Соловьев С.Ю.

soloviev@glossary.ru

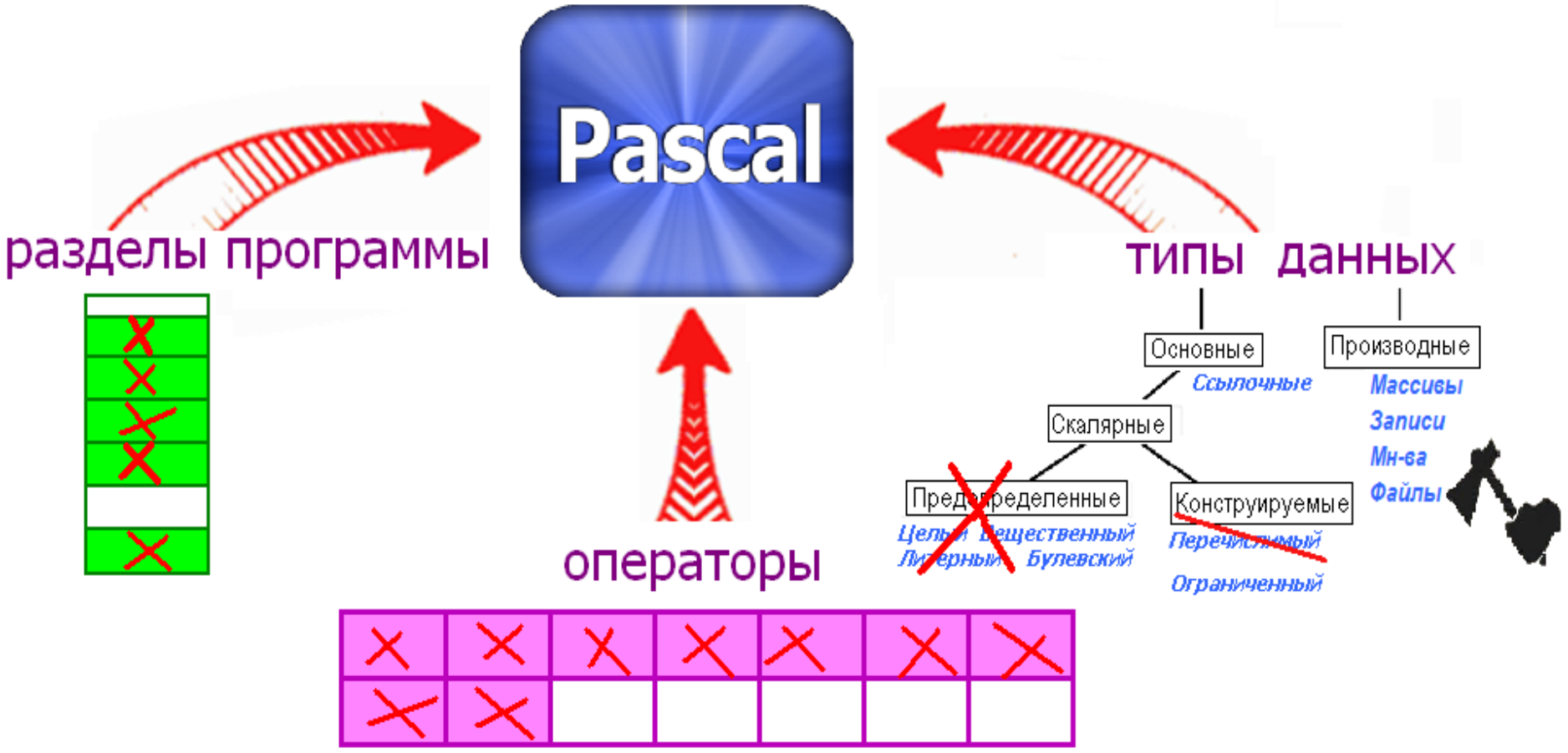
Алгоритмы и Алгоритмические языки

www.park.glossary.ru/pascal/

Лекция No.7

2022

Напоминание



Конструируемые
Перечислимый
Ограниченный

Ограниченные типы

Если M – номер месяца, то M : integer ⊕ (1<=M) and (M <=12)

Ограничения на базовый тип

pred('A') <задание ограниченного типа> ::=
<константа1>..<константа2>

```
type month = 1..12;
     digit = '0'..'9';
     week  = (pon,vto,sre,che,pia,sub,vos);
```

(А) Переменные ограниченного типа

```
var M : month;
    W : pon..pia;
```

(Б) Значения: из диапазона

(В) Операции базового типа (в пределах диапазона)

Если

```
var M : month;
```

то

```
M:=100;    {-}
M:=5;      {+}
M:=M+2;    {+}
```

Регулярные типы данных / Массивы

Массив – упорядоченный набор фиксированного количества однотипных компонент.

Прагматика

Векторы $(a_0, a_1, \ldots, a_n)$	`a[k]`
Матрицы n×m $\|\|b_{ij}\|\|$	`b[i,j]`, `b[i]`
Поезд No. 47, вагон 7	`Train[47,7]`
Дом 5, корпус А	`dom[5,'A']`

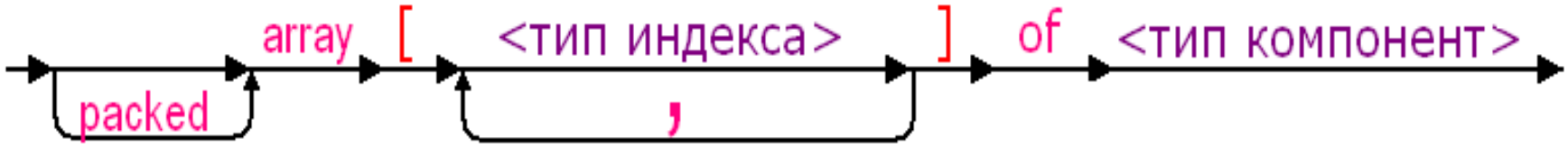
a b Train Dom k i i,j 47,7 5,'A'

ИмяМассива[Индекс/Индексы]

char || boolean || перечислимый || ограниченный

Регулярные типы данных / Массивы

pred('A') <задание регулярного типа> ::=



<тип индекса> ::= char | boolean | <огр.тип> | <пер.тип>
<огр.тип> ::= <имя> | <задание ограниченного типа>
<пер.тип> ::= <имя> | <задание перечислимого типа>
<тип компонент> ::= <тип> *(* без файлов *)*

(A) Массивы:

```
var A : array [char] of integer;
    B : array [0..255,boolean] of real;
    C : array [0..511] of
        array [1..128] of char;
```

Регулярные типы данных / Массивы

(Б) Значение: упорядоченный набор фиксированного количества однотипных компонент.

<переменная с индексами> ::=

<переменная-массив> [<индекс> {,<индекс>}]

<переменная-массив> ::= <имя>

<индекс> ::= <выражение>

```
var A : array [char] of integer;
    B : array [0..255,boolean] of real;
    C : array [0..511] of
        array [1..128] of char;
```

```
A['x']      A[pred('z')]
B[17,X=Y]   B[K+M-1,true]
C[18]       C[18][K+M-1] = C[18,K+M-1]
```

Регулярные типы данных / Массивы

(B) Операции.

<переменная с индексами> -- частный случай

<переменная>

⇨ *участвует в выражениях и операторах*

```
var  A,B : array ...          type artype = array ...
                                   var A : artype;
                                   ...
                                   var B : artype;
```

```
A:=B;
```

Пример

Ввести коэффициеты полинома
и вычислить его значения
для заданных аргументов.

$$a_0x^n + a_1x^{n-1} + \ldots + a_n$$

```
program BRAIN;

Label 999;
Const MXA = 20; (*Max стпнь полинома*)
Type  Coef = array [0..MXA] of real;

Var   I,N : integer;
      R,X : real;
        Y : char;
        A : Coef;

begin   write('Степень полинома =');
        Readln(N);
        if (N < 1) or (MXA < N) then begin
           writeln('Не могу');
           goto 999
        end;
```

```
                    (* Ввод коэффициентов *)
        for I:=0 to N do begin
           write(I:2,'-й коэффициент =');
           Readln(A[I])
        end;
                              (* Вычисления *)
        repeat
           writeln;
           write(' Аргумент =');
           Readln(X);
           R:=0;          (* Схема Горнера *)
           for I:=0 to N do R:=R*X+A[I];
           writeln('Результат =',R:16:4);
           write('Продолжать ');
           Readln(Y);
        until (Y = 'N') or (Y = 'n');
999: end.
```

```
Степень полинома = 2
0-й коэффициент = 16.5
1-й коэффициент = -5.3
2-й коэффициент =  4.4
Аргумент =  -1
Результат = 26.2000
Продолжать N
```

Строки (стандарт ISO)

Для удобства обработки символьной информации

packed array [1..▒] of char

длина, целое > 1

Строки одной длины = строки одного типа.

Дополнительные возможности:

- допускаются строки-константы: AM:='Привет' ;
- допускается сравнение строк одной длины;

  < | <= | > | >= | = | <>
- строки допускаются в качестве элементов вывода.

Пример

Программа (1) запрашивает у пользователя даты в формате YYMMDD,
(2) сортирует эти даты и
(3) выводит (на экран) упорядоченную последовательность дат

```
program DATE;

Label  999;
Const  MXA = 20;
Type   DaType = packed array [1..6] of char;

Var   I,J,N,R : integer;
          DT : array [1..MXA] of DaType;
           Y : DaType;

begin   write('Количество дат =');
        Readln(N);
        if (N < 1) or (MXA < N) then begin
           writeln('Не могу');
           goto 999
        end;

        for I:=1 to N do begin(* Ввод дат *)
           write(I:2,'-я дата =');
           for J:=1 to 6 do Read(DT[I,J]);
           Readln
        end;
```

```
                                (* Сортировка *)
  for I:=1 to N-1 do begin
     R:=I;
     for J:=I+1 to N do
     if DT[J] < DT[R] then R:=J;{Сравнение}
     Y:=DT[R];                (*    Поменять   *)
        DT[R]:=DT[I];         (*    местами    *)
              DT[I]:=Y  (* DT[R] и DT[I] *)
  end;

  writeln('Результат');
  for I:=1 to N do  writeln(I:2,'. ',DT[I]);
999:end.
```

```
Количество дат = 4
 1-я дата = 201020
 2-я дата = 190917
 3-я дата = 200112
 4-я дата = 200229
```

```
190917
200112
200229
201020
```

Программа :: **Раздел процедур и функций**

<заголовок программы>;	
<раздел меток>	::= <пусто> | label <метка> {, <метка> };
<раздел констант>	::= <пусто> | const <описание константы>; { <описание константы>; }
<раздел типов>	::= <пусто> | type <описание типа>; { <описание типа>; }
<раздел переменных>	::= <пусто> | var <описание переменных>; { <описание переменных>; }
<раздел процедур и функций>	::= { <описание процедуры>; | <описание функции>; }
<раздел операторов>.	*есть* <составной оператор>.

Программа

<программа> ::= <заголовок программы>;<блок>.

<блок> ::= <раздел меток>
<раздел констант>
<раздел типов>
<раздел переменных>
<раздел процедур и функций>
<раздел операторов>

<заголовок программы> ::= program <имя программы> |
program <имя программы>(<имя файла>{,<имя файла>})

Два примера: program sep27 **vs.** program sep28(F,G)

Процедуры

Процедура ≈ фрагмент программы,
имеющий самостоятельное значение.

Исторически: многократно повторяющийся.

применение: оператор процедуры ▶

описание

⇩ *неформально*

procedure Имя(параметры; параметры; . . .); Блок

имена тип способ: по имени | по значению

формально

<описание процедуры> ::= <заголовок процедуры>; <блок>

◀ <программа> ::= <заголовок программы>; <блок>.

<описание функции> ::= <заголовок функции>; <блок> ▶

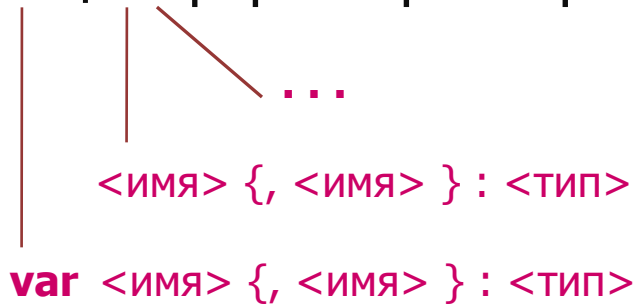
Процедуры :: Описание

<описание процедуры> ::= <заголовок процедуры>; <блок>

<заголовок процедуры> ::= procedure <имя процедуры> |
procedure <имя процедуры>(<список формальных параметров>)

<имя процедуры> ::= <идентификатор>

<список формальных параметров> ::=
<секция форм.параметров> { ; <секция форм.параметров> }



Процедуры :: Пример

Задача. Преобразование дат YYMMDD → DD-MM-20YY.

Например: 210927 → 27-09-2021

Пусть
```
type alfa06 = packed array [1.. 6] of char;
type alfa10 = packed array [1..10] of char;
```

```
procedure RECO(var FD : alfa06; var TD : alfa10);
begin   TD:='**-**-20**';
        TD[1]:=FD[5];  TD[ 2]:=FD[6];   { DD }
        TD[4]:=FD[3];  TD[ 5]:=FD[4];   { MM }
        TD[9]:=FD[1];  TD[10]:=FD[2]    { YY }
end;
```

В описании процедуры можно пользоваться любыми допустимыми именами.

Процедуры :: Оператор

Активизация, вызов, обращение-к

<оператор процедуры> **::=** <имя процедуры> **|**

<имя процедуры>**(**<список фактических параметров>**)**

<список фактических параметров> **::=**

<фактический параметр> **{ ,**<фактический параметр>**}**

Требование: *К-во фактических = К-во формальных*

Семантика: 1. Настройка параметров ▶

2. Вычисление процедуры

Пример оператора процедуры

```
program  Sep21;
type   alpha06 = packed array [1..  6] of char;
       alpha10 = packed array [1..10] of char;

var  A : alpha06;
     B : alpha10;
     procedure RECO(var FD : alfa06; var TD : alfa10);
     begin   TD:='**-**-20**';
             TD[1]:=FD[5];  TD[ 2]:=FD[6];   { DD }
             TD[4]:=FD[3];  TD[ 5]:=FD[4];   { MM }
             TD[9]:=FD[1];  TD[10]:=FD[2]    { YY }
     end;
begin   A:='191023';   RECO(A,B);  writeln(B);
        A:='200807';   RECO(A,B);  writeln(B);
end.
```

```
23-10-2019
07-08-2020
```

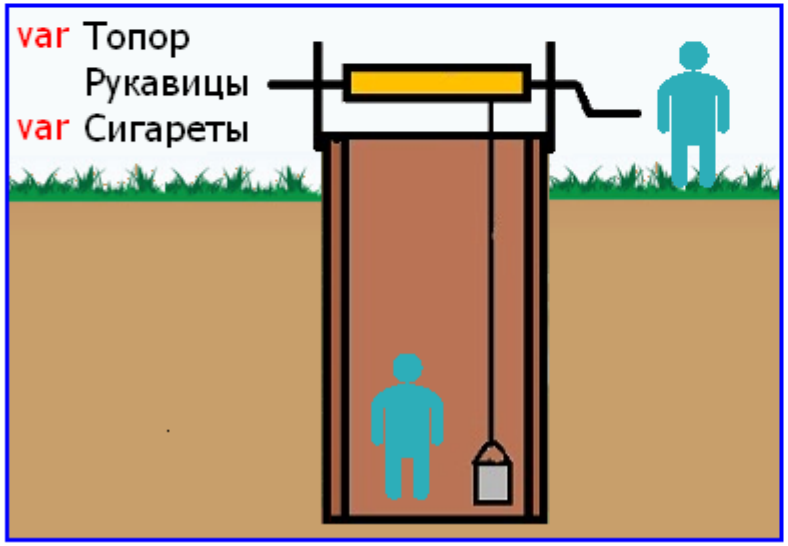
Процедуры :: Настройка параметров

<формальный параметр> ⟷ <фактический параметр>

▒ : <тип>$_2$ ⟶ д.б. выражение типа <тип>$_2$

Имя форм. параметра — Значение выражение

var ▒ : <тип>$_1$ ⟶ д.б. <переменная> : <тип>$_1$



Имя факт. параметра — Значение

Имя форм. параметра — Значение

Обзор лекции No.7

Регулярные типы данных:

- задание регулярного типа;
- значение регулярного типа;
- переменные с индексами;

Строки (стандарт ISO)

Заголовок ПАСКАЛЬ-программы

Раздел процедур и функций ПАСКАЛЬ-программы

Описание процедуры

Оператор процедуры

Параметры-значения и параметры-переменные

--- Конец лекции No. 7 ---