Data: 24.01.2017

Examen Programare Logica
Sesiunea: Iarna 2017

Varianta B

Partea I (20 min) **Nume:**

Sa se scrie predicatele Prolog pentru:

1. Inserarea unui element intr-un arbore binar de *cautare.*
2. Concatenarea eficienta a trei liste. Motivati eficienta (maxim o fraza).
3. Cautarea unui element intr-o lista *incompleta*.
4. Calcularea sumei elementelor unei liste, folosind recursivitate inainte.

Partea a II-a (40 min) **Nume:**

1. Predicatul de mai jos ar trebui sa inlocuiasca toate aparitiile unui element *Old* dintr-o lista, cu elementul *New*. Este corect? Daca nu, efectuati corecturile necesare pentru a obtine un predicat corect.

```
r(_,_,[],[]).
r(H,N,[H|T],[N|T]).
r(X,N,[H|T],R):-r(X,N,T,R).

?- r(1,a,[1,2,3,1,4,2],R).
R=[a,2,3,a,4,2];
no.
```

2. Predicatele de mai jos sorteaza o lista prin inserare. Ce se afiseaza pe ecran in urma executiei intrebarii date? Este predicatul de sortare stabil? Daca nu, efectuati modificarile necesare pentru ca acesta sa devina stabil.

```
ins_sort([H|T],R,Res):- insert_ord(H,R,R1),
                        writeln(R1),
                        ins_sort(T,R1,Res).
ins_sort([],R,R).

insert_ord(X,[H|T],[H|R]):- X>H,
                        !,
                        insert_ord(X,T,R).
insert_ord(X,T,[X|T]).

?- ins_sort([4,7,2,1,4,3,2], [], Res).
```

3. Se da o lista de elemente atomice. Scrieti predicatul(ele) care genereaza toate sub-listele de lungime K (data ca argument).

```
E.g. ?- all_subls([1,2,3,4,5,6], 3, L).
     L=[[1,2,3], [2,3,4], [3,4,5], [4,5,6]];
     no
```

4. Se da un arbore binar de cautare. Scrieti unul (sau mai multe) predicat(e) eficient(e) care verifica daca arborele este Perfect Echilibrat (i.e. diferenta dintre numarul de noduri ale subarborelui stang si drept este de cel mult 1, la orice nod din arbore).

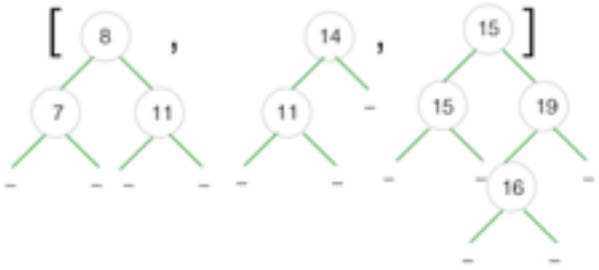
```
E.g. ?- isPBT(t(3, t(2,t(1,nil,nil),nil), t(5,t(4,nil,nil),nil))).
     yes
     ?- isPBT(t(3, t(2,t(1,nil,nil),nil), nil)).
     no
```

Data: 24.01.2017

Varianta B
Partea a III-a (60 min) **Nume:**

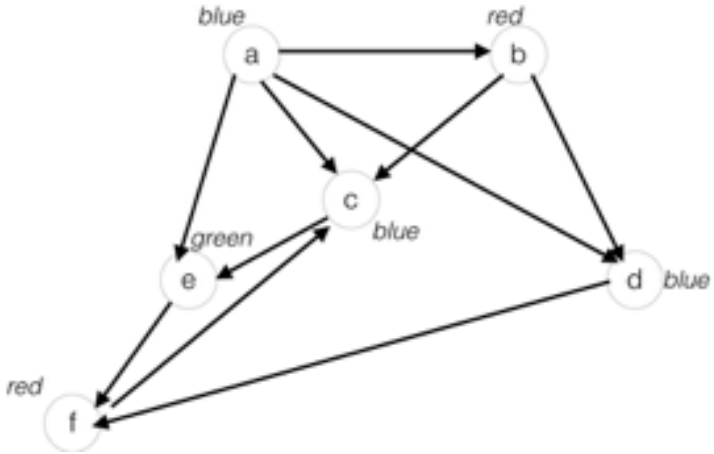
1. Se da o lista ***ordonata***, ale carei elemente sunt arbori ***binari de cautare incompleti***. Se cere sa se scrie secventa de predicate care:
   a. Cauta dupa prima (si doar prima) aparitie a unui element atomic dat in structura complexa (e.g. pentru al treilea arbore din structura, prima aparitie a elementului atomic *15* este cea din radacina)
   b. Colecteaza toate elementele atomice aflate in *noduri frunza* in toti arborii din structura data, intr-o lista ordonata (varianta cu append - lista completa, apoi varianta cu liste diferenta) (e.g. pt exemplul dat, ar trebui colectate cheile: 7, 11, 11, 15 si 16).

   Exemplu de structura:



2. Se da un graf ale carui noduri au asociate culori. Sa se scrie secventa de predicate care gaseste toate caile intre un nod *Sursa* si un nod *Destinatie* care **nu** trec prin noduri intermediare de o anumita *Culoare* (data).

```
?- findAllPathsNoColor(a,f,red,Paths).
Paths=[[a,c,e,f],[a,d,f],[a,e,f]]
```



Nota: Predicatele `member` si `append` pe liste complete sunt considerate cunoscute (nu trebuie sa le scrieti). Pentru problema pe graf, se cere sa specificati si ce reprezentare utilizati (nu este impusa).