real	real
SUCC(X)	а) X + 1 б) следующий символ после X в упорядоченном множестве символов	integer char	integer char
PRED(X)	а) X + 1 б) предыдущий символ по отношению к X в упорядоченном множестве символов	integer char	integer char

Арифметические выражения

В языке Pascal есть **арифметические** и **логические** выражения.

В состав **арифметических** выражений могут входить константы, переменные, стандартные функции, круглые скобки и знаки операций.

Правила записи арифметических выражений:

1. Выражение записывается в одну строку
2. Использовать можно только круглые скобки. Квадратные и фигурные скобки не применяют, т.к. они имеют особое значение. Число открывающихся скобок должно быть равно числу закрывающихся.
3. Нельзя записывать последовательно два знака арифметических операций, их надо разделить круглой скобкой
4. Вычисление выражений производится *по приоритету операций*

Арифметические выражения

Приоритет арифметических операций:

1. Первыми выполняются операции, имеющие высший приоритет.
2. Операции с одинаковым приоритетом выполняются слева направо.
3. Для изменения порядка операций используют круглые скобки. В первую очередь выполняются операции в них. Если выражения в скобках вложены друг в друга, то вычисление начинается в самых внутренних скобках, а далее переходит во внешние скобки.
4. Если аргумент функции задан в виде выражения, то сначала определяется значение этого выражения, а потом значение функции.

Высший	Средний	Низший
Умножение Деление	DIV MOD	Сложение Вычитание

Приоритет ↓

Логические выражения

Строятся из:

- **логических данных**
- **логических операций**
- **операций сравнения** – могут включать в себя арифметические, логические выражения и символьные данные.

Результат логического выражения – это **`True`** или **`False`**.

Приоритет логических операций:

Высший	Средний	Низший
Арифметические операции	Операции сравнения	Логические операции

Приоритет ↓

При наличии скобок сначала выполняются действия в скобках (в первую очередь во внутренних), а затем вне скобок.

Логические выражения

Операции сравнения:

Используются для проверки отношений между переменными:

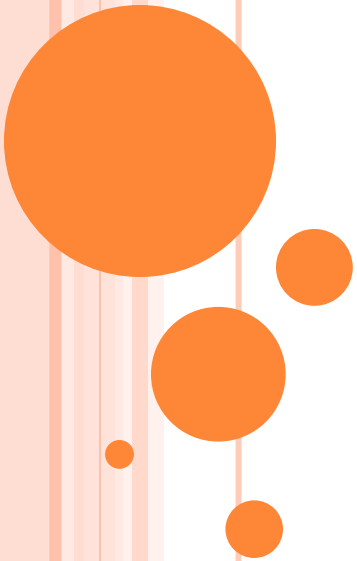
a < b, **c >= d**, **x = y** и т.д.

Над данными типа **real**, **integer**, **char**, **string** можно выполнять следующие операции сравнения:

= равно
<> не равно
> больше
< меньше
>= больше или равно
<= меньше или равно

Сравнивать можно только данные одного типа!

Исключение есть лишь для типов **real** и **integer**, которые можно сравнивать друг с другом.



Операнды и операции. Оператор присваивания

Операнды и операции

Операнды – это данные, к которым применяются операции.
Операции – это действия над данными (операндами).

Строковые операции:

Основной операцией над данными типа **`string`** и **`char`**, кроме операций сравнения **`<`**, **`>`**, **`=`**, **`>=`**, **`<=`** является операция конкатенации (слияния). Результат имеет тип **`string`**.

Пример:

```
'a' + 'b' = 'ab'
'пол' + 'нота' = 'полнота'
```

Строки могут содержать максимум 255 символов, поэтому если при слиянии получается больше символов, то это приведет к ошибке.

Операция @:

Применяется к переменной, возвращает ее адрес.

Оператор присваивания

Как и во всех ЯП в Pascal есть оператор присваивания **:=**, служит для задания значения переменной.

Если переменная уже имела какое-то значение, то оно стирается, и переменной присваивается новое значение.

Синтаксис оператора:

```
Переменная := выражение
```

Переменная в правой части и выражение в правой части должны быть одного типа.

Из этого правила есть исключение: Переменной типа **real** можно присваивать выражение типа **integer**. При этом значение переменной станет вещественным.

В некоторых ЯП символом присваивания является знак равенства **=**. Чтобы не путать его с оператором сравнения, в Pascal введено обозначение **:=** .

Оператор присваивания

Пример 1:

```
a:=10;
b:=5;
a:=a+b;
```

Вначале **a** равно **10**, потом **a** равно **15.**

Пример 2:

```
T:=527.47;
ST:='Pascal';
C:=2*K-SIN(PI/4-X);
```

Здесь **T** и **C** имеют действительные значения, должны быть предварительно описаны в разделе переменных как данные типа **real**.

Переменная **ST** должна иметь строковый тип **string**.

Комментарии к программе

Комментарии к программе

В программе может быть текст, написанный разработчиком для пояснения к программе. Этот текст называется комментарием. Даже опытные программисты считают необходимыми комментарии в своих программах.

Комментарии не воспринимаются компьютером и не обрабатываются программой.

Пример:

```
{ Это комментарий }
{* Это тоже комментарий *}
// Это – тоже комментарий
```

Ввод/вывод данных

Ввод данных

Выполняется операторами **read** и **readln** (read line – прочти строку).

```
read(A1,A2);
readln;
readln(A1,A2);
```

где **A1**, **A2** – переменные, которым последовательно присваиваются вводимые с клавиатуры значения.

При выполнении оператора **read** программа останавливается и ждет ввода значений переменных. Если в операторе указано две и более переменных, то при наборе они печатаются через пробел.

При выполнении оператора ввода без параметров **readln** выполняется переход на новую строку.

При выполнении оператора **readln(A1,A2)** вводятся значения всех переменных, а затем выполняется переход на новую строку.

Вывод данных

Выполняется операторами **write** и **writeln** (write line – напиши строку).

```
write(A1,A2); {Вывод значений A1 и A2 в одну строку}
writeln; {Переход на новую строку}
writeln(A1,A2);{Вывод A1 и A2 и переход на новую строку}
```

Ввод числовых данных с форматом

```
write(A:L:D);
```

Для **целых** чисел указывают только **L**, которое показывает количество цифр в выводимом числе, включая знак.

Для **вещественных** чисел указывается **L** и **D**, где **L** показывает общее количество в выводимом числе вместе со знаком и десятичной точкой, а **D** – количество цифр в дробной части.

Вывод данных

Пример 1:

```
a:=10; b:=28.63;
write(a:6);
write(b:6:2);
```

Результат _ _ _ _ 10

Результат _28.63

Также можно выводить пояснительный текст, используя апострофы

Пример 2:

```
writeln('Значение B = ',B);
write('Ведите значения X и Y:');
```

Первая программа

Первая программа

1. Словесная постановка задачи:

Разработать программу для вычисления суммы и частного от двух введенных чисел.

2. Математическая постановка задачи:

Summa := A + B Chastnoe := A / B

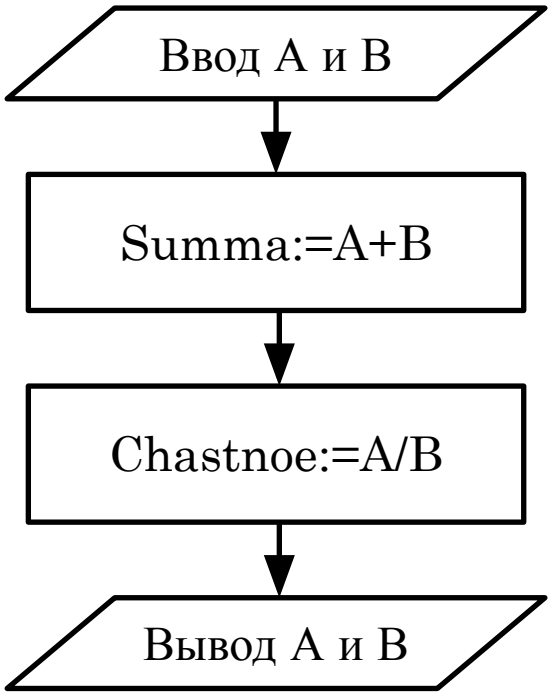
3. Разработка алгоритма и его блок-схемы:

Словесное описание алгоритма:

1. Ввод чисел A и B
2. Вычисление суммы
3. Вычисление частного
4. Вывод результатов

Первая программа

Блок-схема алгоритма:



- Словесная постановка задачи
- Математическая постановка задачи
- Разработка алгоритма и его блок-схемы
- Кодирование
- Контрольное тестирование и отладка программы
- Анализ результатов

Первая программа

4. Кодирование:

PascalABC.NET

Файл Правка Вид Программа Сервис Модули Помощь

•FirstProgram.pas*

```
1  //Программа вычисляет сумму и частное двух введенных чисел
2  program My_First_Program;//Название программы
3  {Объявление переменных}
4  var
5    A, B, Summa, Chastnoe: real;//A и B - вводимые числа, Summa - сумма,
6                               //Chastnoe - частное (результат деления)
7  begin
8    {Ввод исходных данных для расчета}
9    write('Введите первое число: ');  //Ввод A
10   readln(A);
11   write('Введите второе число: ');  //Ввод B
12   readln(B);
13   {Выполнение расчетов}
14   Summa := A + B;        //Вычисление суммы
15   Chastnoe := A / B;     //Вычисление частного
16   {Вывод результатов расчета}
17   writeln('Сумма равна: ', Summa:7:3);  //Вывод суммы
18   writeln('Частное равно: ', Chastnoe:7:3); //Вывод частного
19 end.
```

Для форматирования кода используется кнопка 

- Словесная постановка задачи
- Математическая постановка задачи
- Разработка алгоритма и его блок-схемы
- Кодирование
- Контрольное тестирование и отладка программы
- Анализ результатов

Первая программа

5. Контрольное тестирование и отладка программы:

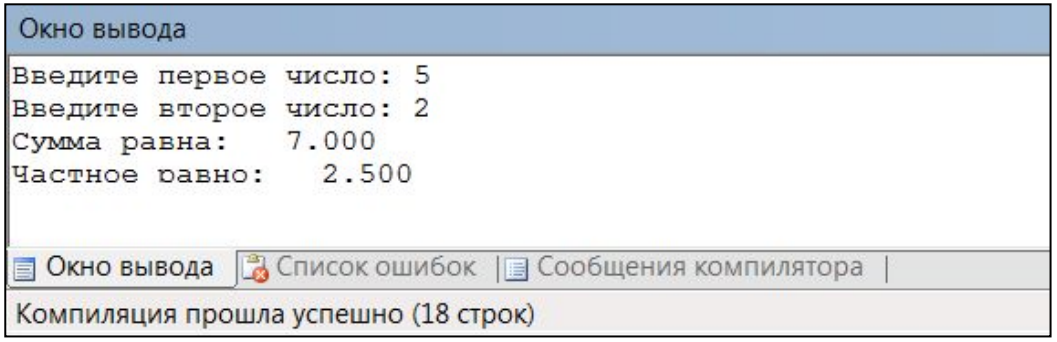
Для компиляции программы нажать Ctrl+F9 или кнопку 

Если программа составлена без ошибок, то появится сообщение об успешной компиляции программы.

Для запуска программы нажать F9 или кнопку 

Ввести исходные данные для расчета.

Просмотреть результаты расчета.



Словесная постановка задачи

Математическая постановка задачи

Разработка алгоритма и его блок-схемы

Кодирование

Контрольное тестирование и отладка программы

Анализ результатов

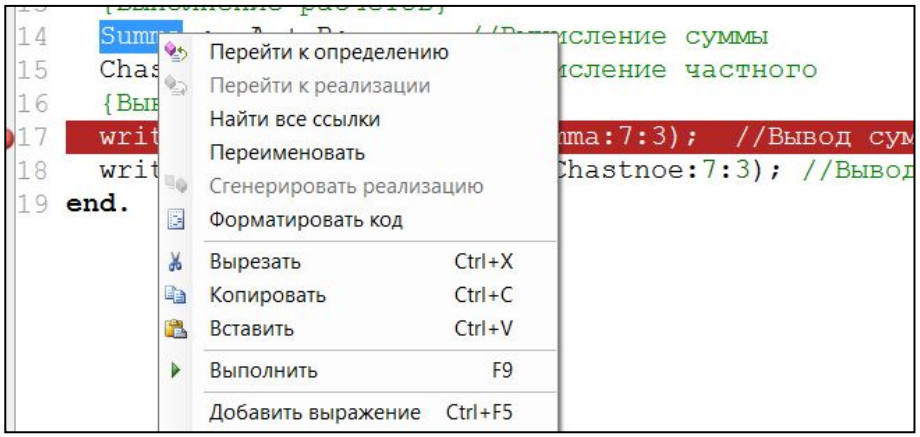
Первая программа

5. Контрольное тестирование и отладка программы:

Для приостановки программы ставится метка слева от строки:

```
15   Chastnoe := A / B;     //Вычисление частного
16   {Вывод результатов расчета}
17   writeln('Сумма равна: ', Summa:7:3);  //Вывод суммы
18   writeln('Частное равно: ', Chastnoe:7:3); //Вывод частного
19 end.
```

Для просмотра значения переменной во время выполнения программы надо выделить эту переменную и выбрать из контекстного меню «Добавить выражение» или нажать Ctrl+F5.



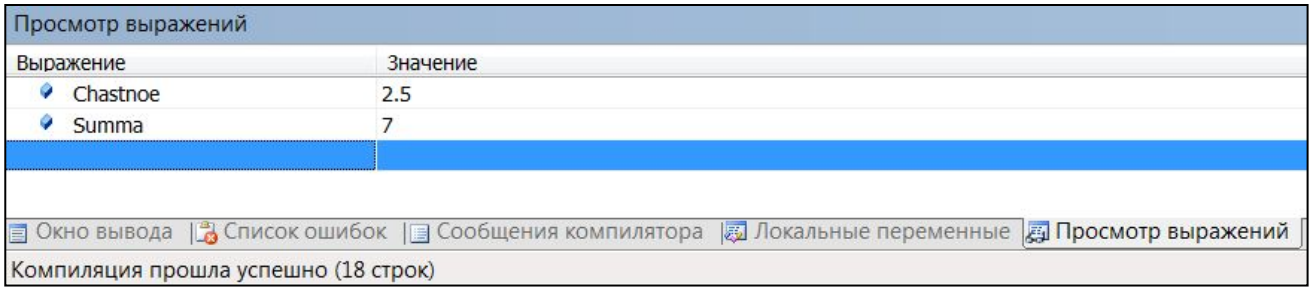
- Словесная постановка задачи
- Математическая постановка задачи
- Разработка алгоритма и его блок-схемы
- Кодирование
- Контрольное тестирование и отладка программы
- Анализ результатов

Первая программа

5. Контрольное тестирование и отладка программы:

Добавим переменные Summa и Chastnoe к списку выражений.

После запуска программа остановится на 17-й строке и на вкладке «Просмотр выражений» можно просмотреть значения интересующих переменных.



Для возобновления работы программы нажать F9.

- Словесная постановка задачи
- Математическая постановка задачи
- Разработка алгоритма и его блок-схемы
- Кодирование
- Контрольное тестирование и отладка программы
- Анализ результатов

Первая программа

6. Анализ результатов:

Программа «прогоняется» с разными значениями исходных данных.

В качестве эксперимента можно ввести нулевое значение B.

Спасибо за внимание!!!