Соловьев С.Ю.

soloviev@glossary.ru

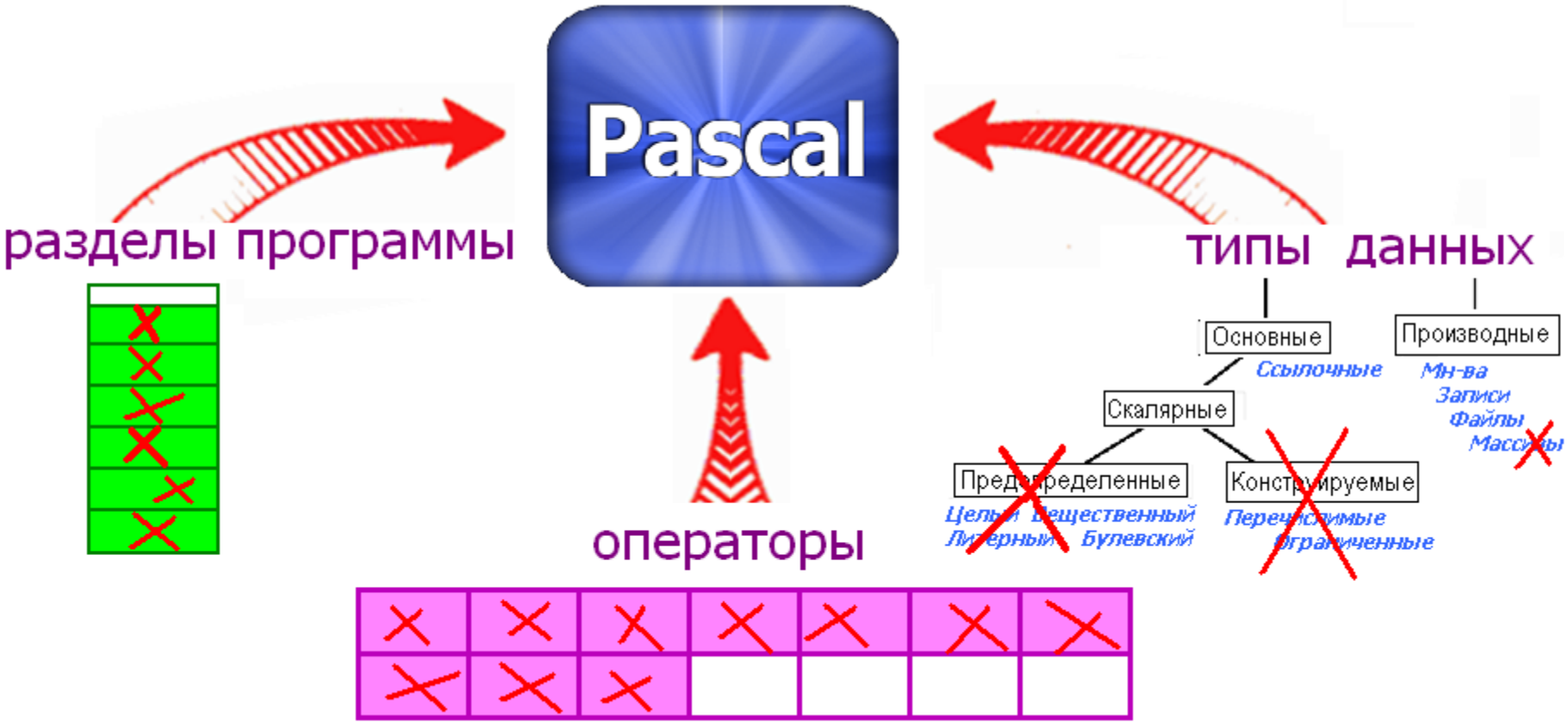
Алгоритмы и Алгоритмические языки

www.park.glossary.ru/pascal/

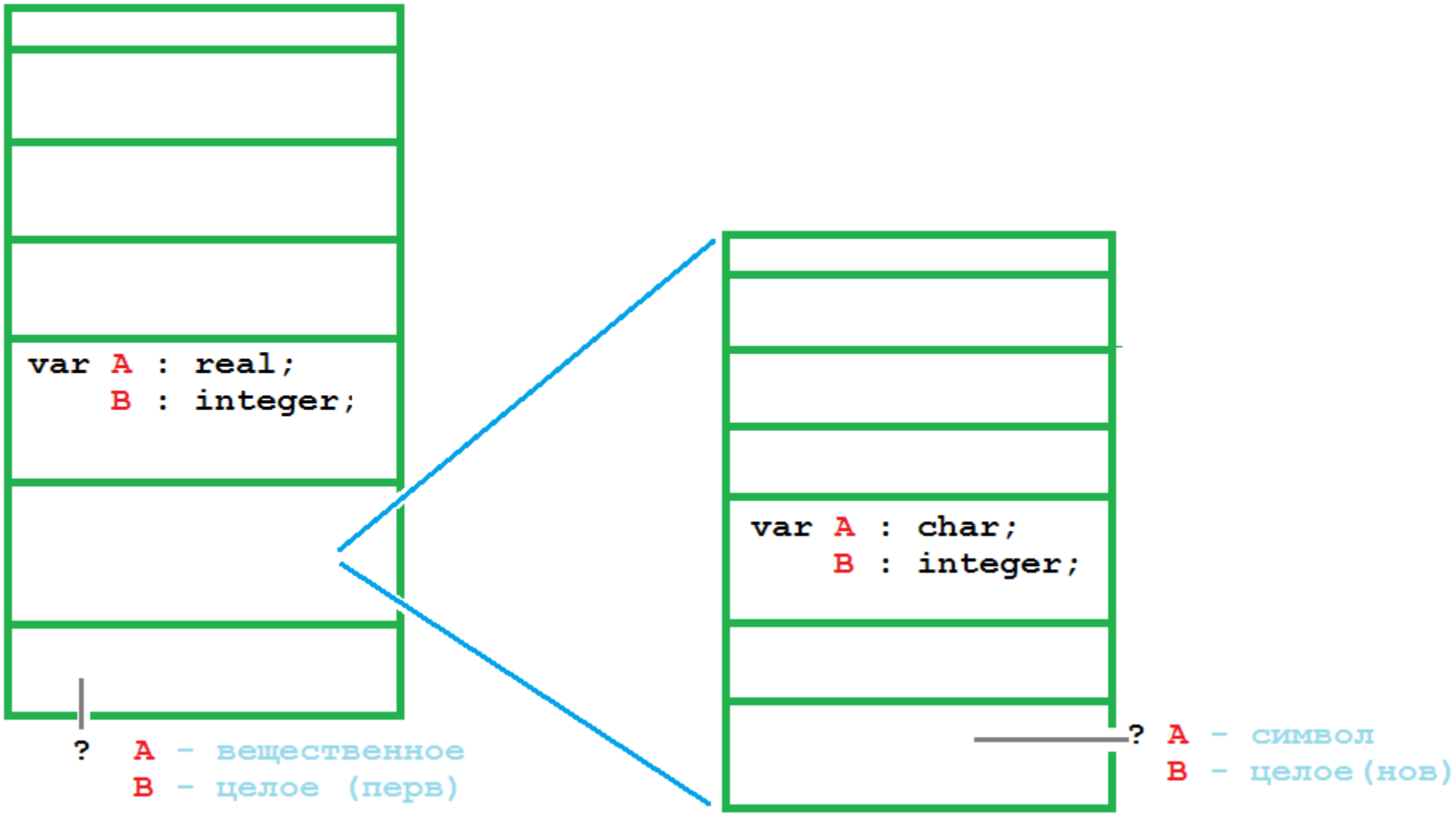
Лекция No. 8

2022

Напоминание



В описании процедуры можно пользоваться любыми допустимыми именами.



Принцип локализации

Принцип локализации

Пусть **P** – произвольная процедура или функция, заданная своим определением.

Обозначим:

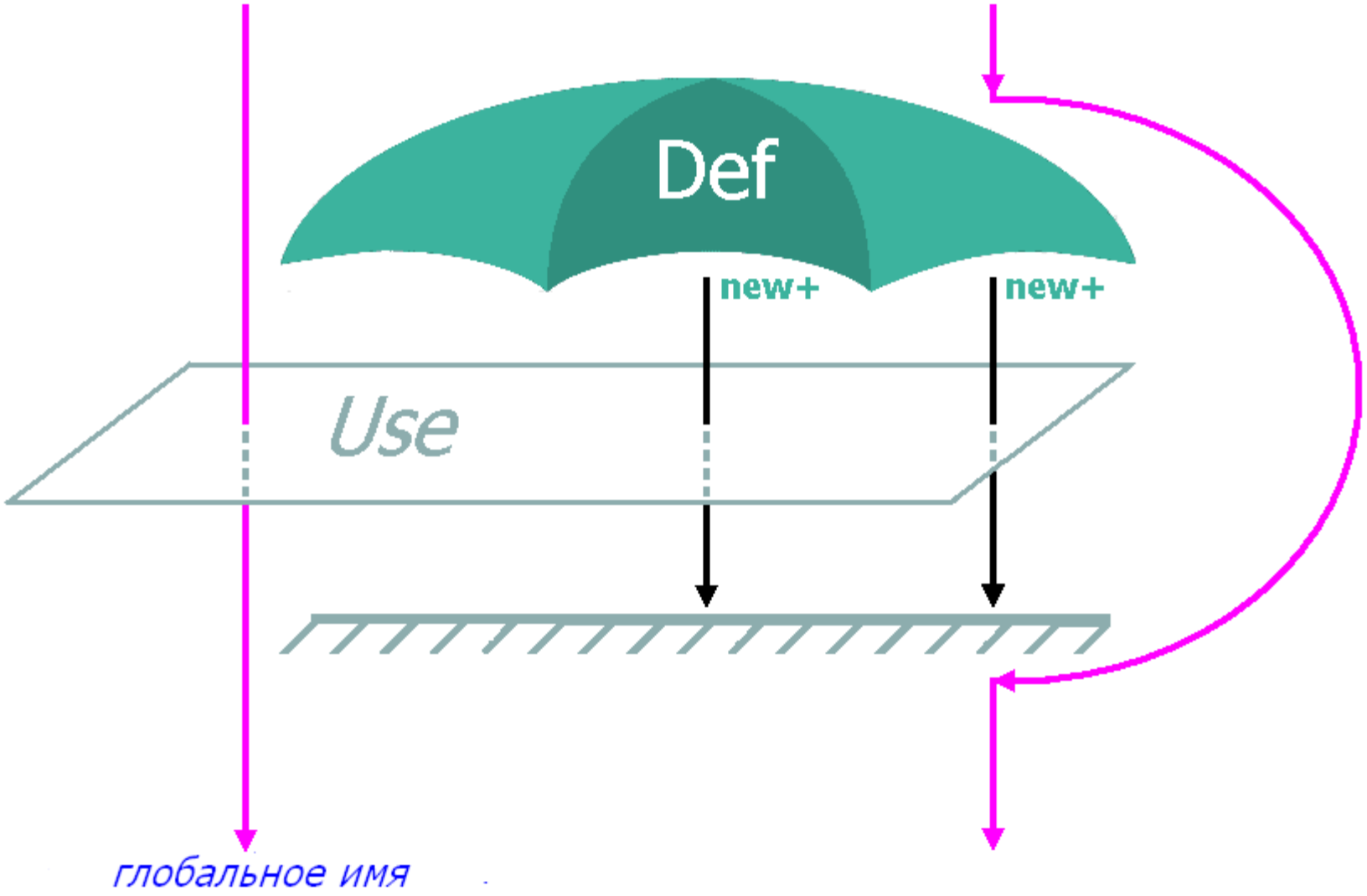
Def – множество имен, определенных в **P**;

Use – множество имен, используемых в **P**.

параметры константы
типы переменные имена
процедур имена функций

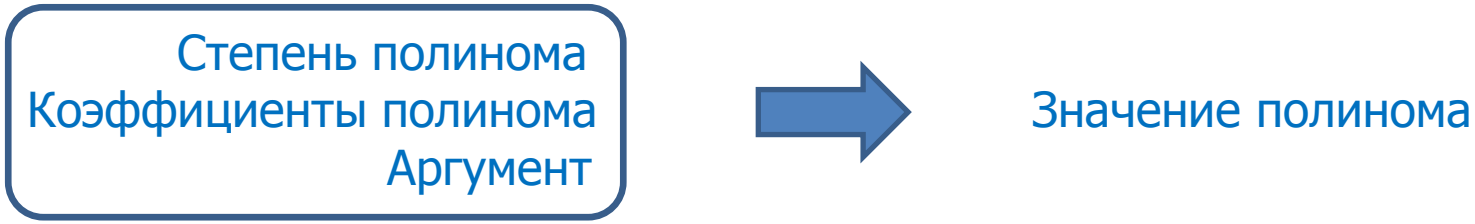
1. Если $\alpha \in \text{Use} \cap \text{Def}$, то α считается определенным $^{+}$ в **P**.
2. Если $\alpha \in \text{Use} \setminus \text{Def}$, то для α действует его определение на момент вызова **P**.
3. При завершении вызова **P** все определения имен из Def теряют силу.

Принцип локализации – 2



Функции. Описание

Функция ≈ частный случай процедуры,
вычисляющей некоторое простое значение.



<описание функции> ::= <заголовок функции>; <блок>
+ правила

<заголовок функции> ::= function <имя функции> : <имя типа> |
function <имя функции>
(<список формальных параметров>) : <имя типа>

<имя функции> ::= <идентификатор>

Функции :: Правила

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

1. ∃ оператор вида <имя функции> := <выражение>
2. Результат функции определяется последним таким оператором.

Пример 1. $P(x) = a_0x^n + a_1x^{n-1} + \ldots + a_n$

Пусть `type COEF : array [0..40] of real;`

```
function Pgorner(var A : COEF; N : integer; X : real) : real;
  var R : real;
      I : integer;
begin R:=0;
      for I:=0 to N do R:=R*X+A[I];
      Pgorner:=R                                    end;
```

Функции :: Пример No.2/1

Пример 2. Поиск в массиве B заданного числа P.
Результат = либо 0, либо номер позиции.

Пусть `type MACC = array [1..2000] of integer;`

```
function FIND(var B : MACC; N,P : integer) : integer;  {Вариант 1. }
   var I,F : integer;
begin   I:=0;
        F:=0;
        while (I < N) and (F = 0) do begin
              I:=I+1;
              if B[I] = P then F:=I;
        end;
        FIND:=F
end;
```

Функции :: Пример No.2/2

```
function FIND(var B : MACC; N,P : integer) : integer;      { Вариант 2. }
  label 999;
  var I : integer;
begin   FIND:=0;
        for I:=1 to N do
        if B[I] = P then begin
             FIND:=I;
             goto 999
        end;
999:end;
```

Вариант 1.

```
function FIND(var B : MACC; N,P : integer) : integer;
  var I,F : integer;
begin  I:=0;
       F:=0;
       while (I < N) and (F = 0) do begin
            I:=I+1;
            if B[I] = P then F:=I;
       end;
       FIND:=F
end;
```

```
function FIND(var B : MACC; N,P : integer) : integer;      { Вариант 3. }
   var I : integer;
begin   FIND:=0;
        for I:=1 to N do
        if B[I] = P then begin
           FIND:=I;
           Exit
      end;                                                  end;
```

Напоминание

procedure : √ Задание √ Оператор

function : ☞ Задание Применение

√ Pgorner Find

Функции :: Вызовы

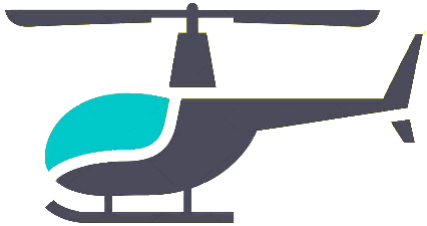
Пусть: var Y : real;
G : COEF;
N : integer;

➡ Y:=Pgorner(G,N,0.2);
Y:=sin(Y)+Pgorner(G,N,Y);

Пусть: var K,N : integer;
D : MACC;

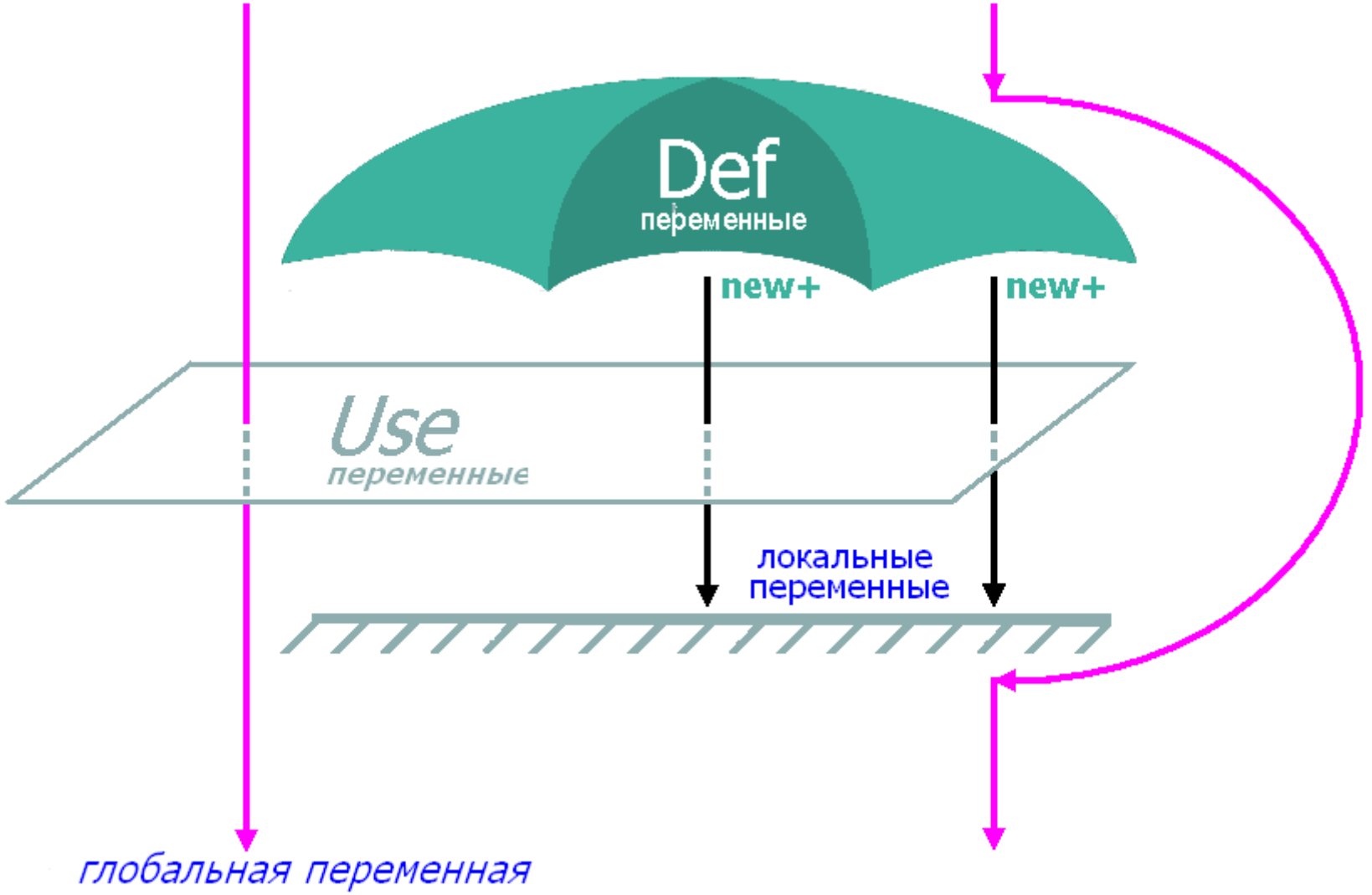
➡ K:=FIND(D,N,-66);
K:=FIND(D,N,K);

Особенности процедур и функций



1. Локальные и глобальные переменные
2. Побочные эффекты
3. Оператор **Exit**
4. Рекурсивные процедуры и функции
   4.1 Поиск пути в лабиринте
5. Предописания
6. Процедуры и функции как параметры

Особенности :: Локальные и глобальные переменные



Особенности :: Побочные эффекты

Побочный эффект – действия, определенные в процедуре или функции, которые

- изменяют глобальные переменные или
- выполняют незапланированный ввод/вывод и т.д.

```
program EFF;
  var A,B : real;
  procedure FCX(X : real) : real;
    var A : real;
  begin   A:=0.7*X+0.6;  {A лок.}
          FCX:=A*X+0.3
  end;
begin   A:=1;
        B:=FCX(1);
        writeln(A:10:2, B:10:2);
end.
```

1.00 1.60

```
program EFF;
  var A,B : real;
  procedure FCX(X : real) : real;
    var A1 : real;
  begin   A:=0.7*X+0.6; {A глоб.}
          FCX:=A*X+0.3
  end;
begin   A:=1;
        B:=FCX(1);
        writeln(A:10:2, B:10:2);
end.
```

1.60 1.60

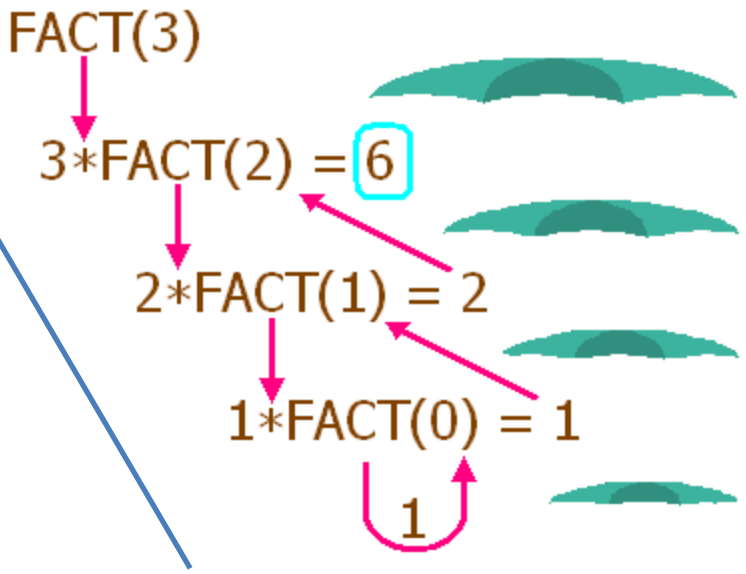
Особенности :: Рекурсивные процедуры и функции

Рекурсивная процедура – процедура, в определении которой имеется обращение к самой себе.

Рекурсивная функция – функция, в определении которой имеется обращение к самой себе.

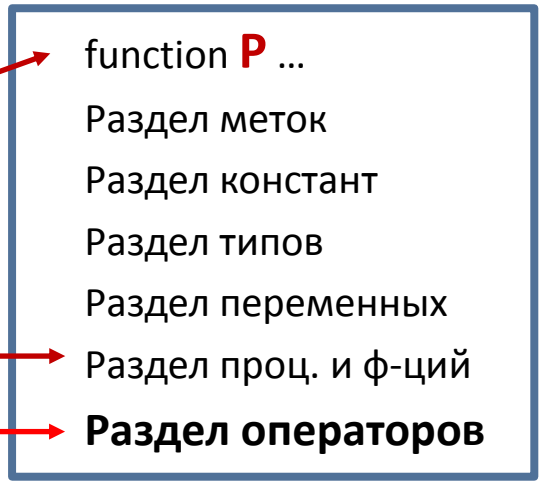
$$N! = \begin{cases} 1, & \text{если } N = 0 \\ N \times (N-1)!, & \text{если } N > 1 \end{cases}$$

```
function FACT(N : integer) : integer;
begin     if N = 0 then FACT:=1
                   else FACT:=N*FACT(N-1)
end;
```



Особенности :: Рекурсивные процедуры и функции

Процедура/функция **P**
называется **рекурсивной**,
если в разделах проц. и ф-ций
и/или операторов
имеется исполняемый оператор*),
при выполнении которого возможно
обращение к P (вообще говоря, с другими аргументами).



---

*) Исполняемый оператор – оператор, который выполняется при некотором наборе значений фактических параметров и/или глобальных переменных.

Особенности :: Рекурсивные процедуры

Соглашения: I → IncOne

δδδδ ... → D ⊕ N

Св-во 1. $I(\delta_1 \ldots \delta_{n-1} 0) = \delta_1 \delta_2 \ldots \delta_{n-1} 1, \quad n > 0$

Св-во 2. $I(\delta_1 \ldots \delta_{n-1} 1) = I(\delta_1 \delta_2 \ldots \delta_{n-1}) 0, \quad n > 0$

Св-во 3. $I(\,) = 1, \quad n = 0$

```
program Sep22;
 type alfa10 = packed array [1..10] of char;
 var   X : alfa10;
       L : integer;
   procedure IncOne(var D : alfa10; var N : integer);
   begin       if    N  = 0  then begin  D[1] :='1'; N:=1 end      {Св. 3}
               else  if D[N] = '0' then          D[N]:='1'          {Св. 1}
               else  begin                                          {Св. 2}
                        N:=N-1;  IncOne(D,N);                  { рек. вызов }
                        N:=N+1;           D[N]:='0'
                     end
     end;
 begin   X:='10111_______';   L:=5;
         IncOne(X,L);   writeln(X)
 end.
```

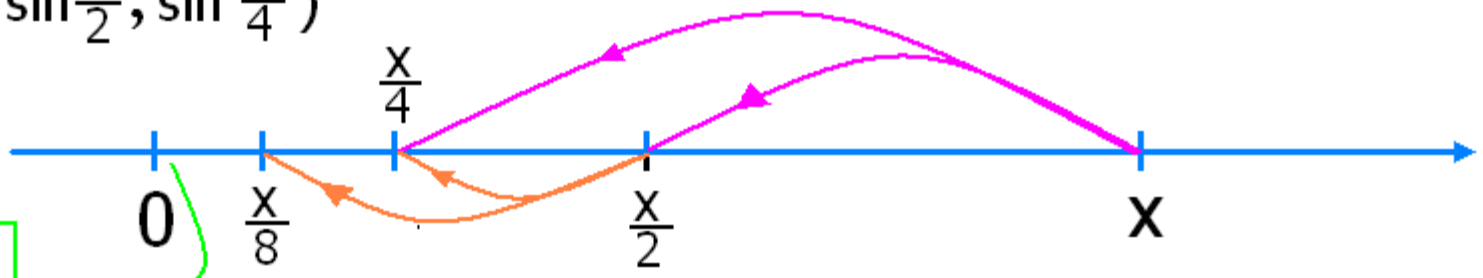
11000_______

Особенности :: Рекурсивное определение sin(x)

Стефанюк В.Л. // Программирование, 1981

$$\sin x = 2\sin\frac{x}{2}\cos\frac{x}{2} = 2\sin\frac{x}{2}\left(\cos^2\frac{x}{4} - \sin^2\frac{x}{4}\right) = 2\sin\frac{x}{2}\left(1 - 2\sin^2\frac{x}{4}\right)$$

$$\sin x = \varphi\left(\sin\frac{x}{2}, \sin\frac{x}{4}\right)$$



$$\frac{\sin x}{x} \xrightarrow[x \to 0]{} 1$$

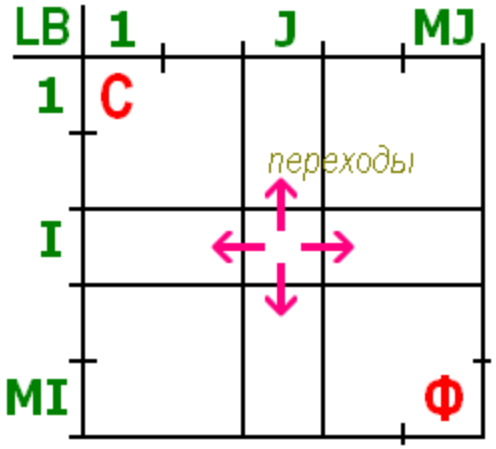
$\sin x \sim x$, если $|x| < 0.0001$

$$\sin x \cong \begin{cases} x & \text{, если } |x| < 0.0001 \\ \varphi\left(\sin\frac{x}{2}, \sin\frac{x}{4}\right) & \text{иначе} \end{cases}$$

```
function  Rsin(X : real) : real;
begin   if abs(X) < 0.0001 then Rsin:=X
                           else Rsin:=2*Rsin(X/2)*(1-2*sqr(Rsin(X/4)))
end;
```

$$\ln(1+x) = \ln\left(1+\frac{x}{x+2}\right) - \ln\left(1-\frac{x}{x+2}\right)$$

Особенности :: Путь в лабиринте

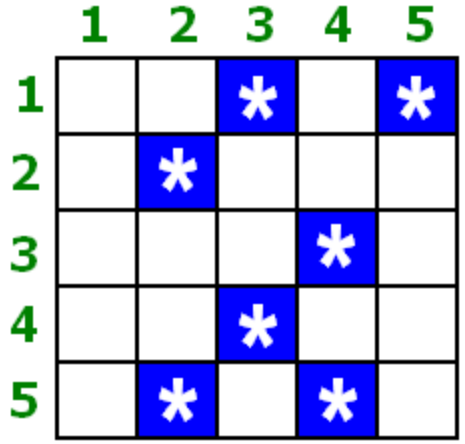


$$LB[i,j] = \begin{cases} \text{'*'}, \text{ переход в клетку } (i,j) \text{ запрещен} \\ \text{'␣'}, \text{ переход в клетку } (i,j) \text{ разрешен} \end{cases}$$

Задача. ∃ последовательность переходов из (1,1) в (MI,MJ) ?

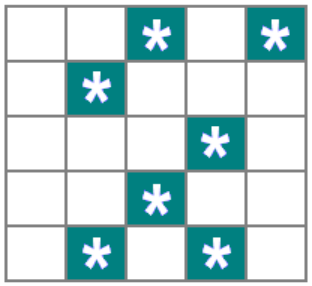
Идея. Отметить знаком '+' те клетки, в которые можно перейти из (1,1) за несколько шагов.

Mark: Отметить клетку (i,j) знаком '+' (если это возможно), и сделать из нее все возможные переходы.



Особенности :: Путь в лабиринте-2

```
program Sep24;
        const MI = 5;  MJ = 5;
        var    LB : array [1..MI] of packed array [1..MJ] of char;
        procedure Mark(I,J : integer);
        begin   if (1 <= I) and (I <= MI) then
                if (1 <= J) and (J <= MJ) then
                if          LB[I,J] = '␣'       then begin
                   LB[I,J]:='+';
                   Mark(I+1,J);   Mark(I-1,J);
                   Mark(I,J+1);   Mark(I,J-1)
                end
        end;
begin   LB[1]:='␣␣*␣*';
        LB[2]:='␣*␣␣␣';
        LB[3]:='␣␣␣*␣';
        LB[4]:='␣␣*␣␣';
        LB[5]:='␣*␣*␣';
```



```
        Mark(1,1);
        if      LB[MI,MJ] = '+'
        then writeln('Есть путь')
        else  writeln('Нет пути');
end.
```

Особенности :: Предописания

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

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

<предписание процедуры> ::= <заголовок процедуры>; forward

<предописание функции> ::= <заголовок функции>; forward

```
program Sep24a;
... procedure ABC;
    begin ... CALL(A,B); ... end;
... procedure CALL(X,Y : real);
    begin                    end;
begin ...
end.
```

Error 3: Unknown identifier

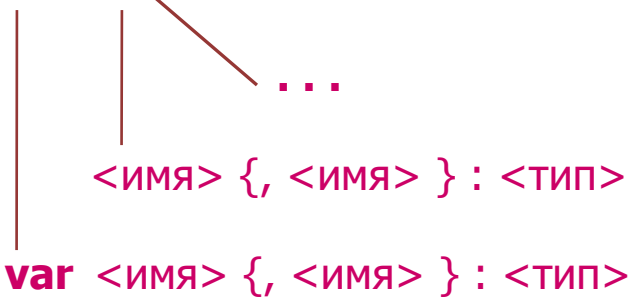
```
program Sep24b;
procedure CALL(X,Y : real); forward;
... procedure ABC;
    begin ... CALL(A,B); ... end;
... procedure CALL(X,Y : real);
begin                        end;
begin ...
end.
```

Compile successful

F1 обращается к **F2**; **F2** обращается к **F3**; **F3** обращается к **F1**

Особенности:: Параметры-процедуры и параметры-функции

<секция форм.параметров>



<секция форм.параметров> **::=** <параметры-значения> |
<параметры-переменные> |
<параметр-функция> |
<параметр-процедура>

<параметр-функция> **::=** <заголовок функции>

<параметр-процедура> **::=** <заголовок процедуры>

В списке фактических параметров – имена функций и процедур, заголовки которых совпадают с заголовками из списка формальных параметров.

с точностью до имен процедур/функций

Особенности:: Параметры-процедуры и параметры-функции

Вычислить $(a_1p_1 + \ldots + a_n p_n) / (p_1 + \ldots + p_n)$

Пусть var A,P : array [1..40] of real;

```
function Sep24c(N : integer) : real;
    function Radd(function F(I : integer) : real;
                                  N : integer) : real;
          var  I : integer;
               R : real;
    begin     R:=0;  for I:=1 to N do R:=R+F(I);
                     Radd:=R                                          end;
    function RC(I : integer) : real;     begin    RC:=A[I]*P[I]   end;
    function RZ(I : integer) : real;     begin    RZ:=      P[I]   end;
begin
                  Sep24c:=Radd(RC,N) / Radd(RZ,N)
end;
```

Обзор лекции No.8

Принцип локализации

Функции в языке ПАСКАЛЬ

Описания функций

Вызовы функций

Локальные и глобальные переменные

Побочные эффекты

Рекурсивные процедуры и функции

Предописания процедур и функций

Определение секции формальных параметров

Параметры-процедуры и параметры-функции

--- Конец лекции No. 8 ---