

**FLIDE**

FLOGOL Integrated Development  
Environment

Система

функционально-логического  
программирования  
на языке S-FLOGOL

# Язык S-FLOGOL

- Основан на теории направленных отношений (НО) (Фальк В.Н., Кутепов В.П.).
- Имеет развитые средства схемного описания НО.
- Допускает использование:
  - индексированных имен НО,
  - параметризованных НО.
- Поддерживает объектно-ориентированный стиль программирования.
- Позволяет строить многомодульные программы.
- Обладает средствами ограничения области видимости определяемых отношений (Private-домены).

# Направленные отношения

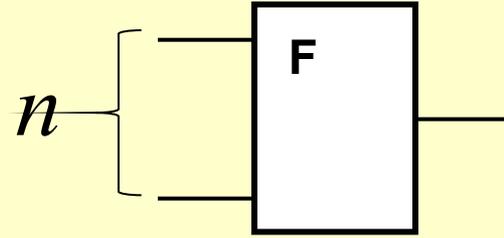
- Направленным отношением  $R$  арности  $(n, m)$  на носителе  $D$  называется множество упорядоченных пар кортежей элементов  $D$  длины  $n$  и  $m$ , соответственно.
- НО  $R^{(n, m)}$  может моделироваться  $(n+m)$ -арным отношением  $R'$  на  $D$ :

$$\langle a_1, \dots, a_n, a_{n+1}, \dots, a_{n+m} \rangle \in R' \approx \langle a_1, \dots, a_n, a_{n+1}, \dots, a_m \rangle \in R.$$

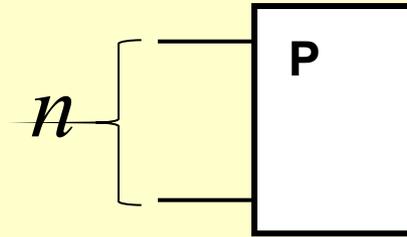
- Естественной и наглядной формой определения НО и основой эффективной организации их вычисления является представление в форме сетевых языков.
- НО характеризуются наличием фундаментальных свойств – *тотальности* и *функциональности* прямых и обратных НО.

# Семантические объекты

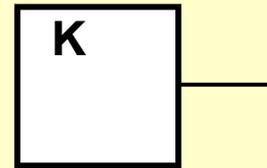
- Функция  $F^{(n,1)}$



- Предикат  $P^{(n,0)}$

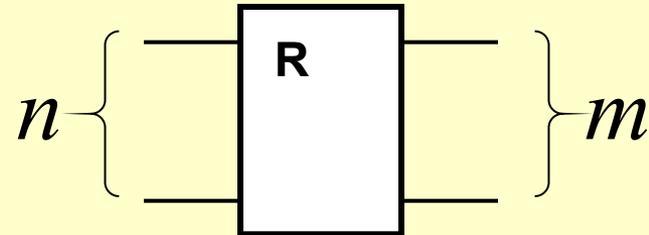


- Константа  $K^{(0,1)}$



- НО общего вида

$$R^{(n,m)}$$



# Варианты определения НО

Пример: определение натурального числа 3

- В форме графика:

$$\text{Nat3} = \{:\text{Succ}(\text{Succ}(\text{Succ}(\text{Null}))))\}$$

- В форме композиции:

$$\text{Nat3} = \text{Null} \cdot \text{Succ} \cdot \text{Succ} \cdot \text{Succ}$$

- При помощи свертки:

$$\text{Nat3} = \text{Null} \cdot (\cdot \text{I}=1..3)\text{Succ}$$

# Определение НО в виде графика

- В форме графика НО задается в виде

$R = \{ \text{Терм1} : \text{Терм2} ? \text{Формула} \}$  , где

Терм1 – входной терм (образец аргументов вызова),

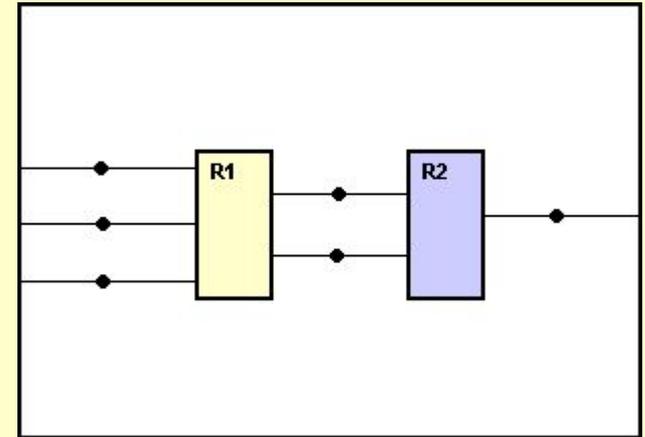
Терм2 – выходной терм (результат вызова),

Формула – набор ограничений вида  $\text{Терм} = \text{Терм}$ ,  $\text{Терм} <> \text{Терм}$ .

# Композиционное определение НО

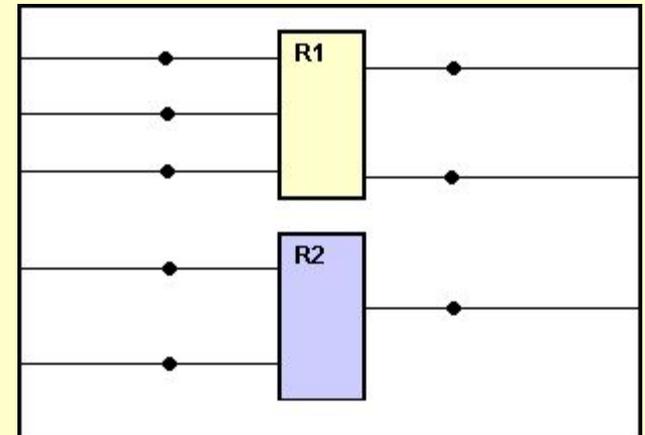
- Последовательная композиция

$$R^{(n,m)} = R1^{(n,n')} \bullet R2^{(n',m)}$$



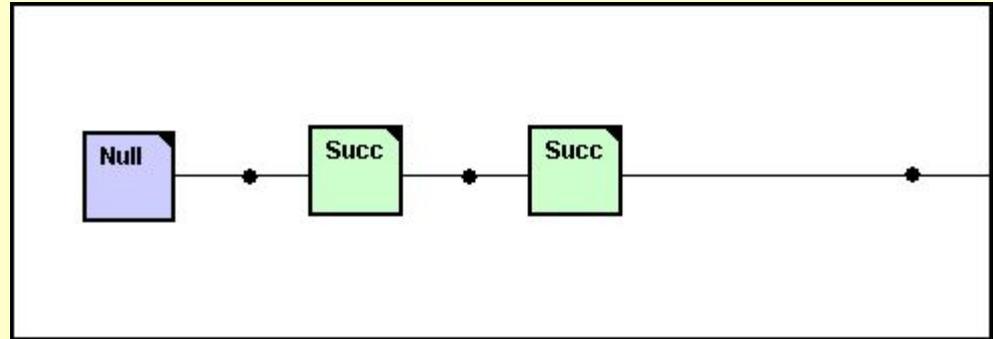
- Параллельная композиция

$$R^{(n'+n'',m'+m'')} = R1^{(n',m')} \# R2^{(n'',m'')}$$

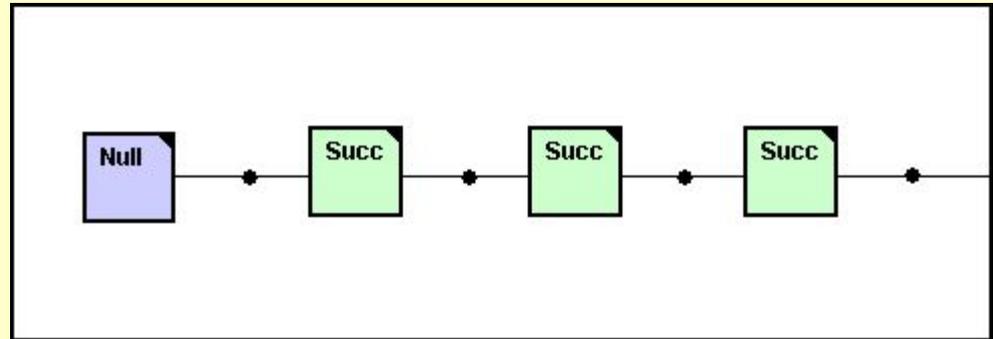


# Представление натуральных чисел

Число 2 (сеть):



Число 3 (сеть):

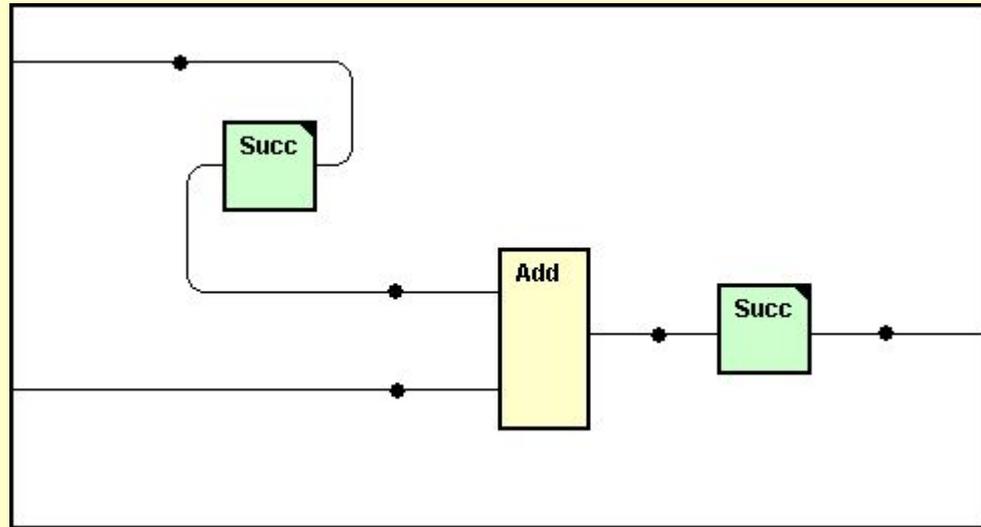
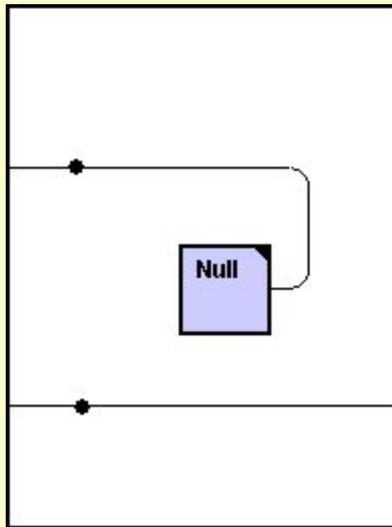


Числа 2,3 (текст):

```
(0:1)Null;  
(1:1)Succ;  
Nat2={:Succ(Succ(Null))};  
Nat3={:Succ(Succ(Succ(Null)))};
```

# Отношения над числами

Пример: НО  $Add^{(2,1)}$



(0:1)Null;

(1:1)Succ;

(2:1)Add={Null,x:x};

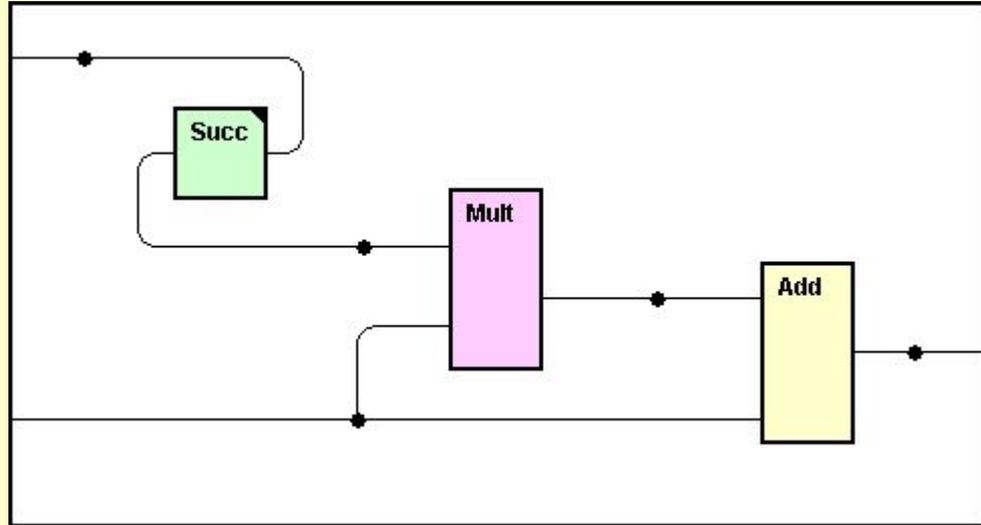
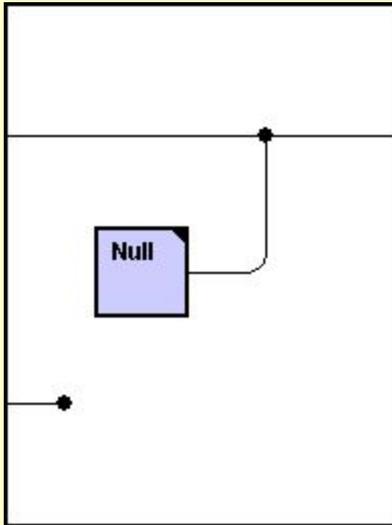
(2:1)Add={Succ(x),y:Succ(@x,y)}

*или*

(2:1)Add={Null,x:x}U{Succ(x),y:Succ(@x,y)}

# Отношения над числами

Пример: НО  $Mult^{(2,1)}$



$(0:1)Null;$

$(1:1)Succ;$

$(2:1)Mult=\{Null,x:Null\};$

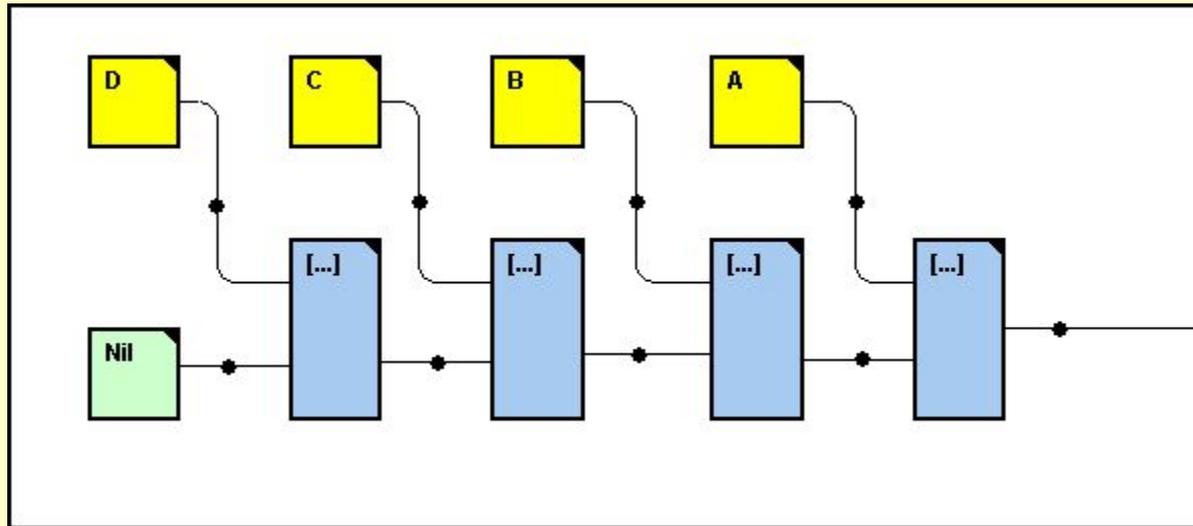
$(2:1)Mult=\{Succ(x),y:Add(@ (x,y),y)\}$

**или**

$(2:1)Mult=\{Null,x:Null\} \cup \{Succ(x),y:Add(@ (x,y),y)\}$

# Списки

Пример: НО  $List^{(0,1)}$  [A,B,C,D]



(0:1)Nil;

(1:1)LCons;

(0:1)A;

(0:1)B;

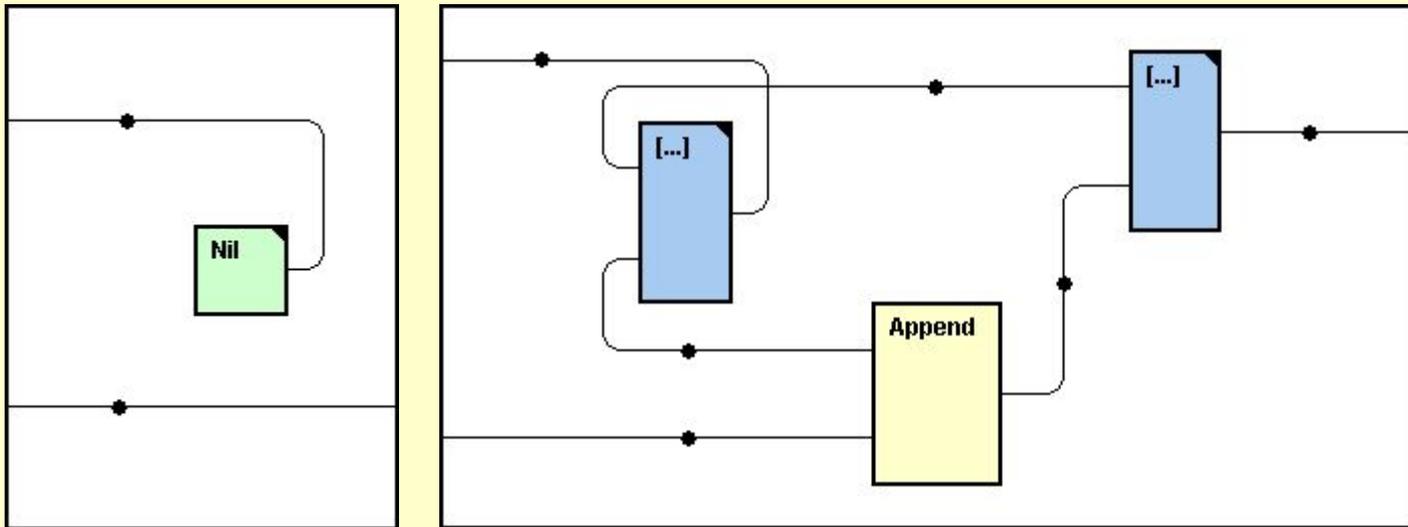
(0:1)C;

(0:1)D;

(0:1)List={:LCons(A,LCons(B,LCons(C,LCons(D,Nil))))}

# Отношения над списками

Пример: НО  $Append^{(2,1)}$



(0:1) Nil;

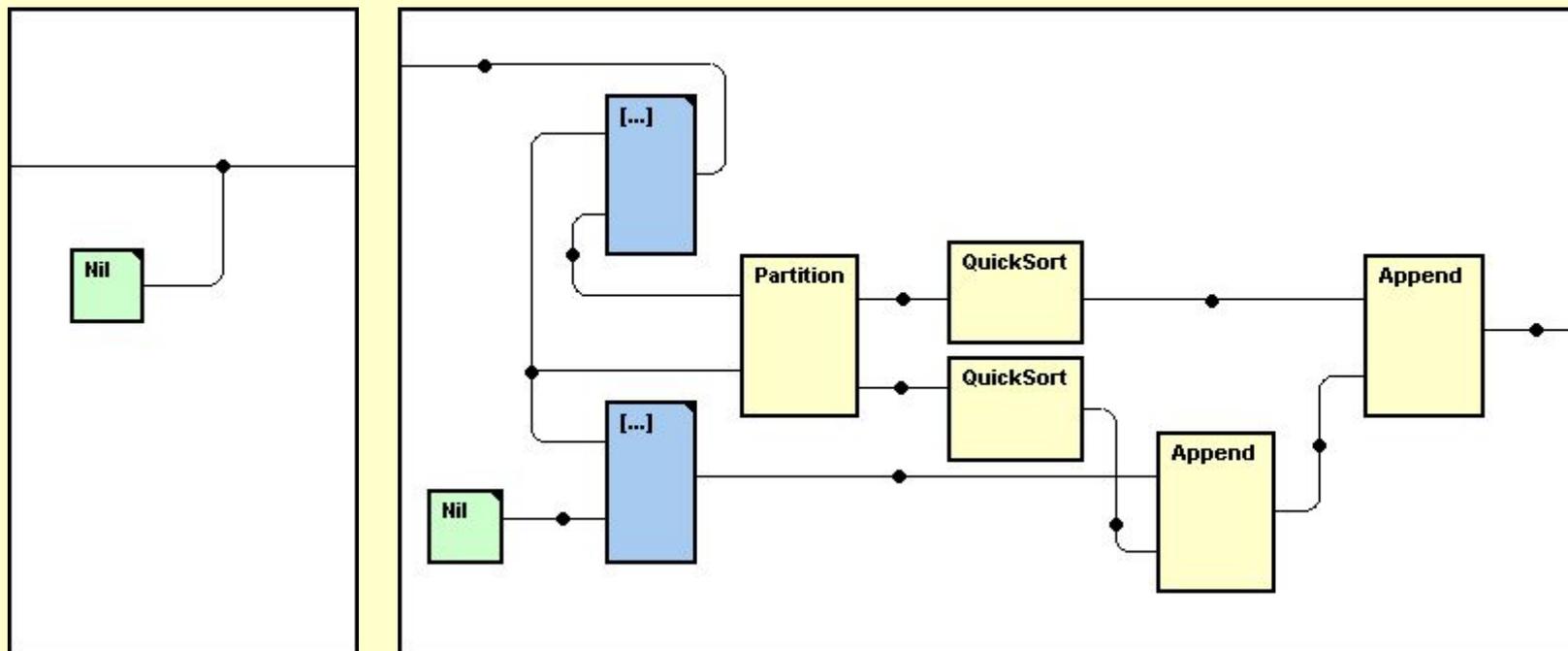
(2:1) LCons;

(2:1) Append = { Nil, x:x };

(2:1) Append = { LCons(x,xs), ys:LCons(x, @(xs,ys)) }

# Отношения над списками

Пример: НО  $QuickSort^{(1,1)}$



QuickSort=

{LCons(x,xs):Append(@ls,Append(LCons(x,Nil),@bs))

?Partition(xs,x)=ls,bs};

QuickSort={Nil:Nil};

# Индексированные имена НО

Пример: множество натуральных чисел

$$(I=1..)[I]Nat = \text{Null} \cdot (\cdot J=1..I) \text{Succ}$$

Эквивалентное множество определений:

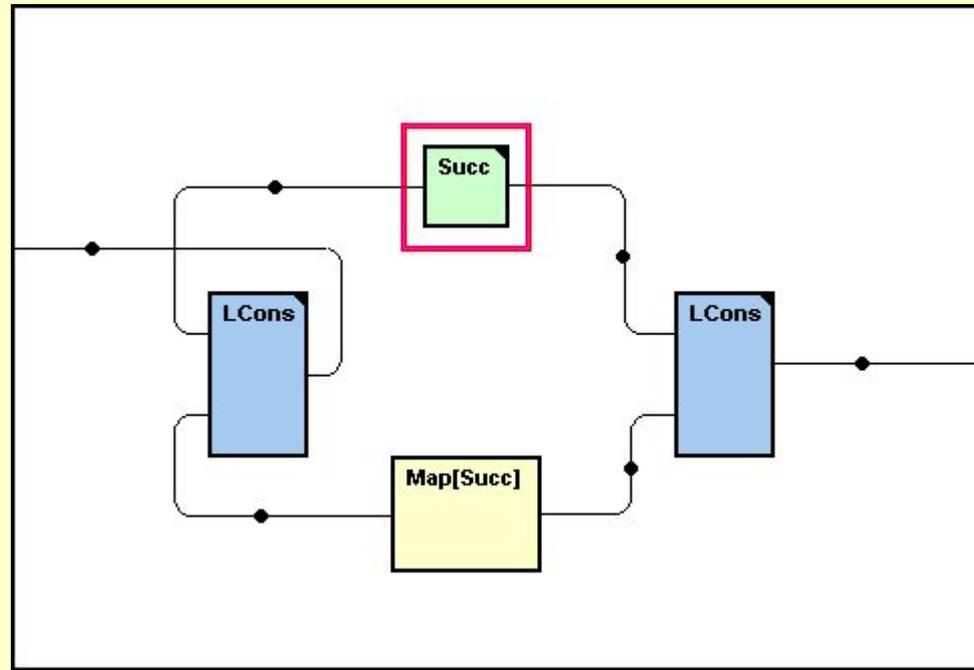
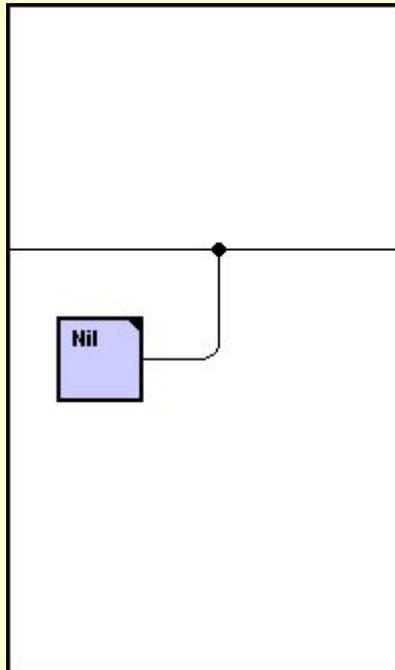
```
[0]Nat = Null;  
[1]Nat = Null · Succ;  
[2]Nat = Null · Succ · Succ;  
[3]Nat = Null · Succ · Succ · Succ;  
...
```

# Параметризованные НО

Пример:НО  $SuccList^{(1,1)}$

(инкрементация элементов списка)

```
Map={LCons(x,xs):LCons(«0»(x),@(xs))};  
Map={Nil:Nil};  
SuccList = Map[Succ]
```



# Типовые НО

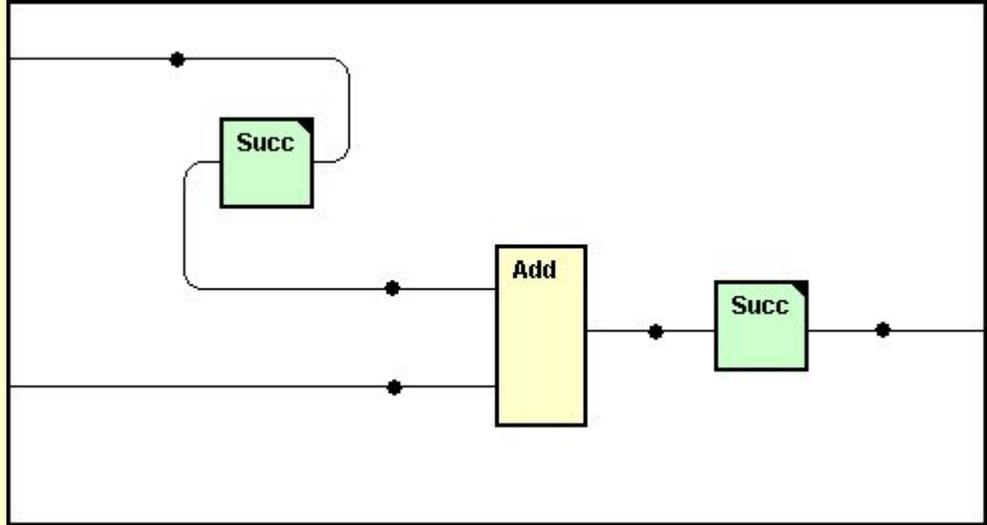
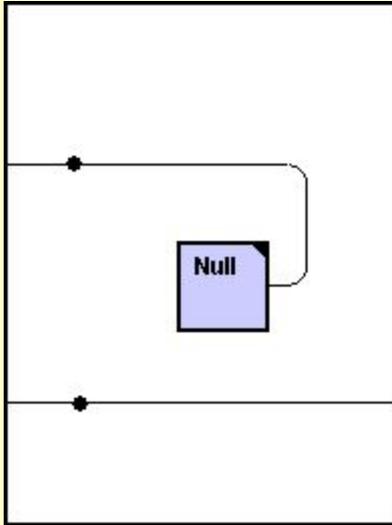
Типовое отношение для типа  $T = (t'_1, \dots, t'_{n'} \rightarrow t''_1, \dots, t''_{n''})$ ,  
где  $t'_i(t''_i)$  – сорта входных(выходных) данных,  
определяется в сетевой форме как:

$$\text{Type}T = \{G_{t'_1}, \boxtimes, G_{t'_{n'}} : G_{t''_1}, \boxtimes, G_{t''_{n''}}\} \cap \langle\langle 0 \rangle\rangle,$$

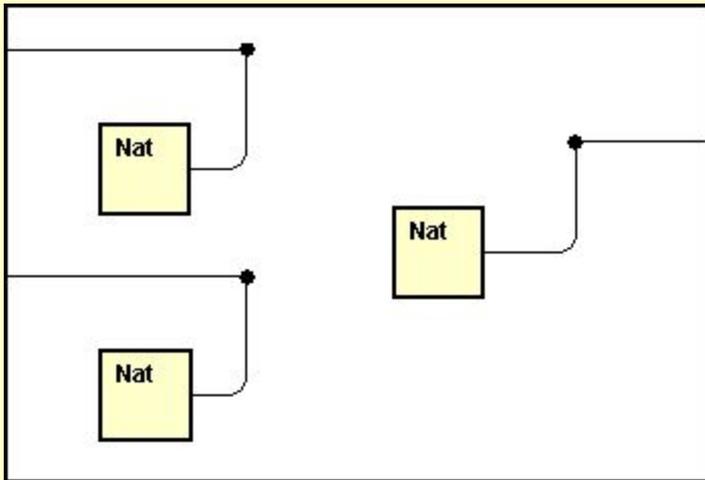
где  $G_t$  – отношение арности  $(0,1)$ , генерирующее  
данные сорта  $t$ . Описание отношения  $R$  типа  $T$   
будет иметь вид  $R = \text{Type}T[e]$ , где  $e$  – реляционное  
выражение, определяющее направленное  
отношение арности  $(n', n'')$ .

# Типизация НО

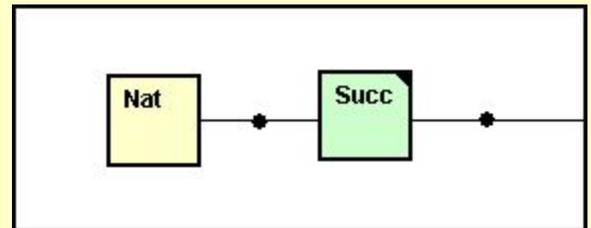
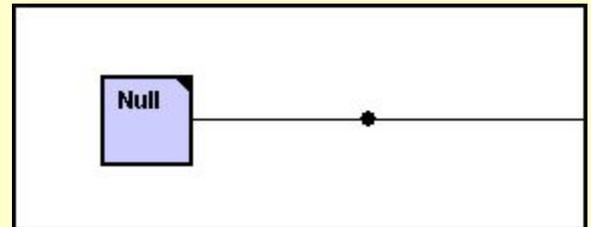
Пример: НО  $Add^{(2,1)}$



$TNatAdd^{(2,1)}$



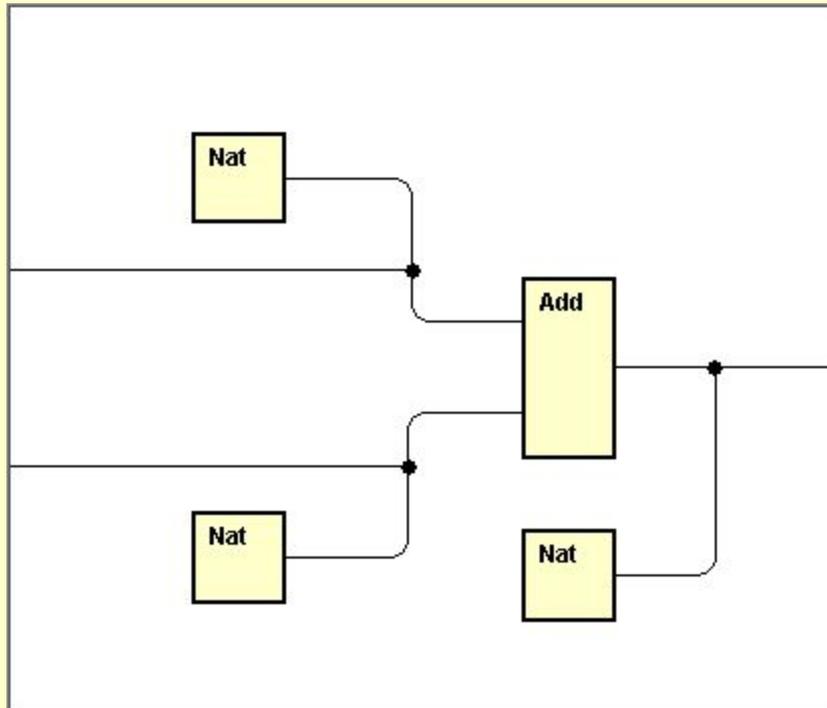
$Nat^{(0,1)}$



# Типизация НО

Пример: типизированное НО  $TypedAdd^{(2,1)}$

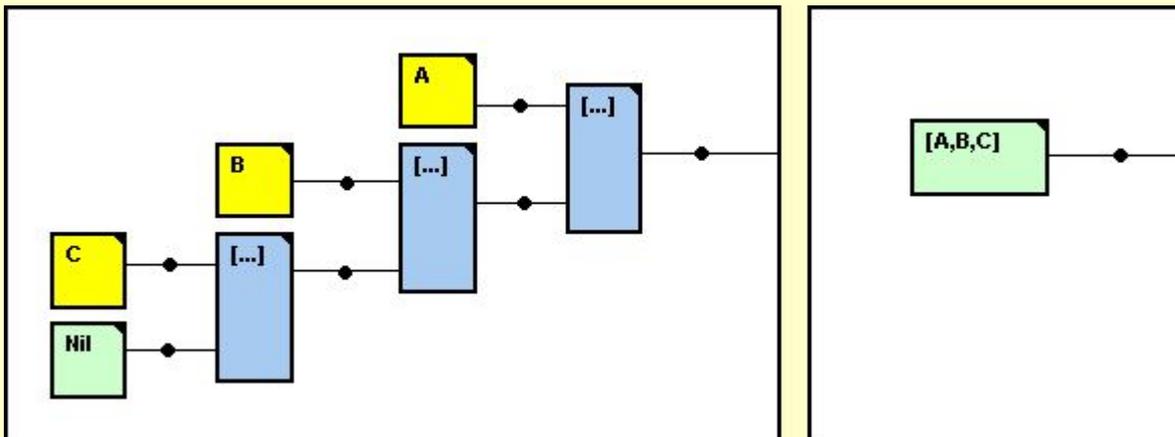
$TypedAdd = TNatAdd[Add]$



# Системные типы данных

- Натуральные числа (0, 1, 2, ...).
- Списки ([ ], [A(B), [D]]).
- Строки ('Some text').

Пример записи списка [A,B,C] в сетевой форме при помощи конструктора списка и системного генератора списков:



# Среда программирования

- Проектный подход к организации работы с комплексами программ.
- Оригинальные технологии графического и структурированного текстового построения программ.
- Интегрированные средства разработки и отладки.
- Собственное сетевое ядро вычислений и компилятор запросов на языке S-FLOGOL.
- Возможность ограниченного импорта программ, написанных на языке Пролог.
- Реализация в среде C++ Builder 5.0.

# Общий вид интерфейса системы

The screenshot displays the FLIDE software interface, which is used for developing and testing logic programs. It is divided into several main sections:

- Top Panel:** Contains the menu bar (Файл, Правка, Вид, Действия, Справка) and the title bar for the active window, "FLIDE - [Arithm.FLT]".
- Left Panel:** A tree view of "Конструкторы" (Constructors) and "Правила" (Rules). Under "Правила", various rules are listed, including "Add.1", "Add.2", "Mult.1", "Mult.2", "Nat0", "Nat2", "Nat3", "Power.1", and "Power.2".
- Center Panel:** A text editor showing the source code of a module named "Arithm". The code defines several rules and constants, such as:

```
MODULE Arithm=  
  (0:1)Null;  
  (1:1)Succ;  
  Nat3={:Succ(Succ(Succ(Null)))};  
  Nat2={:Succ(Succ(Null))};  
  Nat0={:Null};  
  Add={Null,x:x};  
  Add={Succ(x),y:Succ(@x,y)};  
  Mult={Null,x:Null};  
  Mult={Succ(x),y:Add(@x,y)};  
  Power={x,Null:Succ(Null)};  
  Power={x,Succ(y):Mult(x,@x,y)};  
  QAdd={:Add(Nat2,Nat3)};  
  Mult={:Mult(Nat2,Nat3)}.
```
- Bottom Panel:** A graphical editor for "Правила" (Rules). It shows a flow diagram with three main nodes: "Succ" (green), "Mult" (purple), and "Add" (yellow). Lines connect these nodes, illustrating the execution flow of a rule. The "Succ" node feeds into the "Mult" node, which in turn feeds into the "Add" node.
- Right Panel:** A project browser showing a hierarchy of modules under "Demo". The categories are "Сетевые модули" (Network modules), "Дедуктивные модули" (Deductive modules), and "Текстовые модули" (Textual modules). Specific modules listed include "Arithm", "ColorMap", "Hanoi", "Lists", "Queens", "Triggers", "Chang&Yong", "Simple", "SteamRoller", "Arithm", "ColorMap", "Hanoi", "Lists", "Queens", "RelComp", and "Triggers".

# Графический редактор

The screenshot displays the FLIDE graphical editor interface. The title bar reads "FLIDE - [Hanoi.FLN]". The top toolbar includes icons for file operations, editing, and execution, with a dropdown menu currently set to "append". The left sidebar shows a project tree with folders for "Конструкторы" (Constructors), "Константы" (Constants), "Системные" (System), "Правила" (Rules), "Запросы" (Queries), "Результат" (Result), and "Отладка" (Debug). Under "Правила", the rule "HANOI.2" is selected. The main workspace contains a flowchart on a grid background. The flowchart starts with a "Succ" node (green) leading to a "HANOI 1" node (pink). From "HANOI 1", the flow goes to a "to" node (blue), then to a "HANOI 2" node (pink), then to an "[...]" node (blue), and finally to an "append 3" node (yellow). The bottom status bar shows the execution progress: "Начало вычисления..." (Start of calculation...), "Результат 1:" (Result 1:), "Выход 1: [to(A,C),to(A,B),to(C,B)]" (Output 1: [to(A,C),to(A,B),to(C,B)]), and "Вычисление завершено за 0.005 секунд (9 шагов)" (Calculation completed in 0.005 seconds (9 steps)).



**Спасибо за внимание!**