INTELIGENTA ARTIFICIALA

Teme de casa

11 aprilie 2011

Scopul acestei teme este efectuarea unor prelucrari asupra unor structuri simbolice reprezentate sub forma unor termeni Prolog.

Pentru realizarea acestei teme va trebui sa urmati cu strictete urmatoarele indicatii:

i) Drept raspuns va trebui sa furnizati doua fisiere: scenariu.pro si proiect.pro.
ii) Fisierul scenariu.pro va contine datele de test pentru testarea programului dumneavoastra. Veti furniza date de test pentru fiecare punct al problemei urmand intocmai modelul din fisierul scenariu.pro furnizat ca exemplu. Pentru fiecare punct veti pregati minim 5 seturi de date de test, din care cel putin un test cu valori incorecte (vezi exemplul).
iii) Fisierul proiect.pro va contine codul dumneavoastra. Este foarte important ca acest cod sa fie cat mai bine documentat si cat mai ingrijit scris.
iv) Pentru testarea codului dumneavoastra se vor consulta intai fisierele proiect.pro si apoi scenariu.pro, in aceasta ordine.
v) Orice forma de plagiat va fi depunctata drastic. In consecinta, incercati sa scrieti dumneavoastra codul si datele de test, nu sa copiati aceste chestiuni de la colegi.

Problema 1. Pentru reprezentarea matricelor cu puţine elemente diferite de zero se foloseşte următoarea formă: `matriceRara(NrLinii, NrColoane, ListaDeElemente)`, unde `ListaDeElemente` este o listă de elemente de forma `elem(Linie, Coloană, Valoare)`. De exemplu, matricea:

$$\begin{Bmatrix} 0 & 34 & 0 & 0 \\ 67 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & -1 \end{Bmatrix}$$

Se reprezintă:

matriceRara(4, 4, [elem(1, 2, 34), elem(2, 1, 67), elem(4, 1, 5), elem(4, 4, -1)])

Se cere:
i) Să se definească un predicat esteMatriceRară(M) care este adevărat dacă şi numai dacă M este un termen ce reprezintă o matrice rară.
ii) Să se definească un predicat add(A, B, C) care este adevărat dacă şi numai dacă C este matricea rară obţinută prin adunarea matricelor rare A şi B.

iii) Să se definească un predicat multiply(A, B, C) care este adevărat dacă şi numai dacă C este matricea rară obţinută prin înmulţirea matricelor rare A şi B..
iv) Să se definească un predicat eliminăLinie(M, Linie, R) care este adevărat dacă şi numai dacă matricea rară R este matricea obţinută din matricea rară M prin eliminarea liniei Linie.

Problema 2. Pentru reprezentarea matricelor cu puţine elemente diferite de zero se foloseşte următoarea formă: `matriceRara(NrLinii, NrColoane, ListaDeLinii)`, unde `ListaDeLinii` este o listă cu elemente de forma `linie(Linie, ListaValori)`, `ListăValorii fiind o listă numere de forma [Coloana, Valoare, Coloana, Valoare, …]`. De exemplu, matricea:

$$\begin{Bmatrix} 0 & 34 & 0 & 0 \\ 67 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & -1 \end{Bmatrix}$$

Se reprezintă:

matriceRara(4, 4, [linie(1, [2, 34]), linie(2, [1, 67]), linie(4, [1, 5, 4, -1])])

Se cere:
i) Să se definească un predicat esteMatriceRară(M) care este adevărat dacă şi numai dacă M este un termen ce reprezintă o matrice rară.
ii) Să se definească un predicat add(A, B, C) care este adevărat dacă şi numai dacă C este matricea rară obţinută prin adunarea matricelor rare A şi B.
iii) Să se definească un predicat multiply(A, B, C) care este adevărat dacă şi numai dacă C este matricea rară obţinută prin înmulţirea matricelor rare A şi B.
iv) Să se definească un predicat eliminăColoană(M, Coloană, R) care este adevărat dacă şi numai dacă matricea rară R este matricea obţinută din matricea rară M prin eliminarea coloanei Coloană.

Problema 3. Pentru reprezentarea matricelor cu puţine elemente diferite de zero se foloseşte următoarea formă: `matriceRara(NrLinii, NrColoane, ListaDeColoane)`, unde `ListaDeColoane` este o listă cu elemente de forma `coloana(Coloana, ListaValori)`, `ListăValorii fiind o listă numere de forma [Linie, Valoare, Linie, Valoare, …]`. De exemplu, matricea:

$$\begin{Bmatrix} 0 & 34 & 0 & 0 \\ 67 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & -1 \end{Bmatrix}$$

Se reprezintă:

matriceRara(4, 4, [coloana(1, [2, 67, 4, 5]), coloana(2, [1, 34]), coloana(4, [4, -1])])

Se cere:
i) Să se definească un predicat esteMatriceRară(M) care este adevărat dacă şi numai dacă M este un termen ce reprezintă o matrice rară.
ii) Să se definească un predicat add(A, B, C) care este adevărat dacă şi numai dacă C este matricea rară obţinută prin adunarea matricelor rare A şi B.
iii) Să se definească un predicat multiply(A, B, C) care este adevărat dacă şi numai dacă C este matricea rară obţinută prin înmulţirea matricelor rare A şi B..
iv) Să se definească un predicat eliminăLinie(M, Linie, R) care este adevărat dacă şi numai dacă matricea rară R este matricea obţinută din matricea rară M prin eliminarea liniei Linie.

Problema 4. Pentru reprezentarea matricelor cu puţine elemente diferite de zero se foloseşte următoarea formă: `matriceRara(NrLinii, NrColoane, ListaDeLinii)`, unde `ListaDeLinii` este o listă cu elemente de forma `linie(Linie, ListaValori)`, `ListăValorii fiind o listă numere de forma [Coloana, Coloana, Valoare, Valoare, …]`. De exemplu, matricea:

$$\begin{Bmatrix} 0 & 34 & 0 & 0 \\ 67 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & -1 \end{Bmatrix}$$

Se reprezintă:

matriceRara(4, 4, [linie(1, [2, 34]), linie(2, [1, 67]), linie(4, [1, 4, 5, -1])])

Se cere:
i) Să se definească un predicat esteMatriceRară(M) care este adevărat dacă şi numai dacă M este un termen ce reprezintă o matrice rară.
ii) Să se definească un predicat add(A, B, C) care este adevărat dacă şi numai dacă C este matricea rară obţinută prin adunarea matricelor rare A şi B.
iii) Să se definească un predicat multiply(A, B, C) care este adevărat dacă şi numai dacă C este matricea rară obţinută prin înmulţirea matricelor rare A şi B.
iv) Să se definească un predicat eliminăColoană(M, Coloană, R) care este adevărat dacă şi numai dacă matricea rară R este matricea obţinută din matricea rară M prin eliminarea coloanei Coloană.

Problema 5. Pentru reprezentarea matricelor cu puţine elemente diferite de zero se foloseşte următoarea formă: `matriceRara(NrLinii, NrColoane, ListaDeColoane)`, unde `ListaDeColoane` este o listă cu elemente de forma `coloana(Coloana, ListaValori)`, `ListăValorii fiind o listă numere de forma [Linie, Linie, Valoare, Valoare, …]`. De exemplu, matricea:

$$\begin{Bmatrix} 0 & 34 & 0 & 0 \\ 67 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & -1 \end{Bmatrix}$$

Se reprezintă:

matriceRara(4, 4, [coloana(1, [2, 4, 67, 5]), coloana(2, [1, 34]), coloana(4, [4, -1])])

Se cere:
i) Să se definească un predicat esteMatriceRară(M) care este adevărat dacă şi numai dacă M este un termen ce reprezintă o matrice rară.
ii) Să se definească un predicat add(A, B, C) care este adevărat dacă şi numai dacă C este matricea rară obţinută prin adunarea matricelor rare A şi B.
iii) Să se definească un predicat multiply(A, B, C) care este adevărat dacă şi numai dacă C este matricea rară obţinută prin înmulţirea matricelor rare A şi B..
iv) Să se definească un predicat eliminăLinie(M, Linie, R) care este adevărat dacă şi numai dacă matricea rară R este matricea obţinută din matricea rară M prin eliminarea liniei Linie.

Problema 6. Pentru reprezentarea unui polinom de mai multe variabile se foloseşte următorul predicat: polinom(ListăVariabile, ListăTermeni). ListăVariabile este o listă care conţine variabilele polinomului iar ListăTermeni este o listă de forma: termen(Coeficient, ListăExponenţi). ListăExponenţi este o listă cu exponenţii variabilelor din termenul respectiv. De exemplu, polinomul:

$$p(x,y,z) = 120x^7y^3 + 34.5y^5z^2 + 44x^4y^2z$$

se reprezintă: polinom([x,y,z], [termen(120,[7, 3, 0]), termen(34.5, [0, 5, 2]), termen(44, [4, 2, 1])]).

Se cere:
i) Să se scrie un predicat estePolinom(P) care este adevărat dacă şi numai dacă P este un polinom.
ii) Să se scrie un predicat eval(P, Asignari, R) care este adevărat dacă şi numai dacă R este polinomul care se obţine prin înlocuirea variabilelor din lista de asignări. Asignari este o lista de forma: [Variabila, Valoare, Variabila, Valoare, …].
iii) Să se scrie un predicat adună(A, B, C) care este adevărat dacă şi numai dacă C este polinomul obţinut prin adunarea polinoamelor A şi B.
iv) Să se scrie un predicat multiply(A, B, C) care este adevărat dacă şi numai dacă C este polinomul obţinut prin înmulţirea polinoamelor A şi B.

Problema 7. Pentru reprezentarea unui polinom de mai multe variabile se foloseşte următorul predicat: polinom(ListăVariabile, ListăTermeni). ListăVariabile este o listă care conţine variabilele polinomului iar ListăTermeni este o listă de forma: termen(Coeficient, ListăExponenţi). ListăExponenţi este o listă cu elemente de forma exponent(Variabila, Exponent). De exemplu, polinomul:

$$p(x,y,z) = 120x^7y^3 + 34.5y^5z^2 + 44x^4y^2z$$

se reprezintă: polinom([x,y,z], [termen(120,[exponent(x,7), exponent(y,3)]), termen(34.5, [exponent(y, 5), exponent(z, 2)]), termen(44, [exponent(x,4), exponent(y,2), exponent(z,1)])]).

Se cere:
i) Să se scrie un predicat estePolinom(P) care este adevărat dacă şi numai dacă P este un polinom.
ii) Să se scrie un predicat eval(P, Asignari, R) care este adevărat dacă şi numai dacă R este polinomul care se obţine prin înlocuirea variabilelor din lista de asignări. Asignari este o lista de forma: [Variabila, Valoare, Variabila, Valoare, …].
iii) Să se scrie un predicat adună(A, B, C) care este adevărat dacă şi numai dacă C este polinomul obţinut prin adunarea polinoamelor A şi B.
iv) Să se scrie un predicat multiply(A, B, C) care este adevărat dacă şi numai dacă C este polinomul obţinut prin înmulţirea polinoamelor A şi B.

Problema 8. Pentru reprezentarea unei librării se foloseşte un termen de forma: librarie(ListaAutori, Sold), unde Sold reprezintă contravaloarea cărţilor vândute, iar ListaAutori este o lista de termeni de forma autor(Nume, DataNasterii, DataDecesului, ListaCarti). DataNasterii este un termen de forma data(Zi, Luna, An) sau *necunoscută* , DataDecesului este un termen de aceeaşi formă sau "-" dacă autorul este încă în viaţă. ListaCarti este o listă de cel puţin un termen de forma carte(Titlu, ISBN, Categorie, Pret, Exemplare). Categorie e una din: beletristica, tehnică, filozofie, psihologie. Exemplare nu poate fi decât un număr întreg strict pozitiv.

Se cere:
i) Să se scrie un predicat esteLibrarie(L) care este adevărat dacă şi numai dacă L este un termen care desemnează o librărie.
ii) Să se scrie un predicat vindeCarte(L, ISBN, R) unde R este starea în care se transformă librăria L prin vânzarea cărţii cu având ISBN. În urma vânzării soldul librarie creşte cu valoarea cărţii iar numărul de exemplare scade cu unul. Dacă în urma vânzării nu mai rămâne nici un exemplar cartea se şterge. La fel şi autorul rămas fără nici o carte.
iii) Să se scrie un predicat listaTitluri(L, Categorie, Titluri) care este adevărat dacă şi numai dacă şi numai dacă Titluri este lista cu titlurile tuturor cărţilor din librăria L care sunt din categoria Categorie.
iv) Să se scrie un predicat listaAutori(L, Categorie, Autori) care este adevărat dacă şi numai dacă şi numai dacă Autori este lista cu autorii din librăria L care au o carte din categoria Categorie.

Problema 9. Pentru reprezentarea unei librării se foloseşte un termen de forma bookstore(ListăCărţi) unde ListaCarti este o listă de termeni carte(ListaAutori, ListaCategorii, Titlu, Exemplare, ISBN, Pret). ListaCategorii este o lista care conţine cel puţin un element din: fantasy, scifi, horror, history, policier, soap. ListaAutori este o lista de termeni de forma: autor(Nume, Prenume).

i) Să se scrie un predicat esteLibrarie(L) care este adevărat dacă şi numai dacă L este un termen care desemnează o librărie.

ii) Să se scrie un predicat primesteMarfa(Librarie, ListaCarti, R) care este adevărat dacă şi numai dacă R este librăria obţinută prin adăugarea la inventarul librăriei Librarie a listei de cărţi ListaCarti.
iii) Să se scrie un predicat listaTitluri(L, Categorie, Titluri) care este adevărat dacă şi numai dacă şi numai dacă Titluri este lista cu titlurile tuturor cărţilor din librăria L care sunt din categoria Categorie.
iv) Să se scrie un predicat listaAutori(L, Categorie, Autori) care este adevărat dacă şi numai dacă şi numai dacă Autori este lista cu autorii din librăria L care au o carte din categoria Categorie.

Problema 10. Pentru reprezentarea unei biblioteci se foloseşte un termen de forma library(NumarLocatii, ListaLocatii, ListaAbonati). ListaLocatii este o lista de termeni de forma locatie(NumarLocatie, Carte). NumarLocatie este un număr între 1 şi NumărLocaţii, identificând în mod unic o locaţie, în care se poate afla o singură carte la un moment dat. Carte este un termen de forma carte(NumeAutor, Titlu, ISBN, CodExemplar). ListaAbonati este o lista de termeni de forma abonat(NumeAbonat, CNP, Imprumuturi). Imprumuturi este o lista de termeni de forma imprumut(Carte, DataRetur). DataRetur este un termen de forma data(An, Luna, Zi).

Se cere:
i) Să se scrie un predicat esteBiblioteca(B) care este adevărat dacă şi numai dacă B este un termen care reprezintă o bibliotecă.
ii) Să se scrie un predicat împrumută(B, NumeAbonat, CNP, ISBN, DataCurenta, Termen, R) care este adevărat dacă şi numai dacă R este biblioteca B după ce i s-a împrumutat abonatului cartea având ISBN, pentru un număr de Termen zile. Abonatului nu i se mai împrumută cărţi dacă are cărţi neaduse după data de retur.
iii) Să se scrie un predicat returnează(B, CNP, CodExemplar, R) care este adevărat dacă şi numai dacă R este biblioteca B după ce un abonat identificat de CNP a returnat cartea identificată de CodExemplar.
iv) Să se scrie un predicat intarziati(B, Data, ListaAbonati) care este adevărat dacă şi numai dacă ListaAbonaţi conţine toţi abonaţii care au cărţi care trebuiau returnate înainte de Data. Abonaţii sunt reprezentaţi cu ajutorul unor fapte de forma abonat(NumeAbonat, CNP, Imprumuturi), dar lista de împrumuturi nu conţine decât cărţile nereturnate la timp.

Problema 11. Pentru reprezentarea datelor necesare unei biblioteci se foloseşte un termen de forma biblioteca(ListaCarti, ListaAbonati). ListaAbonati este o lista de termeni de forma abonat(CNP, Nume, Adresa). ListaCarti este o lista care conţine termeni de forma carte(Autor, Titlu) pentru cărţile neîmprumutate sau carte(Autor, Titlu, CNP, DataRetur) pentru cele împrumutate, unde DataRetur este un termen de forma data(zi, luna, an), iar CNP-ul este codul numeric personal al unui abonat din lista de abonati.
Se cere:
i) Să se scrie un predicat esteBiblioteca(B) care este adevărat dacă şi numai dacă B este un termen care defineşte o bibliotecă.
ii) Să se scrie un predicat abonează(B, CNP, Nume, Adresa, R) care este adevărat dacă şi numai dacă R este biblioteca B la care s-a mai abonat abonatul cu datele: CNP, Nume, Adresa.
iii) Să se scrie un predicat împrumută(B, Carte, CNP, R) care este adevărat dacă şi numai dacă R este biblioteca B după ce abonatului identificat de CNP i s-a imprumutat cartea Carte.
iv) Să se scrie un predicat întarziati(B, Data, ListaAbonati) care conţine lista tuturor abonaţilor care trebuiau sa returneze cărţi bibliotecii B înainte de data Data.

Problema 12. Pentru reprezentarea datelor necesare unei bănci se foloseşte un predicat cu următoarea formă: bancă(ListăClienţi). ListaClienti conţine termeni de forma: agentEconomic(NumeFirma, CodFiscal, Adresa, LimitaCredit, ListaOperatii) sau persoana(Nume, CNP, Adresa, ListaOperatii). ListaOperatii este o lista cu termeni de forma depunere(Suma, Data) sau extragere(Suma, Data). Data este un termen de forma data(Zi, Luna, An).
Se cere:
i) Să se scrie un predicat esteBanca(B) care este adevărat dacă şi numai dacă B este un termen care reprezintă datele unei bănci.
ii) Să se scrie un predicat adaugaClient(B, Client, R) unde R este banca B după adăugarea clientului Client. Client este agentEconomic(NumeFirma, CodFiscal, Adresa, LimitaCredit) sau persoana(Nume, CNP, Adresa). Nu pot exista doi clienti cu acelasi cod fiscal sau cu acelasi cnp. În cazul în care mai exista un client cu acelasi id (cod fiscal sau cnp) se considera ca se doreste modificarea celorlate date, fara stergerea operatiunilor efectuate de acestia.
iii) Să se scrie un predicat calculSold(B, ID, Data, Sold) care este adevărat dacă şi numai dacă Sold este soldul clientului bancii B, identificat de ID (cod fiscal sau cnp) la data Data. Depunerile se adună la sold, iar extragerile se scad.
iv) Să se scrie un predicat înregistrează0peratie(B, ID, Data, Suma, R) care este adevărat dacă şi numai dacă R este banca B după ce clientul identificat de ID a efectuat operatia definita de Suma si Data. Suma negativă identifică o extragere iar una strict pozitivă o depunere. Cele cu suma zero nu se înregistrează cum nu se înregistrează extragerile pentru agentii economici care ar depăşi limita lor de credit (credit este atunci când soldul este negativ) sau extragerile pentru persoane care ar duce la un sold negativ.

Problema 13. Pentru reprezentarea datelor necesare pentru evidenta profesorilor si studentilor dintr-o facultate intr-un anumit semestru se foloseste un termen de forma *facultate(ListaProfesori, ListaStudenti)* unde *ListaProfesori* este o lista de termeni de forma *profesor(Nume, CNP, CursuriPredate)*, iar *ListaStudenti* este o lista de termeni de forma *student(Nume, CNP, CursuriUrmate)*. *CursuriPredate* este lista de cursuri pe care le preda un profesor in semestrul respectiv; fiecare curs este identificat printr-un cod unic. Similar, *CursuriUrmate* este lista de cursuri pe care le urmeaza un student in semestrul respectiv.
Se presupune ca fiecare curs este predat de un singur profesor.

Un exemplu este urmatorul:

facultate(
[profesor('Ion Popescu', 123, ['IA', 'POO', 'VPE']),
profesor('Mihaela Georgescu', 145, ['CE', 'SDA'])],
[student('Anca Popescu', 235, ['IA', 'ACA', 'PLA', 'DCE'],
student('Mihai Ionescu', 117, ['DNAD', 'CN', 'SE'])])

Se cere:
i) Sa se defineasca un predicat *este_facultate(X)* care este adevarat daca si numai daca *X* este un termen ce reprezinta datele necesare pentru evidenta membrilor unei facultati.
ii) Sa se defineasca un predicat *adaugaMaterie(F1, F2, CNPProfesor, CodCurs)* care este adevarat daca si numai daca *F2* este facultatea *F1* dupa adaugarea cursului *CodCurs* in lista de cursuri predate de profesorul identificat prin *CNPProfesor*.

iii) Sa se defineasca un predicat *adaugaPersoana(F1, F2, Nume, CNP, ListaCursuri, Tip)*, care este adevarat daca si numai daca *F2* este facultatea *F1* dupa adaugarea persoanei cu numele *Nume* si identificatorul *CNP*. *Tip* reprezinta calitatea persoanei – *'Student'* sau *'Profesor'*. Daca *Tip* este *'Student'*, atunci *ListaCursuri* reprezinta lista de cursuri urmate de studentul respectiv in acel semestru; daca *Tip* este *'Profesor'*, atunci *ListaCursuri* reprezinta lista de cursuri predate de profesorul respectiv in acel semestru.
Trebuie verificat faptul ca nu pot exista doua persoane cu acelasi CNP – in cazul in care mai exista o persoana cu acelasi CNP, atunci se considera ca se doreste modificarea celorlalte date (nume, lista de cursuri, tip).
iv) Sa se defineasca un predicat *corespondenta(F, CNPStudent, NumeStudent, ListaProfesori)* care este adevarat daca si numai daca *ListaProfesori* este lista profesorilor din facultatea *F* care predau cursuri studentului *NumeStudent* in semestrul respectiv.

Problema 14. Pentru reprezentarea rezultatelor studentilor la examenele dintr-o sesiune se foloseste un termen de forma *sesiune(ListaStudenti)*, unde *ListaStudenti* este o lista de termeni de forma *student(Nume, CNP, ListaNote)*, iar *ListaNote* este o lista de termeni de forma *nota(Curs, Nota)*.

Un exemplu este urmatorul:

sesiune([student('Ion Popescu', 123, [nota('IA', 9), nota('ACA', 8), nota('PLA', 5)]),
student('Anca Georgescu', 144, [nota('IA', 6), nota('ACA', 5), nota('PLA', 8)]),
student('Dan Marin', 187, [nota('DAR', 9), nota('VPE', 10), nota('SO', 9)])])

Se cere:
i) Sa se defineasca un predicat *este_sesiune(S)* care este adevarat daca si numai daca *S* este un termen ce reprezinta rezultatele studentilor dintr-o sesiune.
ii) Sa se defineasca un predicat *adaugaNota(S1, CNPStudent, nota(Curs, Nota), S2)* care este adevarat daca si numai daca *S2* reprezinta rezultatele din *S1* dupa adaugarea notei *Nota* la materia *Curs* in lista de note a studentului identificat prin *CNPStudent*. Daca cursul Curs exista deja in lista de note a studentului respectiv atunci se va modifica doar valoarea corespunzatoare notei.
iii) Sa se defineasca un predicat *medie_curs(S, C, M)* care este adevarat daca si numai daca *M* reprezinta media tuturor notelor obtinute de studenti la cursul *C* in sesiunea *S*.
iv) Sa se defineasca un predicat *promovati(S, LSP)* care este adevarat daca si numai daca *LSP* reprezinta lista studentilor promovati in sesiunea *S* (care au toate notele peste 5).

Problema 15. Pentru reprezentarea rezultatelor studentilor la examenele dintr-o sesiune se foloseste un termen de forma *sesiune(ListaStudenti)*, unde *ListaStudenti* este o lista de termeni de forma *student(Nume, CNP, LN)*; *LN* este lista de note obtinute in sesiune de studentul cu numele *Nume* si codul numeric personal *CNP*.

Un exemplu este urmatorul:

sesiune([student('Ion Popescu', 123, [5, 10, 8, 9, 7]),
student('Anca Georgescu', 144, [10, 9, 10, 8, 10]),
student('Dan Marin', 187, [0, 9, 4, 5, 7])])

Se cere:
i) Sa se defineasca un predicat *sesiune(S)* care este adevarat daca si numai daca *S* este un termen ce reprezinta rezultatele studentilor dintr-o sesiune. Se presupune ca numarul de examene din sesiune este 5, iar nota in cazul in care studentul este absent la un examen este 0.
ii) Sa se defineasca un predicat *medie(S, CNPStudent, M)* care este adevarat daca si numai daca *M* reprezinta media studentului identificat prin *CNPStudent* in sesiunea *S*. Media se calculeaza doar in cazul studentilor integralisti; daca studentul are cel putin o nota mai mica de 5, atunci media este 0.
iii) Sa se defineasca un predicat *bursier(S, LSB)* care este adevarat daca si numai daca LSB reprezinta lista studentilor care obtin bursa in sesiunea S. Se considera ca obtin bursa primii 20% din studenti in ordinea mediilor.
iv) Sa se defineasca un predicat *elimina_restantieri(S1, S2)* care este adevarat daca si numai daca *S2* este *S1* din care au fost eliminati studentii cu cel putin 3 note sub 5.

Problema 16. Pentru reprezentarea unei multimi de triunghiuri se foloseste un termen de forma *multime_triunghiuri(ListaTri)* unde *ListaTri* e o lista de termeni de forma *triunghi(ID_triunghi, punct(X0, Y0), punct(X1, Y1), punct(X2, Y2))*; *ID_triunghi* e un identificator unic pentru fiecare triunghi din multimea de triunghiuri (numar intreg strict pozitiv), iar *Xi, Yi* sunt coordonatele varfurilor triunghiului.

Un exemplu este urmatorul:

multime_triunghiuri([triunghi(1, punct(2, 3), punct(5, 8), punct(6, 9)),
triunghi(2, punct(-2, 3), punct(1, -7), punct(4, 8)),
triunghi(3, punct(5, 6), punct(5, 9), punct(4, 2))])

Se cere:
i) Sa se defineasca un predicat *este_multime_triunghiuri(M)* care este adevarat daca si numai daca *M* este o multime de triunghiuri. Trebuie verificata conditia ca cele 3 puncte sa poata forma un triunghi.
ii) Sa se defineasca un predicat *suma_arii(M, A)* care este adevarat daca si numai daca A este suma ariilor triunghiurilor din multimea *M*.
iii) Sa se defineasca un predicat *echilaterale(M, LTE)* care este adevarat daca si numai daca *LTE* este lista de identificatori ai triunghiurilor echilaterale din multimea *M*.
iv) Sa se defineasca un predicat *elimina_origine(M1, M2)* care este adevarat daca si numai daca *M2* este multimea de triunghiuri *M1* din care au fost eliminate triunghiurile care contin in interior punctul de coordonate (0,0).

Problema 17. Pentru reprezentarea unei multimi de drepte se foloseste un termen de forma *multime_drepte(ListaDrepte)*, unde *ListaDrepte* este o lista de termeni de forma *dreapta(punct(X0, Y0), punct(X1, Y1))*.

Un exemplu este urmatorul:

multime_drepte([dreapta(punct(0, 1), punct(2, 3)),
dreapta(punct(-1, 4), punct(3, 2)),

dreapta(punct(-1, 0), punct(1, 2)),
dreapta(punct(0, 4), punct(3, 5))])

Se cere:

i) Sa se defineasca un predicat *multime_drepte(M)* care este adevarat daca si numai daca *M* este o multime de drepte.
ii) Sa se defineasca un predicat *elimina_origine(M1, M2)* care este adevarat daca si numai daca *M2* este multimea de drepte *M1* din care au fost eliminate dreptele care trec prin origine
iii) Sa se defineasca un predicat *paralele(M, X)* care este adevarat daca si numai daca *X* este numarul de drepte paralele (doua cate doua) din multimea de drepte *M*.
iv) Sa se defineasca un predicat *intersectii(M, LP)* care este adevarat daca si numai daca *LP* este lista de puncte obtinute prin intersectia (doua cate doua) a dreptelor din multimea *M*

Problema 18. Pentru reprezentarea unei multimi de drepte se foloseste un termen de forma *multime_drepte(ListaDrepte)*, unde *ListaDrepte* este o lista de termeni de forma *dreapta(A, B, C)*; A, B, C sunt coeficientii ecuatiei carteziane generale a unei drepte (Ax + By + C =0).

Un exemplu este urmatorul:

multime_drepte([dreapta(3, -2, 2),
dreapta(4, 1, 1),
dreapta(3, 3, 1)]).

Se cere:

i) Sa se defineasca un predicat *este_multime_drepte(M)* care este adevarat daca si numai daca *M* este o multime de drepte.
ii) Sa se defineasca un predicat *elimina_drepte(M1, M2, punct(X,Y))* care este adevarat daca si numai daca *M2* este multimea de drepte *M1* din care au fost eliminate dreptele care trec prin punctul de coordonate (*X, Y*).
iii) Sa se defineasca un predicat *perpendiculare(M, X)* care este adevarat daca si numai daca *X* este numarul de drepte perpendiculare (doua cate doua) din multimea de drepte *M*.
iii) Sa se defineasca un predicat *intersectii(M, LP)* care este adevarat daca si numai daca *LP* este lista de puncte obtinute prin intersectia (doua cate doua) a dreptelor din multimea *M.*

Problema 19. Pentru reprezentarea unei multimi de figure geometrice se foloseste un termen de forma *multime_figuri(ListaFiguri)*, unde *ListaFiguri* este o lista de termeni de forma *patrat(punct(X, Y), L), dreptunghi(punct(X, Y), L1, L2)* sau *cerc(punct(X,Y), R).* Se considera ca patratele si dreptunghiurile au laturile paralele cu axele. Un patrat este definit prin coordonatele varfului din stanga sus si lungimea laturii; un dreptunghi e definit prin coordonatele varfului din stanga sus si lungimile celor doua laturi; iar un cerc este definit prin coordonatele centrului si lungimea razei. *L, L1, L2* si *R* sunt numere pozitive.

Un exemplu este urmatorul:

multime_figuri([patrat(punct(0, 3), 10),
dreptunghi(punct(6, 2), 3, 6),
patrat(punct(-1, 2), 3),
cerc(punct(3, 5), 4),
cerc(punct(-2, 0), 3),
patrat(punct(-1, 8), 7)])

Se cere:
i) Sa se defineasca un predicat *este_multime_figuri(M)* care este adevarat daca si numai daca *M* este o multime de figuri geometrice.
ii) Sa se defineasca un predicat *contor_figuri(M, Tip, N)* care este adevarat daca si numai daca *N* este numarul de figuri geometrice de tipul *Tip* din multimea *M*, unde *Tip* poate avea una din valorile *'patrat', 'dreptunghi', 'cerc'*.
iii) Sa se defineasca un predicat *arie_figuri(M, A)* care este adevarat daca si numai daca *A* este suma ariilor figurilor geometrice din *M*.
iv) Sa se defineasca un predicat *elimina_figuri(M1, M2, punct(X,Y))* care este adevarat daca si numai daca *M2* este multimea de figuri *M1* din care au fost eliminate figurile care contin in interior punctul de coordonate (*X, Y*).

Problema 20. Pentru reprezentarea ofertelor unei agentii de turism se foloseste un termen de forma *oferte(ListaHoteluri, ListaPreturi)*. *ListaHoteluri* este o lista de termeni de forma *statiune(S, LHS)*, unde *S* este numele statiunii si *LHS* lista de hoteluri disponibile in statiunea respectiva. ListaPreturi este o lista de termeni de forma *pret(H, LPT)*, unde *H* este numele hotelului si *LPT* este lista de preturi pentru fiecare tip de camera disponibil. *LPT* este o lista de termeni de forma *tip(T, P)*, unde *T* este tipul camerei (care poate avea una din valorile *'single', 'double'* sau *'apartament'*) si *P* este pretul pe noapte al camerei respective. Se considera ca numele hotelurilor sunt unice (nu pot exista doua hoteluri cu acelasi nume nici in aceeasi statiune nici in statiuni diferite).

Un exemplu este urmatorul:

oferte([statiune('Sinaia', ['Alpin', 'Cabana Vanatorilor', 'Palace']),
statiune('Mamaia', ['Malibu', 'Patria', 'Victory'])],
[pret('Alpin', [tip('single', 200), tip('double', 280), tip ('apartament', 340)]),
pret('Malibu', [tip('double', 400)])])

Se cere:
i) Sa se defineasca un predicat *oferte_agentie(A)* care este adevarat daca si numai daca *A* este o multime de oferte ale unei agentii de turism.
ii) Sa se defineasca un predicat *adauga_oferta(A1, H, S, T, P, A2)* care este adevarat daca si numai daca *A2* reprezinta multimea de oferte *A1* la care se adauga hotelul *H* din statiunea *S* cu tipul de camera *T* si pretul corespunzator *P*. Hotelul respectiv trebuie adaugat atat in lista de hoteluri cat si in lista de preturi; daca datele despre hotelul si tipul respectiv de camera exista deja atunci se considera ca se doreste modificarea pretului.
iii) Sa se defineasca un predicat *pret_minim(A, S, H, T, P)* care este adevarat daca si numai daca *P* reprezinta cel mai mic pret dintre toate ofertele disponibile in *A* pentru statiunea *S* si tipul de camera *T*, iar *H* reprezinta hotelul la care este dipsonibil acest pret minim.

iv) Sa se defineasca un predicat *hoteluri(A, S, LH, T, P1, P2)* care este adevarat daca si numai daca *LH* reprezinta lista de hoteluri din oferta *A* din statiunea *S*, ale caror preturi pentru tipul de camera *T* se incadreaza in intervalul [*P1, P2*].

Problema 21. Pentru reprezentarea meniului unui restaurant se foloseste un termen de forma *meniu(ListaPreparate)*, unde *ListaPreparate* este o lista de termeni de forma *preparat(Nume, ListaIngrediente, Pret).*

Un exemplu este urmatorul:

meniu([preparat('ciorba de perisoare', ['carne de vita', 'orez', 'bors'], 8),
preparat('sarmale', ['carne de porc', 'orez', 'varza'], 10)
preparat('clatite', ['faina', 'oua', 'zahar'], 5)])

Se cere:
i) Sa se defineasca un predicat *meniu(M)* care este adevarat daca si numai daca *M* reprezinta meniul unui restaurant.
ii) Sa se defineasca un predicat *elimina_carne(M1, M2)* care este adevarat daca si numai daca *M2* este meniul *M1* din care au fost eliminate toate preparatele care contin: *'carne de pui'* si/sau *'carne de vita'* si/sau *'carne de porc'* si/sau *'carne de miel'*.
iii) Sa se defineasca un predicat *adauga_preparat(M1, M2, N, LI, P)* care este adevarat daca si numai daca *M2* este meniul *M1* la care a fost adaugat preparatul cu numele *N*, avand lista de ingrediente *LI* si pretul *P*. In cazul in care preparatul *N* exista deja in meniu se presupune ca se doreste modificarea datelor corespunzatoare (lista de ingrediente si/sau pret).
iv) Sa se defineasca un predicat *preparate_scumpe(M, LPS, X)* care este adevarat daca si numai daca *LPS* este lista celor mai scumpe *X*% preparate din meniul *M*.

Problema 22. Pentru reprezentarea meciurilor dintr-un campionat de fotbal se foloseste un termen de forma *campionat(ListaMeciuri)*, unde *ListaMeciuri* este o lista de termeni de forma *meci(Echipa1, Echipa2, ListaMarcatoriGazda, ListaMarcatoriOaspeti).* Daca un jucator a marcat mai multe goluri in acelasi meci, atunci numele lui va aparea de mai multe ori in lista de marcatori.

Un exemplu este urmatorul:
campionat([meci('Steaua', 'Rapid', ['Nicolae Dica', 'Valentin Badea', 'Nicolae Dica'], []),
meci('U Craiova', 'Steaua', ['Florin Costea', 'Mihai Dina'], ['Nicolae Dica'])
meci('U Craiova', 'CFR Cluj', [], [])]

Se cere:
i) Sa se defineasca un predicat *campionat(C)* care este adevarat daca si numai daca *C* reprezinta meciurile dintr-un campionat de fotbal.
ii) Sa se defineasca un predicat *meciuri_disputate(C, E, LMD)* care este adevarat daca si numai daca *LMD* reprezinta lista de meciuri disputate (o lista de termeni de forma *meci(EchipaAdversa, Scor)*) de echipa E in campionatul C.
iii) Sa se defineasca un predicat *golgheter(C, J, N)* care este adevarat daca si numai daca *J* este jucatorul care a inscris cele mai multe goluri (*N*) in campionatul *C*.

iv) Sa se defineasca un predicat *clasament(C, LE)* care este adevarat daca si numai daca *LE* este lista de echipe din campionatul *C*, ordonata crescator dupa numarul de puncte acumulate si eventual golaveraj. Se considera ca o echipa primeste 3 puncte pentru victorie, 1 punct pentru egalitate si 0 puncte pentru infrangere.

Problema 23. Se considera agenda de intalniri a unui avocat pentru o anumita zi a saptamanii. Fiecare intalnire are durata fixa de o ora si se caracterizeaza prin urmatoarele informatii: un identificator unic, o descriere, o locatie de desfasurare, un client si o ora de inceput.
Agenda se reprezinta printr-un termen agenda (Intalniri). Intalniri este lista intalnirilor dintr-o zi a saptamanii. O intalnire este un termen de forma intalnire (IdInt, Descr, IdLocatie, IdClient, OraInceput) .
Un exemplu de agenda este urmatorul:
agenda (
[intalnire (i1, 'Proces juridic 1', loc1 , c1, 8),
intalnire (i4, 'Proces juridic 2', loc1, c2, 9),
intalnire (i6, 'Proces juridic 3', loc2, c2, 10),
intalnire (i2, 'Proces juridic 4', loc3, c3, 12),
intalnire (i5, 'Proces 5', loc3, c1,13),
intalnire (i3, 'Proces 1', loc2, c2,14)]).
Se cere:
i) Sa se defineasca un predicat este_agenda (X) care este adevarat daca si numai daca X este un termen ce reprezinta o agenda.
ii) Sa se defineasca un predicat determina_locatii_clienti(A, L, C) care este adevarat daca si numai daca L si C sunt listele de locatii, respectiv clienti care apar in cadrul agendei A. Fiecare locatie, respectiv client se reprezinta prin identificatorul sau unic.
iii) Sa se defineasca un predicat conflict (I1,I2,A,C) care este adevarat daca C este lista conflictelor intalnirilor Il si I2 din cadrul agendei A. Doua intalniri diferite pot fi in conflict de locatie sau de ora.
iv) Sa se defineasca un predicat contorizare(A,N) care este adevarat daca N este numarul intalnirilor cu clienti diferiti din cadrul agendei A de intalniri ale unui avocat.

Problema 24. Se considera o imagine 2D ce contine diferite forme geometrice. Fiecare forma geometrica se caracterizeaza prin urmatoarele informatii: un identificator unic, coordonatele x si y ale centrului de greutate, lista de forme geometrice vecine, lista de culori care apar in cadrul formei.
Imaginea 2D se reprezinta printr-un termen imagine2D (ListaForme). ListaForme este lista formelor geometrice din cadrul imaginii. O forma este un termen cu urmatoarea structura formaGeom (IdForma, XCentruGreut, YCentruGreut, ListaFormeGeomVecine, ListaCulori) .
Un exemplu de reprezentare pentru o imagine 2D este urmatorul:
imagine2D (
[formaGeom (f1, 20, 30, [f2,f3,f4], [rosu, galben]),
formaGeom (f2, 20, 40, [f3,f4], [alb, galben]),
formaGeom (f3, 40, 60, [f2,f4], [rosu, violet]),
formaGeom (f4, 80, 100, [f1,f2,f3], [negru, galben, alb]),
formaGeom (f5, 150, 30, [f2,f3,f4], [rosu, galben, verde]),
formaGeom (f6, 80, 40, [f1,f3,f4], [rosu, galben, violet])]).
Se cere:

i) Sa se defineasca un predicat este_imagine2D (X) care este adevarat daca si numai daca X este un termen ce reprezinta o imagine 2D.
ii) Sa se defineasca un predicat determina_forme_culori(I, F, C) care este adevarat daca si numai daca F si C sunt listele de forme, respectiv de culori care apar in cadrul imaginii I. Fiecare forma se reprezinta prin identificatorul sau unic, iar o culoare va fi adaugata o singura data la lista C de culori.
iii) Sa se defineasca un predicat forme_cu_culoarea (I, F, C) care este adevarat daca F este lista formelor din imaginea I care contin in lista de culori, culoarea C.
iv) Sa se defineasca un predicat contorizare(I,N) care este adevarat daca N este numarul perechilor de forme vecine din imagine care au cel putin o culoare comuna.

Problema 25. Se considera o imagine 2D care este formata din patrate de dimensiuni diferite care au aceeasi culoare. Fiecare patrat se caracterizeaza prin urmatoarele informatii: un identificator unic, coordonatele x si y ale punctului din stanga sus, lista de patrate adiacente, culoare si lungimea laturii.
Imaginea 2D se reprezinta printr-un termen imagine2D (ListaPatrate). ListaPatrate este lista patratelor din cadrul imaginii. Un patrat este un termen cu urmatoarea structura patrat (IdPatrat, XPctStgSus, YPctStgSus, ListaPatrateAdiacente, Culoare, Latura) .
Un exemplu de reprezentare pentru o imagine 2D este urmatorul:
imagine2D (
[patrat (p1, 0, 0, [p2,p4], rosu,20),
patrat (p2, 20, 0, [p1,p3,p5], alb,20),
patrat (p3, 40, 0, [p2,p4], violet,20),
patrat (p4, 0, 20, [p1,p5], negru, 20),
patrat (p5, 20, 20, [p2,p4,p6], verde, 20),
patrat (p6, 40, 20, [p3,p5], galben, 30)]).
Se cere:
i) Sa se defineasca un predicat este_imagine2D (X) care este adevarat daca si numai daca X este un termen ce reprezinta o imagine 2D.
ii) Sa se defineasca un predicat determina_patrate_culori(I, P, C) care este adevarat daca si numai daca P si C sunt listele de patrate, respectiv de culori care apar in cadrul imaginii I. Fiecare patrat se reprezinta prin identificatorul sau unic, iar o culoare va fi adaugata o singura data la lista C de culori.
iii) Sa se defineasca un predicat patrate_cu_culoarea_dimensiunea (I, P, C, D) care este adevarat daca P este lista patratelor din imaginea I care contin in lista de culori, culoarea C si care au dimensiunea laturii egala cu D.
iv) Sa se defineasca un predicat aria_imagine_culoare (I, A, C) care este adevarat daca A este suma ariilor patratelor care au culoare C.

Problema 26. Se considera o imagine 2D care este formata din dreptunghiuri de dimensiuni diferite. Fiecare dreptunghi se caracterizeaza prin urmatoarele informatii: un identificator unic, coordonatele x si y ale punctului din stanga sus, lista de dreptunghiuri adiacente, lungimea si latimea dreptunghiului.
Imaginea 2D se reprezinta printr-un termen imagine2D (ListaDreptunghiuri). ListaDreptunghiuri este lista dreptunghiurilor din cadrul imaginii. Un dreptunghi este un termen cu urmatoarea structura dreptunghi (IdDreptunghi, XPctStgSus, YPctStgSus, ListaDreptunghiuriAdiacente, Lungime, Latime) .
Un exemplu de reprezentare pentru o imagine 2D este urmatorul:
imagine2D (
[dreptunghi (d1, 0, 0, [d2,d4], 20,20),

dreptunghi (d2, 20, 0, [d1,d3,d5], 30,20),
dreptunghi (d3, 40, 0, [d2,d4], 40,20),
dreptunghi (d4, 0, 20, [d1,d5], 20, 20),
dreptunghi (d5, 20, 20, [d2,d4,d6], 25, 20),
dreptunghi (d6, 40, 20, [d3,d5], 30, 30)]).
Se cere:
i) Sa se defineasca un predicat este_imagine2D (X) care este adevarat daca si numai daca X este un termen ce reprezinta o imagine 2D.
ii) Sa se defineasca un predicat determina_dreptunghiuri (I, D) care este adevarat daca si numai daca D este lista de dreptunghiuri care sunt adiacente cu un dreptunghi pentru care lungimea este egala cu latimea (patrat). Fiecare dreptunghi se reprezinta prin identificatorul sau unic.
iii) Sa se defineasca un predicat dreptunghiuri_cu_aria (I, D, A) care este adevarat daca D este lista dreptunghiurilor din imaginea I care au aria mai mica sau egala cu A.
iv) Sa se defineasca un predicat aria_imagine (I, A, X1,Y1,X2,Y2,) care este adevarat daca A este suma ariilor dreptunghiurilor cuprinse intre punctele de coordonate (X1,Y1), respectiv (X2, Y2).

Problema 27. Se considera o baza de cunostinte de forma unei clase compuse.
Baza de cunostinte se reprezinta printr-un termen de forma bc(ClasaCompusa). Un obiect din aceasta clase este un termen de forma obiect(IdObiect, Lista, ListaMembri). IdObiect este un intreg pozitiv care identifica in mod unic obiectul. Lista este o lista de perechi atribut-valoare sau comentarii, unde tipul atributului este un tip primar de date (de ex.: int, char, float). O pereche atribut-valoare este de forma atVal (NumeAtribut, Valoare). NumeAtribut este un nume de atribut si Valoare este o valoare primitiva. Un comentariu este de forma coment(Text). ListaMembri este o lista de obiecte care sunt membri in clasa compusa din care este instantiat obiectul. La randul lor aceste obiecte pot fi obiecte compuse.
Spre exemplu baza de cunostinte "Masina Dacia are un motor de 100CP, 4 usi si un rezervor de 50 litri. Consumul este de 8 litri/ 100km" se reprezinta astfel:
bc(obiect(0, [coment('Clasa Masina Dacia'), atVal(marca, 'Dacia'), atVal(nrUsi, 4)], [obiect(1, [coment('Motor'), atVal(putere, 100), atVal(consum,8)],[]), obiect(2,[coment('Rezervor'), atVal(volum, 50)] ,[])))

Se cere:
i) Sa se defineasca un predicat baza_cun(X) care este adevarat daca si numai daca X este un termen ce reprezinta o baza de cunostinte de forma unui obiect compus.
ii) Sa se defineasca un predicat elimina_comentarii(Bi,Be) care este adevarat daca si numai daca Be este baza de cunostinte rezultata prin eliminarea comentariilor din baza de cunostinte Bi.
iii) Sa se defineasca un predicat lista_atribute(B, ListaObiecte) care este adevarat daca si numai daca ListaObiecte este lista tuturor identificatorilor obiectelor din baza de cunostinte B.
iv) Sa se defineasca un predicat valori(IdObiect, NumeAtribut, BazaCun,Valori) care este adevarat daca si numai daca Valori reprezinta lista valorilor atributului NumeAtribut pentru obiectul identificat prin IdObiect din baza de cunostinte BazaCun.

Problema 28. Se considera agenda de evenimente care apar de-a lungul unei zile la o centrala nucleara. Fiecare eveniment se caracterizeaza prin urmatoarele informatii: un identificator unic, o descriere, o

locatie de desfasurare, o lista de persoane care trebuie informate in legatura cu evenimentul produs, un grad de alerta.
Agenda se reprezinta printr-un termen agenda (Evenimente). Evenimente este lista evenimentelor dintr-o zi a saptamanii. Un eveniment este un termen de forma eveniment (IdEv, Descr, IdLocatie, ListaPersoane, GradAlerta) .
Un exemplu de agenda este urmatorul:
agenda (
[eveniment (e1, 'Suprapresiune turbina 1', loc1 , [p1, p2], 1),
eveniment (e4, 'Suprapresiune turbina 2', loc1, [p3, p2], 2),
eveniment (e6, 'Suprapresiune turbina 3', loc2, [p2, p4], 3),
eveniment (e2, 'Suprapresiune turbina 4', loc3, [p4, p5], 4),
eveniment (e5, 'Suprapresiune turbina 5', loc3, [p1, p5],5),
eveniment (e3, 'Suprapresiune turbina 6', loc2, [p1, p6],6)]).
Se cere:
i) Sa se defineasca un predicat este_agenda (X) care este adevarat daca si numai daca X este un termen ce reprezinta o agenda.
ii) Sa se defineasca un predicat determina_locatii_persoane(A, L, P) care este adevarat daca si numai daca L si P sunt listele de locatii, respectiv persoane care apar in cadrul agendei de evenimente A. Fiecare locatie, respectiv persoana se reprezinta prin identificatorul sau unic.
iii) Sa se defineasca un predicat pericol (E1, E2, A, C, G) care este adevarat daca C este lista pericolelor generate de evenimentele El si E2 din cadrul agendei de evenimente A. Doua evenimente diferite pot genera pericole daca gradul de alerta este mai mare sau egal cu gradul G si apar la aceeasi locatie.
iv) Sa se defineasca un predicat contorizare(A,N) care este adevarat daca N este numarul de persoane care sunt informate de-a lungul unei zile despre aparitia de evenimente in cadrul unei centrale nucleare.

Problema 29. Se considera o baza de cunostinte de forma unei ierarhii arborescente de obiecte. Baza de cunostinte se reprezinta printr-un termen de forma bc (Ierarhie). Ierarhie este o ierarhie arborescenta de obiecte. Ea este un termen de forma obiect(IdObiect, Lista, ListaCopii). IdObiect este un intreg pozitiv care identified in mod unic obiectul radina al ierarhiei. Lista este o lista de perechi atributvaloare sau comentarii. O pereche atributvaloare este de forma atVal (NumeAtribut, Valoare). NumeAtribut este un nume de atribut si Valoare este o valoare primitiva. Un comentariu este de forma com (Text). ListaCopii este o lista de obiecte care reprezinta radacinile ierarhiilor de obiecte derivate din obiectul radacina.
Spre exemplu baza de cunostinte "Arterele si venele sunt vase de sange. Un vas de sange transports sange si are forma tubulara" se reprezinta astfel:
bc(obiect(0, [com('Clasa vaselor de sange'), atVal(nume,'Vas de sange'), atVal(transporta,sange), atVal(forma,tubulara)], [obiect(1, [com('Clasa arterelor'), atVal(nume,'Artera')], []), obiect(2, [atVal(nume,'Vena')] ,

Se cere:
i) Sa se defineasca un predicat baza_cun(X) care este adevarat daca si numai daca X este un termen ce reprezinta o baza de cunostinte de forma unei ierarhii arborescente de obiecte.
ii) Sa se defineasca un predicat elimina_comentarii(Bi,Be) care este adevarat daca si numai daca Be este baza de cunostinte rezultata prin eliminarea comentariilor din baza de cunostinte Bi.

iii) Sa se defineasca un predicat lista_atribute(B,ListaObiecte) care este adevarat daca si numai daca ListaObiecte este lista tuturor identificatorilor obiectelor din baza de cunostin1ie B, obtinuta prin parcurgerea ierarhiei in preordine.
iv) Sa se defineasca un predicat valori(IdObiect,NumeAtribut,BazaCun,Valori) care este adevarat daca si numai daca Valori reprezinta lista valorilor atributu-lui NumeAtribut pentru obiectul identificat prin IdObiect din baza de cunostinte BazaCun. Valorile pot fi proprii obiectului respectiv sau pot fi dobandite prin mostenire. Se presupune ca mostenirea se face cu suprascriere.

Problema 30. Se considera agenda de livrari la domiciliu a unui distribuitor de pizza pentru o anumita zi. Fiecare livrare se caracterizeaza prin urmatoarele informatii: un identificator unic, o descriere, o locatie de livrare, un client si o ora de livrare.
Agenda se reprezinta printr-un termen agendaLivrari (Livrari). Livrari este lista livrarilor dintr-o zi a saptamanii. O livrare este un termen de forma livrare (IdLivr, Descr, IdLocatie, IdClient, OraLivrare) .
Un exemplu de agenda este urmatorul:
agendaLivrari (
[livrare (i1, 'Livrare 1', loc1 , c1, 8),
livrare (i4, 'Livrare 2', loc2, c2, 9),
livrare (i6, 'Livrare 3', loc3, c2, 10),
livrare (i2, 'Livrare 4', loc4, c3, 12),
livrare (i5, 'Livrare 6', loc5, c1,13),
livrare (i3, 'Livrare 7', loc6, c2,14)]).
Se cere:
i) Sa se defineasca un predicat este_agendaLivrari (X) care este adevarat daca si numai daca X este un termen ce reprezinta o agenda de livrari.
ii) Sa se defineasca un predicat determina_locatii_clienti(A, L, C) care este adevarat daca si numai daca L si C sunt listele de locatii, respectiv clienti care apar in cadrul agendei de livrari A. Fiecare locatie, respectiv client se reprezinta prin identificatorul sau unic.
iii) Sa se defineasca un predicat conflict (L1,L2,A,C) care este adevarat daca C este lista conflictelor livrarilor Ll si L2 din cadrul agendei A. Doua livrari diferite sunt in conflict daca au aceeasi ora de livrare si locatii diferite.
iv) Sa se defineasca un predicat contorizare(A, N) care este adevarat daca N este numarul livrarilor cu clienti diferiti din cadrul agendei A de livrari.

Problema 31. Un articol este compus din urmatoarele elemente: titlu, lista de autori, abstract, continut si bibliografie. Continutul este o lista de sectiuni. Bibliografia este o lista de referinte bibliografice. In lista de autori, in lista de sectiuni si in lista de referinte bibliografice pot apare optional comentarii. Pentru reprezentarea unui articol se foloseste un termen de forma articol (Titlu, ListaAutori, Abstract, Continut, Bibliografie). Titlul este un termen de forma titlu(Text). Lista de autori se reprezinta printr-un termen de forma autori(ListaAutori). Un autor se reprezinta printr-un termen autor(Text). Abstractul este un termen de forma abstract(Text). Continutul este un termen de forma continut(ListaSectiuni). O sectiune este de forma sectiune (TextTitlu, TextCorp). Bibliografia este un termen de forma bibl(ListaReferinte). O referinta este de forma ref(Text). Comentariile sunt termeni de forma com(Text).
Un exemplu de articol este urmatorul:
articol (titlu('Articol 1'),

autori([coment(' Articol revista Else'), autor('Autor1'),]), abstract ('Structuri de date'), continut([sectiune('Liste','Listele simplu ....'), coment('Un comentariu'), coment('Altul'), sectiune ('Arbori',' Arbori binari de cautare...')]), bibl([ref('Referinta 1'), ref('Referinta 2')]))

Se cere:
i) Sa se defineasca un predicat articol(X) care este adevarat daca si numai daca X este un termen ce reprezinta un articol.
ii) Sa se defineasca un predicat elimina_comentarii(Ai, Ae) care este adevarat daca si numai daca Ae este articolul rezultat prin eliminarea comentariilor din articolul Ai.
iii) Sa se defineasca un predicat lista_continut(A, L) care este adevarat daca L este lista textelor sectiunilor care apar in articolul A.
iv) Sa se defineasca un predicat contorizare(A, Na, Nc, Nr) care este adevarat daca Na, Nc si Nr sunt numarul de autori, sectiuni respectiv referinte bibliografice pentru articolul A.

Problema 32. Se considera imagine 2D ce contine diferite texturi. Fiecare textura se caracterizeaza prin urmatoarele informatii: un identificator unic, numarul de pixeli din cadru regiuni unei texturi, lista de texturi geometrice vecine, aria regiune.
Imaginea 2D se reprezinta printr-un termen imagine2D (ListaRegiuniTextura). ListaRegiuniTextura este lista regiunilor segmentate folosind textura, din cadrul imaginii. O regiune este un termen cu urmatoarea structura regiuneTextura (IdRegiune, NumarPixeli, ListaRegiuniVecine, Aria) .
Un exemplu de reprezentare pentru o imagine 2D este urmatorul:
imagine2D (
[regiuneTextura (r1, 100, [r1, r2],100),
regiuneTextura (r2, 200, [r2, r3], 200),
regiuneTextura (r3, 400, [r2, r4], 340),
regiuneTextura (r4, 300, [r1,r2,r3], 210),
regiuneTextura (r5, 150, [r2,r3,r4], 270),
regiuneTextura (r6, 431, [r3,r4,r5], 120)]).
Se cere:
i) Sa se defineasca un predicat este_imagine2D (X) care este adevarat daca si numai daca X este un termen ce reprezinta o imagine 2D.
ii) Sa se defineasca un predicat determina_regiune (I, R) care este adevarat daca si numai daca R sunt listele de regiuni care apar in cadrul imaginii I. Fiecare regiune se reprezinta prin identificatorul sau unic.
iii) Sa se defineasca un predicat regiuni_cu_aria (I, R, A) care este adevarat daca R este lista regiunilor din imaginea I care aria mai mare sau egala cu A.
iv) Sa se defineasca un predicat contorizare (I, N, Np) care este adevarat daca N este numarul de regiuni cu numarul de pixeli mai mic sau egal cu Np.