

# Tính giai thừa

domains

i = integer

predicates

factor(i, i)

clauses

factor(0, 1) :- !.

factor(N, KQ) :- N1=N-1, factor(N1, KQ1), KQ1\*N=KQ.

# Tính lũy thừa

domains

i = integer

predicates

LuyThua(i, i, i)

clauses

LuyThua(\_, 0, 1) :- !.

LuyThua(X, N, KQ) :- N1 = N-1,

LuyThua(X, N1, KQ1),

X\*KQ1 = KQ.

# Tổ hợp chập k của n phần tử

domains

i = integer

predicates

combinatorial(i, i, i)

clauses

combinatorial(0, \_, 1) :- !.

combinatorial(K, K, 1) :- !.

combinatorial(K, N, KQ) :- N1=N-1, K1=K-1,

combinatorial(K, N1, KQ1),

combinatorial(K1, N1, KQ2),

KQ = KQ1+KQ2.

# Fibonacci thứ n

domains

i = integer

predicates

fibonacci(i, i)

clauses

fibonacci(1, 1) :-!.

fibonacci(2, 1) :-!.

fibonacci(N, KQ) :- N1 = N-1, N2= N-2,

fibonacci(N1, KQ1),

fibonacci(N2, KQ2),

KQ = KQ1 + KQ2.

# Phép chia lấy nguyên

*int DivideWithoutRemainder(int divisor, int dividend)*

*{*

*int quotient = 0;*

*do*

*{*

*quotient++;*

*if (quotient \* dividend ≤ divisor &&*

*(quotient + 1) \* dividend > divisor)*

*return quotient;*

*} while (true);*

*}*

domains

i = integer

predicates

Sinh\_1\_So(i)

ThuongNguyen(i, i, i)

clauses

Sinh\_1\_So(0).

Sinh\_1\_So(Sau) :- Sinh\_1\_So(Truoc), Sau = Truoc + 1.

ThuongNguyen(BC, C, KQ) :- Sinh\_1\_So(KQ),

KQ\*C <= BC,

(KQ+1) \* C > BC, !.

# Tổng các phần tử trong danh sách

domains

i = integer

ds = i\*

predicates

Tong(ds, i)

clauses

Tong([], 0) :- !.

Tong([H|T], KQ) :- Tong(T, KQ1),

KQ1 + H = KQ.

# Độ dài các phần tử trong danh sách

domains

i = integer

ds = i\*

predicates

DoDai(ds, i)

clauses

DoDai([], 0).

DoDai([\_|T], KQ) :- DoDai(T, KQ1), KQ1 + 1 = KQ.

# Nối 2 danh sách số nguyên

domains

i = integer

ds = i\*

predicates

Noi(ds, ds, ds)

clauses

Noi([], L2, L2).

Noi([H1|T1], L2, [H1|T3]) :- Noi(T1, L2, T3).

# Tìm các hoán vị của một danh sách số nguyên

domains

i = integer

ds = i\*

predicates

Noi(ds, ds, ds)

HoanVi(ds, ds)

clauses

Noi([], L2, L2).

Noi([H1|T1], L2, [H1|T3]) :- Noi(T1, L2, T3).

HoanVi([], []).

HoanVi(L, [H|DuoiHV]) :-

Noi(Gi, [H|T], L), !,

Noi(Gi, T, ConLai),

HoanVi(ConLai, DuoiHV).