Seminar 4 – Backtracking in Prolog

**Enuntul problemei:**

Se da un sir de numere distincte.

Se cere sa se genereze toate subsirurile sirulu dat, avand aspect de vale.

**Exemplu:**

**L = {28, 7, 8, 3, 42, 10, 16}**

**=> [28, 7, 8], [42, 16, 28], [8, 3, 10], [16, 8, 3, 10] …**

**[10] flag = -1**

**[3, 10] flag = -1**

**[8, 3, 10] flag = 1**

**[16, 8, 3, 10] flag = 1 ...**

Candidat(l1l2..ln) = 1. l1, daca L nu e vida

2. candidat(l2l3...ln), daca n >= 2

% flag = 1 crescatoare

% Flag = -1 descrescatoare

Wrapper(l1 l2 .. ln) = Vale(l1 l2 .. ln, [e1,e2], -1) e1=candidat (l1...ln), e2 = candidac(l1...ln), e1 < e2

Candidat([H|\_],H).

Candidat([\_|T],E):-candidat(T,E).

% Vale(L1: lista, Col: Lista, Flag: Intreg, Rezultat: Lista)

% (I, I, I, o) nedeterminist

Vale(l1l2..ln, c1c2..cm, flag) = 1. c1c2..cm, daca flag = 1

2. Vale(l1l2..ln, cand + c1c2..cm, -1 ), daca flag = -1 si

cand = candidat(l1l2..ln) si cand < c1

3. Vale(l1l2..ln, cand + c1c2..cm, 1), daca

Cand = candidat(l1l2..ln) si cand > c1 si ∄ candidat(c1...cm) = cand

Vale(\_, Col, Flag, Col) :-

Flag is 1.

Vale(X, [H | T], Flag, Col) :-

Flag is –1,

Candidat(X, Y),

H > Y,

Vale(X, [ Y | [ H | T ]], -1, Col).

Vale(X, [H | T], Flag, Col) :-

Candidat(X, Y),

Y> H,

Not(Candidat([H | T], Y)),

Vale(X, [ Y | [ H | T ]], 1, Col).

Wrapper(l1 l2 .. ln) = Vale(l1 l2 .. ln, [e1,e2], -1) e1=candidat (l1...ln), e2 = candidat(l1...ln), e1 < e2

Wrapper (L, R) :- Candidat(L, C1),

Candidat(L, C2),

C = [C1,C2],

C1 < C2,

Vale(L, C, -1, R).

Impar(1).

Impar(3).

Par(2).

Par(4).

ParImpar(X,Y):- impar(X), par(Y).

ParImpar(X,Y):-par(X), impar(Y), !.

=> (1,2), (1, 4), (3, 2), (3, 4), (2,1), (2,3), (4, 1), (4, 3)

=> (1,2), (1, 4), (3, 2), (3, 4)

=> (1,2), (1, 4)

=> (1,2)

=> (1,2), (1, 4), (3, 2), (3, 4), (2,1), (2,3), (4, 1), (4, 3)

=> (1,2), (1, 4), (3, 2), (3, 4), (2,1), (2,3)

=> (1,2), (1, 4), (3, 2), (3, 4), (2,1)