**ПРОЛОГ**

**Факториал числа**

**Рекурсия вверх:**

fact\_up(0, 1) :– !.

fact\_up(N, X) :– N>0, N1 is N-1,

fact\_up(N1, X1),

X is N\*X1.

**Рекурсия вниз:**

С её помощью память сжирается меньше, т.к. подсчет происходит до вызова

(Когда попадаем в дно, у нас уже есть ответ, который потом прокидываем вверх)

fact\_down(N, X) :– fact\_down\_acc(N, 1, X).

fact\_down\_acc(0, Acc, Acc) :– !.

fact\_down\_acc(N, Acc, X) :– N>0, N1 is N-1,

AccTemp is N\*Acc,

fact\_down\_acc(N1, AccTemp, X).

**Сумма цифр числа**

**Рекурсия вверх:**

digit\_sum(0, 0) :– !.

digit\_sum(X, Sum) :– X1 is X // 10,

Ost is X mod 10,

digit\_sum(X1, Sum1),

Sum is Sum1 + Ost**.**

**Рекурсия вниз:**

digit\_sum\_down(X, Sum) :– digit\_sum\_down(X, 0, Sum).

digit\_sum\_down(0, SumCur, SumCur) :– !.

digit\_sum\_down(X1, SumCur, Sum) :–

X2 is X1 // 10,

Ost is X1 mod 10,

SumNew is SumCur + Ost,

digit\_sum\_down(X2, SumNew, Sum).

**Сумма элементов списка Черча**

**Рекурсия вверх:**

sum\_el([ ], 0) :– !.

sum\_el([H | T], Sum) :– sum\_el(T, SumT),

Sum is H + SumT.

**Рекурсия вниз:**

sum\_el\_down(List, Sum) :– sum\_el\_down(List, 0, Sum).

sum\_el\_down([ ], Sum, Sum) :– !.

sum\_el\_down([H | T], CurSum, Sum) :– NewSum is CurSum + H,

sum\_el\_down(T, NewSum, Sum).

**Числа Фибоначчи**

**Рекурсия вверх:**

fib(0, 0).

fib(1, 1).

fib(N, Res) :– N > 1,

N1 is N - 1,

N2 is N - 2,

fib(N1, Res1), fib(N2, Res2),

Res is Res1 + Res2.

**Рекурсия вниз:**

fib(0, 0).

fib(N, Res) :– N > 0,

fib(N, 0, 1, Res).

fib(1, \_ , Acc, Acc).

fib(N, Val, Acc, Res) :– N1 is N-1,

AccNew is Val + Acc,

fib(N1, Acc, AccNew, Res).

**Совершенное число**

**Рекурсия вверх:**

sum\_del(N, Sum) :– sum\_del\_help(N, 1, 0, Sum).

sum\_del\_help(N, Del, SumSoFar, Sum) :– Del > N, Sum is SumSoFar.

sum\_del\_help(N, Del, SumSoFar, Sum) :–

Del =< N,

0 is N mod Del,

NewSumSoFar is SumSoFar + Del,

NextDel is Del + 1,

sum\_del\_help(N, NextDel, NewSumSoFar, Sum).

perfect\_number(N) :– sum\_del(N, N).

**Рекурсия вниз:**

perfect(N) :– perfect(N, 1, 0).

perfect(1, \_ , 0).

perfect(N, D, Sum) :– D < N,

0 is N mod D,

NewSum is Sum + D,

NewD is D + 1,

perfect(N, NewD, NewSum).

perfect(N, D, Sum) :– D >= N,

Sum = N.

**Простое число**

isDiv(Num, Div) :– Num mod Div is 0, !.

isDiv(Num, Div) :– Div \* Div <= Number,

NextDiv = Div + 1,

isDiv(Num, NextDiv).

isPrime(1) :– !.

isPrime(Num) :– Num > 0,

not(isDiv(Num, 2)).

**Максимум из 3-ёх чисел**

max(X, Y, U, Z) :– TempMax is max(X, max(Y, U)),

Z = TempMax.

**Максимальная цифра в числе**

**рекурсия вниз:**

max\_digit\_down(Num, MaxDigit) :– max\_digit\_down\_help(Num, 0, MaxDigit).

max\_digit\_down\_help(0, CurMax, CurMax).

max\_digit\_down\_help(Num, CurMax, MaxDigit) :–

Num > 0,

NextDigit is Num mod 10,

NextNum is Num div 10,

NewMax is max(CurMax, NextDigit),

max\_digit\_down\_help(NextNum, NewMax, MaxDigit).

**Произведение цифр числа**

digit\_prod(Num, Pr) :– Num > 0,

NextDigit is Num mod 10,

ResNum is Num // 10,

digit\_prod(ResNum, ResPr),

Pr is NextDigit \* ResPr.

**Реализация append()**

append([ ], List2, List2).

append([Head | Tail], List2, [Head | TailRes]) :– append(Tail, List2, TailRes).

**Реализация reverse()**

reverse(List, Res) :– reverse(List, [ ], Res).

reverse([ ], Buffer, Buffer) :– !.

reverse([Head | Tail], Buffer, Res) :– reverse(Tail, [Head | Buffer], Res).

**Реализация nth0()**

nth0(0, [Elem | \_Tail], Elem) :– !.

nth0(Index, \_List, \_Elem) :– Index < 0, !, fail.

nth0(Index, [ \_Head | Tail], Elem) :– NextIndex is Index – 1,

nth0(NextIndex, Tail, Elem).

**РАЗМЕЩЕНИЯ**

**С повторениями:**

comb( \_, 0, [ ]).

comb(List, K, [X **|** Comb]) :– K > 0,

append( \_ , [X **|** T] , List),

K1 is K – 1,

comb(List, K1, Comb).

**Без повторений:**

permute([ ], [ ]).

permute(List, [H **|** Perm]) :– select(H, List, Rest),

permute(Rest, Perm).

**СОЧЕТАНИЯ**

**С повторениями:**

comb( \_ , 0, [ ]). // чтобы было без повторений ( \_ заменить на [ ])

comb([H | T], K, [H | R]) :– K > 0,

K1 is K – 1,

comb(T, K1, R).

comb([ \_ | T], K, R) :– K>0,

comb(T, K, R).

**ПЕРЕСТАНОВКИ**

**С повторениями:**

permute( [ ], [ ] ).

permute( [X | Xs], Ys) :– permute(Xs, Zs),

insert(X, Zs, Ys).

insert(X, Ys, Zs) :– del(X, Zs, Ys).

del(X, [X | Ys], Ys).

del(X, [Z | Ys], [Z | Zs]) :– del(X, Ys, Zs).

**Без повторений:**

permute( [ ], [ ] ).

permute(List, [X | Perm]) :– del(X, List, Rest),

permute(Rest, Perm).

del(X, [X | Tail], Tail).

del(X, [Y | Tail], [Y | Rest]) :– del(X, Tail, Rest).

ЛЯМБДА

<https://neerc.ifmo.ru/wiki/index.php?title=%D0%9B%D1%8F%D0%BC%D0%B1%D0%B4%D0%B0-%D0%B8%D1%81%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5>