**I Laborator 3 PLF P3**

**II** **Utilizarea facilității de backtracking în Prolog**

**III Predat la:** 11.11.2021

**IV Problema 15**

Se dă un număr *n* pozitiv. Se cere să se determine toate descompunerile lui *n* ca sumă de numere naturale consecutive.

**V Model matematic recursiv**

Fie *n* un număr natural pozitiv. Metoda list creează o listă de la 1 la *n*( [1, 2, 3, …, n]). Acestă listă este trimisă funcției submultimi, care realizează toate submulțimile listei noastre. Fiecare submulțime în parte este verificată dacă îndeplinește criteriul cerut prin metoda consecutive. Dacă da, se face suma elementelor prin funcția suma. Am găsit un răspuns valid dacă suma rezultată este egală cu *n* inițial.

lista(n, c, L)=

submultimi([a1,a2, ..., an], Rez)=

consecutive([a1,a2, ..., an]) =

suma([a1,a2, ..., an], S) =

**VI Cod sursă**

domain

N = integer

S = integer

Rez = N\*

predicates

lista(N, 1, Rez1) %(i, i, o)

submultimi(Rez1, Rez2) %(i, o)

consecutive(Rez2) %(i)

suma(Rez2, S) %(i, o)

main(N, Rez) %(i, o)

Rez =:= S

clauses

lista(N, N, [N]) :- !.

lista(N, I, [I|Rez]) :-

I1 is I + 1,

lista(N, I1, Rez).

submultimi([], []).

submultimi([\_|T], Rez) :-

submultimi(T, Rez).

submultimi([H|T], [H|Rez]) :-

submultimi(T, Rez).

consecutive([]) :- true, !.

consecutive([\_]) :- true, !.

consecutive([H1, H2|T]) :-

H1 =:= H2 - 1,

consecutive([H2|T]).

suma([], 0).

suma([H|T], Rez) :-

suma(T, Rez1),

Rez is H + Rez1.

main(N, Rez) :-

lista(N, 1, List),

submultimi(List, Rez),

consecutive(Rez),

suma(Rez, Sum),

Sum =:= N.

**VII Exemple de testare**

Cazurile sunt scrise de forma(date de intrare=>rezultat).

n = 3 => main(3, Rez) = [3] && [1, 2]

lista(3, 1, Rez) = [1, 2, 3]

submultimi([1, 2, 3], Rez) =>

Rez = [] ;

Rez = [3] ;

Rez = [2] ;

Rez = [2, 3] ;

Rez = [1] ;

Rez = [1, 3] ;

Rez = [1, 2] ;

Rez = [1, 2, 3].

consecutive(Rez) => true, doar pentru: [1, 2, 3], [] ,[3], [2], [1], [2, 3], [1, 2]

suma(Rez, Sum) =>

Sum = 6;

Sum = 0;

Sum = 3;

Sum = 2;

Sum = 1;

Sum = 5;

Sum = 3.

În final trebuie să avem Sum = N, adică doar în cazurile Rez = [3] și Rez = [1, 2] =>

main(3, Rez) = [3] && [1, 2].