ВВЕДЕНИЕ В ПАСКАЛЬ

Лекция 1

Символы языка

1) 26 латинских строчных и 26 латинских прописных букв
2) _ подчеркивание
3) 10 цифр: 0 1 2 3 4 5 6 7 8 9
4) знаки операций:
+ - * / = <> < > <= >= := @
5) ограничители:
. , ' ( ) [ ] (. .) { } (* *) .. : ;
6) спецификаторы: ^ # $

Состав языка

- Символы:
  - буквы: A-Z, a-z, _
  - цифры: 0-9
  - спец. символы: +, *, {, …
  - пробельные символы

- Лексемы:
  - константы 2 0.11 ‘Вася’
  - имена Vasia a _11
  - ключевые слова begin var if
  - знаки операций + - :=
  - разделители ; [ ] ,

- Выражение:
  - правило вычисления значения a + b
- Операторы:
  - исполняемые c := a + b
  - описания var a, b : real;

Константы Паскаля

Целые		Вещественные		Символь-ные	Строко-вые
Деся-тичные	16-ричные	С плавающ. точкой	С порядком		
2 15	$0101 $FFA4	-0.26 .005	1.2e4 0.1E-5	‘k’ #186 ^M	‘абырвалг’ ‘I’’m fine’

+ **Булевские**: true и false

Имена (идентификаторы)

- имя должно начинаться с буквы или _;
- имя должно содержать только буквы, знак подчеркивания и цифры;
- прописные и строчные буквы не различаются в Паскале и различаются в др. языках;
- длина имени практически не ограничена, но значащими являются первые 63 символа.

Примеры правильных имен:

Vasia, A, A13, A_and_B.

Примеры неправильных имен:

2late, Big gig,

Sюр (для C# - правильное)

Нотации

Понятные и согласованные между собой имена — основа хорошего стиля. Существует несколько *нотаций* — соглашений о правилах создания имен.

- *Нотация Паскаля*: каждое слово начинается с прописной буквы:
  - MaxLength, MyFuzzyShooshpanchik
- *Венгерская нотация* отличается от предыдущей наличием префикса, соответствующего типу величины:
  - iMaxLength, lpfnMyFuzzyShooshpanchik
- *Camel notation:* с прописной буквы начинается каждое слово, составляющее идентификатор, кроме первого:
  - maxLength, myFuzzyShooshpanchik
- Еще одна традиция — разделять слова, составляющие имя, знаками подчеркивания, при этом все составные части начинаются со строчной буквы:
  - max_length, my_fuzzy_shooshpanchik

Ключевые слова и знаки операций

- *Ключевые слова* — идентификаторы, имеющие специальное значение для компилятора. Их можно использовать только в том смысле, в котором они определены.
  - Например, для оператора перехода определено слово goto.
- *Знак операции* — один или более символов, определяющих действие над операндами. Внутри знака операции пробелы не допускаются.
  - Например, целочисленное деление в Паскале записывается div.
- Операции делятся на *унарные* (с одним операндом) и *бинарные* (с двумя).
  - В C# есть также одна тернарная операция
- *Разделители* используются для разделения или, наоборот, группирования элементов. Примеры разделителей: скобки, точка, запятая.

Концепция типа данных

Тип данных определяет:

- внутреннее представление данных, а следовательно и множество их возможных значений
- допустимые действия над данными (операции и функции)

Классификация типов Паскаля

<table>
<tr><td>Стандартные</td><td colspan="3">Определяемые программистом</td></tr>
<tr><td rowspan="2">логические
целые
вещественные
символьный
строковый
адресный
файловые</td><td>Простые</td><td colspan="2">Составные</td></tr>
<tr><td>перечисляемый
интервальный
адресные</td><td>массивы
строки
записи
множества</td><td>файлы
объекты
процедурные
типы</td></tr>
</table>

Логические типы

- Основной логический тип данных - **boolean**. Величины этого типа занимают в памяти 1 байт и могут принимать два значения: true (истина) или false (ложь). Внутреннее представление значения false — 0 (нуль), значения true — 1.
- Для совместимости с другими языками определены типы **ByteBool**, **WordBool** и **LongBool** длиной 1, 2 и 4 байта соответственно. Истинным в них считается любое отличное от нуля значение.
- К величинам логического типа применяются *логические операции* **and**, **or**, **xor** и **not** и *операции отношения*.

Операции отношения

Операция	Знак операции
больше	>
больше или равно	>=
меньше	<
меньше или равно	<=
равно	=
не равно	<>

Целые типы

Тип	Размер	Знак	Диапазон значений
integer	2 байта	есть	-32768 .. 32767 (-2^{15} .. $2^{15}-1$)
shortint	1 байт	есть	-128 .. 127 (-2^{7} .. $2^{7}-1$)
byte	1 байт	нет	0 .. 255 (0 .. $2^{8}-1$)
word	2 байта	нет	0 .. 65535 (0 .. $2^{16}-1$)
longint	4 байта	есть	-2147483648 .. 2147483647 (-2^{31} .. $2^{31}-1$)

Операции с целыми

- Арифметические операции

Операция	Знак операции
сложение	+
вычитание	-
умножение	*
деление	div
остаток от деления	mod

- Операции отношения
- Поразрядные операции: and, or, xor, not
- Операции сдвига: shl, shr

Стандартные функции для целых

Имя	Описание	Результат	Пояснения
abs	модуль	целый	|x| записывается abs(x)
arctan	арктангенс угла	вещественный	arctg x записывается arctan(x)
cos	косинус угла	вещественный	cos x записывается cos(x)
exp	экспонента	вещественный	e^x записывается exp(x)
ln	натуральный логарифм	вещественный	$\log_e x$ записывается ln(x)
odd	проверка на четность	логический	odd(3) даст в результате true
pred	предыдущее значение	целый	pred(3) даст в результате 2
sin	синус угла	вещественный	sin x записывается sin(x)
sqr	квадрат	целый	x^2 записывается sqr(x)
sqrt	квадратный корень	вещественный	$\sqrt{x}$ записывается sqrt(x)
succ	следующее значение	целый	succ(3) даст в результате 4

Стандартные процедуры

Имя	Описание	Пояснения
inc	инкремент	inc(x) — увеличить x на 1 inc(x, 3) — увеличить x на 3
dec	декремент	dec(x) — уменьшить x на 1 dec (x, 3) — уменьшить x на 3

Вещественные типы

- Внутреннее представление вещественного числа состоит из двух частей — мантиссы и порядка, и каждая часть имеет знак.
- Существует несколько вещественных типов, различающихся точностью и диапазоном представления данных.
- Точность числа определяется длиной мантиссы, а диапазон — длиной порядка.

Характеристика вещественных типов

Тип	Размер	Значащих цифр	Диапазон значений
real	6 байт	11-12	2.9e-39 .. 1.7e+38
single	4 байта	7-8	1.5e-45 .. 3.4e+38
double	8 байт	15-16	5.0e-324 .. 1.7e+308
extended	10 байт	19-20	3.4e-4932 .. 1.1e+4923
comp	8 байт	19-20	-9.22e18 .. 9.22e18 (-2^{63} .. $2^{63}-1$)

Операции с вещественными величинами

- Арифметические

  \+ - * /

- Операции отношения

  < = <= > >= <>

Функции для вещественных величин

Имя	Описание	Имя	Описание
abs	модуль	ln	натуральный логарифм
arctan	арктангенс угла	pi	значение числа π
cos	косинус угла	round	округление до целого
exp	экспонента	sin	синус угла
frac	дробная часть аргумента	sqr	квадрат
int	целая часть аргумента	sqrt	квадратный корень
		trunc	целая часть аргумента

Символьный тип

- Этот тип данных, обозначаемый ключевым словом char, служит для представления любого символа из набора допустимых символов. Под каждый символ отводится 1 байт.

- Символьная константа может записываться в тексте программы тремя способами:

- как один символ, заключенный в апострофы, например: 'A' `a';
- с помощью конструкции вида #K, где K - код соответствующего символа, при этом значение K должно находиться в пределах 0..255;
- с помощью конструкции вида ^C, где C - код соответствующего управляющего символа, при этом значение C должно быть на 64 больше кода управляющего символа.

- К символам можно применять *операции отношения* (<, <=, >,>=, =, <>), при этом сравниваются коды символов.

Функции для символьных величин

Имя	Описание	Результат
ord	порядковый номер символа	целый
chr	преобразование в символ	символьный
pred	предыдущий символ	символьный
succ	последующий символ	символьный
upcase	перевод в верхний регистр	символьный

Порядковые типы

Все возможные значения порядкового типа представляют собой ограниченное упорядоченное множество.

К любому порядковому типу могут быть применены функции:

- Ord - возвращает порядковый номер конкретного значения в данном типе;
- Pred и Succ - возвращают предыдущее и последующее значения соответственно;
- Low и High - возвращают наименьшее и наибольшее значения величин данного типа.

К порядковым относятся: логические, целые, символьный, перечисляемый, интервальный.

Переменные

- *Переменная* — это величина, которая во время работы программы может изменять свое значение.
- Все переменные, используемые в программе, должны быть описаны.
- *Для* каждой переменной задается ее *имя и тип*:

```
var number    : integer;
    x, y      : real;
    option    : char;
```

Тип переменной выбирается исходя из диапазона и требуемой точности представления данных.

- В Паскале переменные описываются в разделе описания переменных, начинающемся со служебного слова **var**.

Инициализация переменных

При объявлении можно присвоить переменной некоторое начальное значение (инициализировать).

Инициализированные переменные в Паскале описываются после ключевого слова const:

```
const
    number : integer = 100;
    x   : real = 0.02;
    option    : char = ’ю’;
```

Именованные константы

Вместо значений констант можно (и нужно!) использовать в программе их имена.

Это облегчает читабельность программы и внесение в нее изменений:

```
const
    weight = 61.5;
    n = 10;
    g = 9.8;
```

Выражения

- *Выражение* — правило вычисления значения.
- В выражении участвуют *операнды*, объединенные знаками операций.
- Операндами выражения могут быть константы, переменные и вызовы функций.
- Операции выполняются в соответствии с *приоритетами*.
- Для изменения порядка выполнения операций используются *круглые скобки*.
- Результатом выражения всегда является значение определенного типа, который определяется типами операндов.
- Величины, участвующие в выражении, должны быть *совместимых типов*.

- t + sin(x)/2 * x

результат имеет вещественный тип

- a <= b + 2

результат имеет логический тип

- (x > 0) and (y < 0)

результат имеет логический тип

Совместимость типов данных

Типы являются совместимыми, если:

-они эквивалентны;

-являются оба либо целыми, либо действительными;

-один тип – интервальный, другой – его базовый;

-оба интервальные с общим базовым;

-один тип – строковый, другой – символьный.

Для приведения типов используется конструкция
Имя_Типа(переменная или значение)
Например, **Integer('Z')** представляет собой значение кода символа **'Z'** в двухбайтном представлении целого числа,
а **Byte(534)** даст значение **22**, поскольку целое число **534** имеет тип **Word** и занимает два байта, а тип **Byte** занимает один байт, и в процессе приведения старший байт будет отброшен.

Приоритеты операций Паскаля

1. Первичные - (), []
2. Унарные - not, минус -, взятие адреса @.
3. Операции типа умножения:

   * / divmod and shlshr.
3. Операции типа сложения:

   \+ - or xor.
4. Операции отношения:

   = <> < > <= >= in.

Контрольный вопрос

Чему равно значение выражений:

```
2 +  1e1 / 2  *  5
```

```
10E-1 + 1 div 2
```

Структура простейшей программы на Паскале

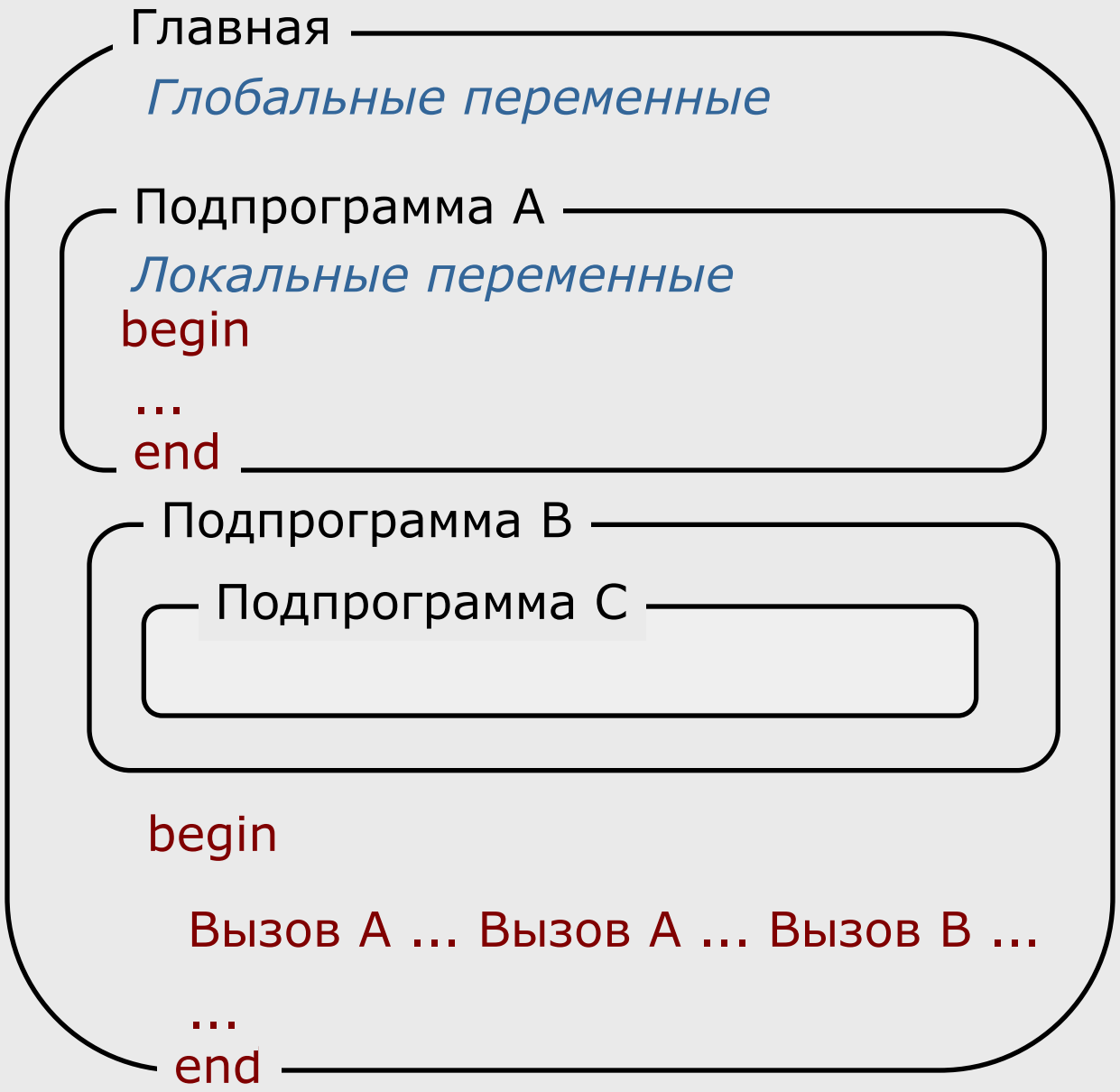
```
Program <имя>;          { заголовок }
   <разделы описаний>

begin
   <раздел операторов>
end.

(*    программа заканчивается точкой    *)
```

Разделы описаний

- *Раздел описания модулей*
  uses crt, graph, my_module;
- *Раздел описания констант*
  const MaxLen = 100; g = 9.8;
  koeff : integer = 5;
- *Раздел описания переменных*
  var number : integer;
  x, y : real;
- *Раздел описания меток*
  label 1, 2, error;

Разделы описания типов, процедур и функций будут рассмотрены позже

Общая структура программы на Паскале



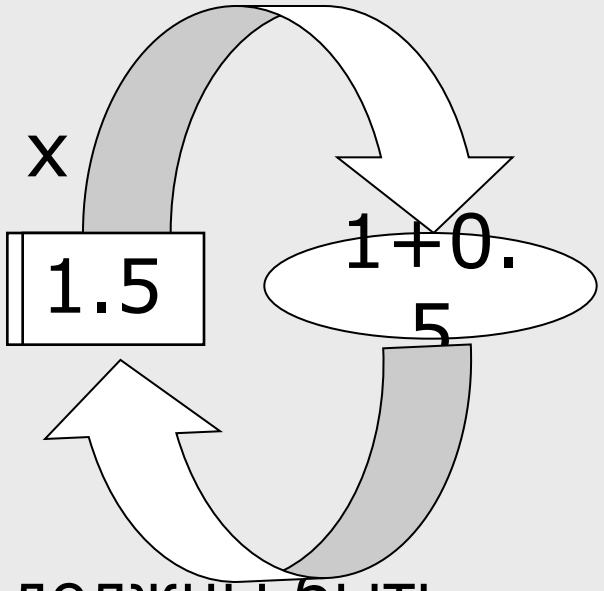
Оператор присваивания

Присваивание – это замена старого значения переменной на новое.

Старое значение стирается бесследно

- переменная := выражение

- a := b + c;
- x := 1;
- x := x + 0.5;



Величины в левой и правой части должны быть **совместимы по присваиванию**, например:

вещественная переменная := целое выражение;

Ввод с консоли

read(список);
readln[(список)];

Значения при вводе разделяются пробелами, Tab или Enter

Ввод значения каждой переменной выполняется так:

- значение переменной выделяется как группа символов, расположенных между разделителями;
- эти символы преобразуются во внутреннюю форму представления, соответствующую типу переменной;
- значение записывается в ячейку памяти, определяемую именем переменной.

```
var    a : integer; b : real;
...
readln(a, b);
```

Вывод на экран

write(список);
writeln[(список)];

```
var    a : integer;
       b : real;
       d : char;
...
writeln(’Значение a = ’, a:3, ’ b = ’, b:5:2,
          sin(a) + b);
```

```
Значение a =   1 b =  4.124.9614709848E+00
```

Правила записи процедур вывода

- Список вывода разделяется ***запятыми***.
- Список содержит выражения логических, целых, вещественных, символьного и строкового типов.
- После любого значения можно через двоеточие указать ***количество отводимых под него позиций***.
- Для вещественных чисел можно указать второй формат, указывающий, сколько позиций ***из общего количества*** позиций отводится ***под дробную часть*** числа.
- Если форматы не указаны, под целое число, символ и строку отводится минимально необходимое для их представления количество позиций. Под вещественное число отводится 17 позиций, 10 из них — под дробную часть.
- Форматы могут быть выражениями целого типа.

Пример: перевод температуры из F в C

```
program temperature;
var fahr, cels : real;
begin
    writeln('Введите температуру по Фаренгейту');
    readln(fahr);
    cels := 5 / 9 * (fahr - 32);
    writeln('По Фаренгейту: ', fahr:6:2,
            ' в градусах Цельсия: ', cels:6:2);

end.
```

Тест №1 для самопроверки

1) Выберите все правильные ответы.
В переменной типа byte можно хранить число:
1 13
2 213
3 -13
4 -213
5 1213

2) Выберите все правильные ответы.
Число 256 можно хранить в переменной типа:
1 byte
2 word
3 shortint
4 longint
5 real

3) Выберите все допустимые константы

1 '\\'
2 $00FH
3 -7.12e-13
4 'Ж''
5 1,23

4) Какие выражения не содержат синтаксических ошибок?

1 -0.18 * Pi / r - 0.2 * t
2 (-0.18) * Pi / 1(r - 0.2)
3 cos^2 * x + 0,2
4 (-0.18) * Pi) \ (r - 0.2 * t))

5) Чему равно значение выражения

e + sqrt(e) * 1e1 / 2 * a при e=4, a=3 ?

6) Чему равно значение выражения

a and not b xor c

при a = true, b = true, c = false?

1 false
2 true
3 нечто среднее

7) Какие выражения не содержат синтаксических ошибок?

1 sin(abs(0.6e3 * y_t))
2 a div b / c * mod
3 $EF01 * 1.34E-2 / i7_17
4 1_2i - exp(y) / 2 * t

Ответы

1) Выберите все правильные ответы.
В переменной типа byte можно хранить число:
1 13
2 213
3 -13
4 -213
5 1213

2) Выберите все правильные ответы.
Число 256 можно хранить в переменной типа:
1 byte
2 word
3 shortint
4 longint
5 real

3) Выберите все допустимые константы

1 '\\'
2 $00FH
3 -7.12e-13
4 'Ж''
5 1,23

4) Какие выражения не содержат синтаксических ошибок?

1 -0.18 * Pi / r - 0.2 * t
2 (-0.18) * Pi / 1(r - 0.2)
3 cos^2 * x + 0,2
4 (-0.18) * Pi) \ (r - 0.2 * t))

5) Чему равно значение выражения

(e + (((sqrt(e) * 10) / 2) * a)) при e=4, a=3 ? **34**

6) Чему равно значение выражения

при a = true, b = true, c = false?

1 false
2 true
3 нечто среднее

7) Какие выражения не содержат синтаксических ошибок?

1 sin(abs(0.6e3 * y_t))
2 a div b / c * mod
3 $EF01 * 1.34E-2 / i7_17
4 1_2i - exp(y) / 2 * t

Итоги

- 7 баллов – «отлично»
- 6 баллов – «хорошо»
- 4-5 баллов – «удовлетворительно»