Ministerul Educaţiei, Culturii și Cercetării al Republicii Moldova

Universitatea Tehnică a Moldovei

Departamentul Informatică și ingineria sistemelor

**Teză de curs**

Disciplina: Sisteme expert

Tema: SE pentru recomandarea rețetelor dietice

A efectuat: st.gr. IA-182 , Grădinaru Daniela

A verificat lect.univ.: Burlacu Natalia

Chișinău 2020

**Cuprins**

[Introducere 3](#_Toc65632416)

[Noţiunile de cunoaştere şi raţionament din perspectiva sistemelor expert 4](#_Toc65632417)

[Paradigma programării logice și compararea cu alte paradigme 5](#_Toc65632418)

[Logică și control 6](#_Toc65632419)

[Rezolvarea problemelor 6](#_Toc65632420)

[Analiza celorlalte paradigm 9](#_Toc65632421)

[Paradigma de programare iperativă 9](#_Toc65632422)

[Paradigma de programare procedurală 10](#_Toc65632423)

[Programare orientată pe obiecte 10](#_Toc65632424)

[Abordare de procesare paralelă 10](#_Toc65632425)

[Paradigmă de programare declarative 10](#_Toc65632426)

[Paradigme de programare logică 11](#_Toc65632427)

[Paradigme de programare funcțională 11](#_Toc65632428)

[Sistemul expert realizat 11](#_Toc65632429)

[Concluzii 20](#_Toc65632430)

[Bibliografie 21](#_Toc65632431)

[Anexe 22](#_Toc65632432)

[Codul sursă 22](#_Toc65632433)

[Rezultat 28](#_Toc65632434)

# Introducere

Sistemele expert fac parte din ramura informaticii denumită inteligenţă artificială. Prin inteligenţă artificială, AI, (Artificial Intelligence) se înţelege tehnologia informatică care priveşte simularea pe calculatorul electronic a unor elemente ale inteligenţei umane (deducţia logică, capacitatea de a învăţa din experienţă, obţinerea unor concluzii pe baza unor date incomplete, recunoaşterea vorbirii etc.). Prin inteligenţă artificială, cunoştinţele umane sunt captate şi prelucrate simbolic de un sistem de calcul.

Dintre domeniile principale de aplicaţie ale inteligenţei artificiale se enumeră: sistemele expert, învăţarea automată, demonstrarea automată a teoremelor, robotica, jocurile dintre om şi calculator, traducerea automată, recunoaşterea formelor, realitatea virtuală.

Sistemul expert reprezintă un sistem de programe pe calculator, fundamentat pe metode şi tehnologii ale inteligenţei artificiale, care stochează cunoştinţele mai multor experţi umani dintr-un domeniu bine definit şi pe care apoi le utilizează pentru rezolvarea problemelor dificile specifice domeniului.

Expertul uman execută acţiuni de recunoaştere, definire şi rezolvare a problemelor specifice domeniului său de expertiză. Aceste probleme sunt, de regulă, slab structurate (adică rezolvarea lor nu se poate face pe cale algoritmică), incomplete, complexe, imprecise, inconsistente, incerte, adeseori confuze şi care conţin aprecieri vagi. Expertul uman converteşte aceste probleme din sfera expertizei în sfera cunoştinţelor comune de specialitate. Astfel, aceste probleme devin complete, simple, clare, consistente, certe, precise.

Dicţionarul de calculatoare Collin consideră sistemul expert, ES (Expert System) ca *un soft care aplică cunoştinţele, sfaturile şi regulile definite de un expert uman într-un domeniu particular, asupra datelor utilizatorului, pentru a ajuta la rezolvarea unei probleme.*

Dicţionarul de informatică Larousse defineşte sistemul expert ca *un soft ce exploatează cunoştinţele explicite relative la un domeniu de expertiză, pentru a asigura un ajutor prin simularea comportamentului uman în faza de analiză şi decizie.*

Din definiţiile de mai sus sunt de reţinut cuvintele-cheie ale definirii sistemelor expert: acestea reprezintă sisteme de programe pe calculator (software); folosesc cunoştinţele ce aparţin unui expert uman (domeniu de expertiză); asigură un ajutor în rezolvarea problemelor.

Un sistem expert prezintă următoarele caracteristici:

* relevanţa mare;
* specializarea într-un domeniu bine definit;
* natura simbolică a cunoştinţelor prelucrate;
* independenţa cunoştinţelor de mecanismul de raţionament (inferenţiere);
* abordarea declarativă a cunoştinţelor;
* posibilitatea de a trata cunoştinţe incomplete şi inexacte;
* capacitatea de a explica raţionamentele efectuate;
* exploatarea dinamică a cunoştinţelor prin intermediul mecanis¬mului de raţionament;
* folosirea unor metode empirice (bazate numai pe experienţă) alături de metode ştiinţifice în rezolvarea problemelor.

Sistemele expert asigură extinderea prelucrărilor în domenii dificil de algoritmizat, ce sunt dominate de informaţii calitative (macroeconomic, juridic, politic, educaţional etc.).

## Noţiunile de cunoaştere şi raţionament din perspectiva sistemelor expert

Sistemele expert fac parte din categoria sistemelor informatice bazate pe cunoştinţe, KBS (Knowledge-Based Systems). Dicţionarele englez-român asigură pentru termenul englezesc knowledge cel puţin două echivalente româneşti: cunoaştere şi cunoştinţe.

*Cunoaşterea*, în sensul cel mai general prezentat de DEX, reprezintă acţiunea de a cunoaşte şi rezultatul ei. Cunoaşterea reprezintă “reflectarea în conştiinţă a realităţii existente independent de subiectul cunoscător.” Cunoaşterea prezintă astfel caracter individual.

*Cunoştinţa*, sub aspectele care interesează în teoria sistemelor expert, este definită în acelaşi DEX prin:

1. cunoaştere;
2. “totalitatea noţiunilor, ideilor, informaţiilor pe care le are cineva într-un domeniu oarecare”.

În sfârşit, noţiunea de *raţionament* este definită în DEX ca o “înlănţuire logică de judecăţi, care duce la o concluzie”.

Cunoaşterea, privită ca proces, şi raţionamentul reprezintă două domenii puternic interconectate: cunoaşterea se îmbogăţeşte prin rezultatele raţionamentului, iar raţionamentele au la bază cunoştinţe. Altfel exprimat, cunoaşterea şi raţionamentul pot fi considerate ca o consecinţă logică una alteia.

Abordată ca acţiune, cunoaşterea asigură stabilirea unui transfer de informaţie de la obiect la observator. Sub acest aspect, procesul de cunoaştere este denumit observaţie şi asigură informaţie relativă la timpul prezent. Informaţia obţinută pe această cale reprezintă cunoştinţe care sunt reţinute, memorate de către subiectul cunoscător şi folosite ulterior, singular sau prin asociere cu alte cunoştinţe (cunoştinţa este informaţie care prezintă utilitate şi care acţionează).

Cunoştinţele posedă două componente: componenta subiectivă (asociată cu subiectul cunoscător) şi componenta obiectivă (asociată cu obiectul cunoaşterii). Informaţia rezultată din observaţie devine cunoştinţă dacă este tezaurizată în memoria individuală şi, apoi, în memoria colectivă (cunoaştere). Cunoaşterea este compusă din piese ale cunoaşterii (pieces) care reprezintă entităţi, fiecare cu o semnificaţie anume, pe baza căreia aceste piese ale cunoaşterii sunt capturate (achiziţionate) din ansamblul cunoaşterii generale şi stocate în baze de cunoştinţe. Totalitatea pieselor cunoaşterii formează un model al domeniului în care sistemul expert are acces cu ajutorul unor proceduri de organizare, clasificare, căutare şi recunoaştere (sistemul cognitiv).

Într-un raţionament, o judecată {premisă) este legată de o altă judecată (concluzie) prin operaţie logică de derivare (inferenţiere). Există mai multe tipuri de raţionament: inducţia (trecerea de la particular la general), deducţia (trecerea de la general la particular) şi transpoziţia (trecerea de la general la general). Piesele cunoaşterii sunt reprezentate în memorie prin simboluri care se obţin prin abstractizarea universului real şi trecerea în domeniul conceptual. Raţionamentul operează cu simbolurile pieselor cunoaşterii.

# Paradigma programării logice și compararea cu alte paradigme

Paradigma poate fi denumită și metodă de rezolvare a unei probleme sau de îndeplinire a unor sarcini. Paradigma de programare este o abordare pentru a rezolva problema folosind un anumit limbaj de programare sau, de asemenea, putem spune că este o metodă pentru a rezolva o problemă folosind instrumente și tehnici care sunt disponibile pentru noi în urma unei abordări. Există multe limbaje de programare care sunt cunoscute, dar toate trebuie să urmeze o anumită strategie atunci când sunt implementate, iar această metodologie / strategie este paradigme. În afară de varietățile limbajului de programare, există o mulțime de paradigme pentru a satisface fiecare cerere.

Programarea logică este o paradigmă de programare care se bazează în mare parte pe logica formală . Orice program scris într-un limbaj de programare logic este un set de propoziții în formă logică, care exprimă fapte și reguli despre un anumit domeniu de probleme. Familiile majore de limbaj de programare logică includ Prolog , programarea setului de răspunsuri (ASP) și Datalog . În toate aceste limbi, regulile sunt scrise sub formă de clauze.

## 

## Logică și control

Programarea logică poate fi privită ca deducere controlată. Un concept important în programarea logică este separarea programelor în componenta lor logică și componenta lor de control. Cu limbaje de programare logice pure, numai componenta logică determină soluțiile produse. Componenta de control poate fi variată pentru a oferi modalități alternative de executare a unui program logic. Această noțiune este surprinsă de slogan:

*Algoritm = Logică + Control*

unde „Logică” reprezintă un program logic și „Control” reprezintă diferite strategii de demonstrare a teoremei.

## Rezolvarea problemelor

În cazul simplificat, propozițional, în care un program logic și un obiectiv atomic de nivel superior nu conțin variabile, raționamentul înapoi determină un arbore și-sau , care constituie spațiul de căutare pentru rezolvarea obiectivului. Scopul de nivel superior este rădăcina arborelui. Având în vedere orice nod din copac și orice clauză al cărei cap se potrivește cu nodul, există un set de noduri copil corespunzătoare sub-obiectivelor din corpul clauzei. Aceste noduri copil sunt grupate împreună printr-un „și”. Seturile alternative de copii care corespund unor modalități alternative de rezolvare a nodului sunt grupate împreună printr-un „sau”.

Orice strategie de căutare poate fi utilizată pentru a căuta în acest spațiu. Prolog folosește o strategie de backtracking secvențială, ultimă intrare-primă ieșire, în care se ia în considerare o singură alternativă și un sub-obiectiv la un moment dat. Sunt posibile și alte strategii de căutare, cum ar fi căutarea paralelă, urmărirea inteligentă înapoi sau cea mai bună căutare pentru a găsi o soluție optimă.

În cazul mai general, în care sub-obiectivele împărtășesc variabile, pot fi folosite alte strategii, cum ar fi alegerea sub-obiectivului care este cel mai puternic instanțiat sau suficient de instantaneat astfel încât să se aplice o singură procedură. Astfel de strategii sunt utilizate, de exemplu, în programarea logică concurentă.

Poate fi denumită ca model abstract de calcul. S-ar rezolva probleme logice, cum ar fi puzzle-uri, serii etc. În programarea logică avem o bază de cunoștințe pe care o cunoaștem înainte și împreună cu întrebarea și baza de cunoștințe care este dată mașinii, aceasta produce rezultate. În limbajele de programare normale, un astfel de concept de bază de cunoștințe nu este disponibil, dar în timp ce folosim conceptul de inteligență artificială, învățarea automată avem câteva modele precum modelul Percepție care utilizează același mecanism.

În programarea logică, accentul se pune pe baza cunoștințelor și a problemei. Executarea programului seamănă foarte mult cu dovada afirmației matematice.

Un program este alcătuit din două elemente, logica şi controlul. Prin termenul “logică” se desemnează toate noţiunile care stabilesc CE face un program, în timp ce termenul “control“ semnifică toate noţiunile sintactice care stabilesc CUM o face (de exemplu algoritmul care rezolvă o problemă). Un program scris în PASCAL sau vreun alt limbaj de programare tradiţional constă din instrucţiuni care descriu acţiunile ce trebuie executate pas cu pas de către calculator pentru ca programul să producă rezultatul dorit. Dimpotrivă un program în PROLOG este o bază de natură logică, în care programatorul defineşte obiecte şi relaţii între aceste obiecte (fie relaţii care există direct între obiecte, fie regulile după care se pot deduce alte relaţii sau proprietăţi ale acestora).

Obiectele sunt reprezentate în PROLOG prin nume simbolice. Relaţiile existente între obiecte se definesc cu ajutorul clauzelor. Clauza codifică următorul enunţ:

“Dacă c1 şi c2 şi … cn sunt adevărate, atunci cn+1 este adevărată.”

Observăm că orice faptă şi orice regulă se încheie cu caracterul “.”, iar “,” prezentă în corpul regulii suplineşte operatorul “şi”. Dacă în loc de “,” utilizăm “;” atunci în locul operatorului “şi” se consideră operatorul “sau”. Prin urmare clauza a :- b; c. codifică enunţul “dacă b sau c, atunci a”.

În general, într-un limbaj de programare tradiţional, un program exprimă o funcţie de la intrarea la ieşirea programului, în timp ce un program într-un limbaj de programare logică exprimă o relaţie între date. Întrucât relaţiile sunt mai generale decât funcţiile, programarea logică are posibilităţi mai mari decât are programarea tradiţională.

Să luăm drept exemplu un program care citeşte două numere reale şi îl afişează pe cel mai mare dintre ele. Pentru a face diferenţa dintre programarea tradiţională şi programarea logică mai explicită, vom da mai întâi programul în PASCAL, iar apoi acelaşi program în PROLOG:

Program în PASCAL:

var x,y: real;

begin

write(‘numar1 = ‘); readln(x);

write(‘numar2 = ‘); readln(y);

if (x < y) then writeln(x)

else writeln(y);

end.

Program în PROLOG:

predicates

program

mai\_mare(real,real,real)

clauses

program :- write("n1="), readreal(X), write("n2="), readreal(Y), mai\_mare(X,Y,Z),

write(Z), nl.

mai\_mare(X,X,X).

mai\_mare(X,Y,Y):-X<Y.

mai\_mare(X,Y,X):-Y<X.

goal

program

Secţiunea var (din programul PASCAL) şi secţiunea predicates (din programul PROLOG) au un caracter declarativ: în secţiunea var se declară variabilele reale x şi y, iar în secţiunea predicates se declară predicatele program (fără argumente) şi mai\_mare (cu trei argumente reale). Programul în PASCAL este doar un şir de instrucţiuni. Aceste instrucţiuni, care sunt executate în ordinea indicată de program, constituie “controlul”. “Elementul logic” în programul din PASCAL se află în relaţia “>”. Dimpotrivă, programul în PROLOG este o colecţie de clauze care descriu complet relaţia de ordine totală a două numere reale, în speţă predicatul “mai\_mare”. Această colecţie de clauze exprimă logica programului, care are şi rolul dominant într-un program PROLOG, în timp ce controlul se află în ordinea în care aranjăm şi definim predicatele. De asemenea lansarea în execuţie a unui program PROLOG, nu se face definind un punct de început după care instrucţiunile să fie preluate secvenţial, aşa cum se întâmplă în limbajele tradiţionale de programare. Execuţia unui program cere definirea unui scop (în secţiunea goal), care trebuie verificat pe baza elementelor din program (în exemplul nostru scopul este dat de predicatul program).

O altă diferenţă între programarea logică şi cea clasică constă în semnificaţia variabilelor. Programarea în PROLOG nu utilizează conceptul de atribuire. Actualizarea variabilelor prin obiecte PROLOG date se realizează prin unificare. În general, cu unificarea se analizează dacă două predicate pot fi identice. Dacă ele nu pot fi identice, procedura de unificare eşuează. Dacă ele pot fi identice, unificarea reuşeşte, iar rezultatul unificării este actualizarea variabilelor celor două predicate cu anumite de valori astfel încât cele două predicate să coincidă (“match”). Pentru actualizare se foloseşte adesea termenul legare (“binding”) sau instanţire. O variabilă prezentă într-o clauză are efect local clauzei respective. Variabilei i se leagă o valoare, iar ieşirea din clauză determină dezlegarea ei de valoare. Există o variabilă care joacă un rol special în PROLOG: ea se numeşte variabilă anonimă şi se reprezintă prin “\_”. Prezenţa variabilei anonime pe locul unui argument precizează că nu interesează valoarea legată de argument ci numai existenţa unei asemenea valori.

## Analiza celorlalte paradigm

### *Paradigma de programare iperativă*

Este una dintre cele mai vechi paradigme de programare. Prezintă o relație strânsă cu arhitectura mașinilor. Se bazează pe arhitectura Von Neumann. Funcționează schimbând starea programului prin instrucțiuni de atribuire. Efectuează sarcină pas cu pas schimbând starea. Accentul principal este asupra modului de realizare a obiectivului. Paradigma constă din mai multe afirmații și după executarea tuturor rezultatului este stocat.

Avantaj:

* Foarte simplu de implementat
* Conține bucle, variabile etc.

Dezavantaj:

* Problema complexă nu poate fi rezolvată
* Mai puțin eficient și mai puțin productiv
* Programarea în paralel nu este posibilă

Programarea imperativă este împărțită în trei mari categorii: procesare, OOP și procesare paralelă. Aceste paradigme sunt după cum urmează:

### Paradigma de programare procedurală

Această paradigmă pune accentul pe procedură în ceea ce privește modelul de mașină sub acoperire. Nu există nicio diferență între abordarea procedurală și cea imperativă. Are capacitatea de a reutiliza codul și a fost un avantaj în acel moment când era folosit din cauza reutilizării sale.

### Programare orientată pe obiecte

Programul este scris ca o colecție de clase și obiecte destinate comunicării. Cea mai mică și de bază entitate este obiectul și orice fel de calcul se efectuează numai pe obiecte. Se pune mai mult accent pe date, mai degrabă pe procedură. Poate rezolva aproape tot felul de probleme din viața reală care sunt astăzi în scenariu.

Avantaje:

* Securitatea datelor
* Moştenire
* Reutilizarea codului
* Flexibilitatea și abstractizarea sunt, de asemenea, prezente.

### Abordare de procesare paralelă

Procesarea paralelă este procesarea instrucțiunilor programului prin împărțirea acestora între mai multe procesoare. Un sistem de procesare paralel are numeroase procesoare cu scopul de a rula un program în mai puțin timp, împărțindu-le. Această abordare pare a fi ca împărțiți și cuceriți. Exemple sunt NESL (unul dintre cele mai vechi) și C / C ++ acceptă, de asemenea, din cauza unor funcții de bibliotecă.

### Paradigmă de programare declarative

Este împărțită ca Logică, Funcțională, Bază de date. În informatică programarea declarativă este un stil de construire a programelor care exprimă logica calculului fără a vorbi despre fluxul de control al acestuia. Adesea consideră programele ca teorii ale unor logici și poate simplifica scrierea de programe paralele. Accentul se pune pe ceea ce trebuie făcut, mai degrabă pe modul în care ar trebui să se facă, subliniați practic ceea ce face codul de cod. Declară doar rezultatul pe care îl dorim mai degrabă cum a fost produs. Aceasta este singura diferență între paradigmele de programare imperative (cum se face) și declarative (ce se face). Intrând în profunzime vom vedea logică, funcționalitate și bază de date.

### Paradigme de programare logică

Poate fi denumită ca model abstract de calcul. S-ar rezolva probleme logice, cum ar fi puzzle-uri, serii etc. În programarea logică avem o bază de cunoștințe pe care o cunoaștem înainte și împreună cu întrebarea și baza de cunoștințe care este dată mașinii, produce rezultate. În limbajele de programare normale, un astfel de concept de bază de cunoștințe nu este disponibil, dar în timp ce folosim conceptul de inteligență artificială, învățarea automată avem câteva modele precum modelul Percepție care utilizează același mecanism.

### Paradigme de programare funcțională

Paradigmele de programare funcțională își au rădăcinile în matematică și este independent de limbaj. Principala cheie a acestor paradigme este execuția seriilor de funcții matematice. Modelul central pentru abstractizare este funcția menită pentru anumite calcule specifice și nu structura datelor. Datele sunt cuplate slab la funcții. Funcția ascunde implementarea lor. Funcția poate fi înlocuită cu valorile lor fără a schimba semnificația programului. Unele dintre limbile cum ar fi perl, javascript folosesc în mare parte această paradigmă.

# Sistemul expert realizat

În scopul realizării unui sistem expert cu tema propusă al utilizat limbajul Prolog. Prolog este un limbaj specific inteligenței artificiale (Prolog - Programming in Logic). Originea lui aparține lui Alain Colmerauer de la Universitatea din Marsilia, Franța. Inițial a fost conceput pentru analiză lexicală.

Prima parte a programului coprinde datele care urmează a fi prelucrate.

/\*\* supe si ciorbe \*/

/\*\*nume, anul adaugarii, categorie, timp preparare, produs principal, link reteta, alergii, raiting\*\*/

mancare('Supa cu somon',2020,'supe',60,300,'peste','https://www.e-retete.ro/retete/supa-cu-somon','peste', 'fericit').

mancare('Supa de fasole cu pesto',2017,'supe',60,100,'fasole','https://www.e-retete.ro/retete/supa-de-fasole-cu-pesto','fasole', 'normal').

mancare('Supa de coada de vita',2016,'supe',60,100,'carne','https://www.e-retete.ro/retete/supa-de-coada-de-vita','carne', 'fericit').

mancare('Supa de pui cu legume si orez',2014,'supe',80,120,'carne','https://www.e-retete.ro/retete/supa-de-pui-cu-legume-si-orez','carne', 'normal').

mancare('Bors rosu',2011,'supe',90,150,'carne','https://www.e-retete.ro/retete/bors-rosu','carne', 'normal').

/\*\* salate \*/

mancare('Salata spaniola de rosii si ton',2018,'salate',30,80,'peste','https://www.e-retete.ro/retete/salata-spaniola-de-rosii-si-ton','peste', 'fericit').

mancare('Morcovi marinati',2017,'salate',50,50,'morcov','https://www.e-retete.ro/retete/morcovi-marinati','morcov', 'fericit').

mancare('Salsa de avocado si ananas',2016,'salate',90,300,'creveti','https://www.e-retete.ro/retete/salsa-de-avocado-si-ananas','creveti', 'fericit').

mancare('Cartofi noi cu macris',2013,'salate',70,50,'cartofi','https://www.e-retete.ro/retete/cartofi-noi-cu-macris','cartofi', 'normal').

mancare('Orez cu legume si susan',2013,'salate',70,90,'ciuperci','https://www.e-retete.ro/retete/orez-cu-legume-si-susan','ciuperci', 'normal').

/\*\* deserturi \*/

mancare('Pavlova cu zmeura si catina',2018,'desert',120,150,'smantana','https://www.e-retete.ro/retete/pavlova-cu-zmeura-si-catina','oua', 'fericit').

mancare('Brownie cu ciocolata',2018,'desert',120,120,'ciocolata','https://www.e-retete.ro/retete/brownie-cu-ciocolata','ciocolata', 'trist').

mancare('Brownie cu crema de ciocolata si cafea',2016,'desert',120,300,'ciocolata','https://www.e-retete.ro/retete/brownie-cu-crema-de-ciocolata-si-cafea','ciocolata', 'trist').

mancare('Tarte cu branza de capra si mere',2018,'desert',120,80,'lactate','https://www.e-retete.ro/retete/tarte-cu-branza-de-capra-si-mere','lactate', 'normal').

mancare('Prajitura cu mere',2018,'desert',90,70,'mere','https://www.e-retete.ro/retete/prajitura-cu-mere','mere', 'normal').

/\*\* sandwich \*/

mancare('Sandwich toast cu vinete si branza de capra',2018,'sandwich',30,50,'branza','https://www.e-retete.ro/retete/sandwich-toast-cu-vinete-si-branza-de-capra','lactate', 'normal').

mancare('Bruschetta cu ciuperci',2014,'sandwich',30,50,'ciuperci','https://www.e-retete.ro/retete/bruschetta-cu-ciuperci','ciuperci', 'normal').

mancare('Bruschetta cu rosii',2012,'sandwich',20,50,'rosii','https://www.e-retete.ro/retete/brownie-cu-crema-de-ciocolata-si-cafea','rosii', 'normal').

mancare('Muffuletta',2010,'sandwich',30,80,'carne','https://www.e-retete.ro/retete/muffuletta','carne', 'fericit').

mancare('Tartine cu rosii si branza feta (Dakos)',2008,'sandwich',30,50,'branza','https://www.e-retete.ro/retete/tartine-cu-rosii-si-branza-feta-dakos','lactate', 'fericit').

/\*\* bauturi \*/

mancare('Lichior de vin rosu',2017,'bauturi',30,50,'vin','https://www.e-retete.ro/retete/lichior-de-vin-rosu','vin', 'trist').

mancare('Suc de mere fiert',2014,'bauturi',50,50,'mere','https://www.e-retete.ro/retete/suc-de-mere-fiert','mere', 'fericit').

mancare('Limonada rapida',2014,'bauturi',20,50,'lamaie','https://www.e-retete.ro/retete/limonada-rapida','citrice', 'normal').

mancare('Ayran',2010,'bauturi',50,60,'lactate','https://www.e-retete.ro/retete/ayran','lactate', 'normal').

mancare('Lapte cu banane',2010,'bauturi',30,50,'lactate','https://www.e-retete.ro/retete/lapte-cu-banane','lactate', 'fericit').

/\*\* sosuri si salsa \*/

mancare('Sos rustic de rosii coapte cu pancetta',2020,'sos',50,40,'rosii','https://www.e-retete.ro/retete/sos-rustic-de-rosii-coapte-cu-pancetta','rosii', 'normal').

mancare('Pate de broccoli copt',2020,'sos',50,50,'broccoli','https://www.e-retete.ro/retete/pate-de-broccoli-copt','fara', 'normal').

mancare('Dressing de parmezan',2020,'sos',40,120,'cascaval','https://www.e-retete.ro/retete/dressing-de-parmezan','lactate', 'fericit').

mancare('BBQ Ketchup',2020,'sos',70,60,'rosii','https://www.e-retete.ro/retete/bbq-ketchup','rosii', 'normal').

mancare('Mustar de casa',2008,'sos',50,30,'mustar','https://www.e-retete.ro/retete/mustar-de-casa','mustar', 'fericit').

Secțiunea respective conține rețete din diferite categorii de mâncăruri, care vor fi recomandate utilizatorului în dependență de anumite preferințe și alergii pe care le deține. Fiecare din aceste câmpuri conține câteva puncta descriptive, în dependență de care și sunt filtrate.

Aceste puncta descriptive sunt: nume, anul adăugării, categori, timp de preparare, produs principal, link rețetă, alergii, rating.

În continuare au fost create funcții care să ajute utilizatorul să perceapă mai bine datele exprimate prin cifre și să poată allege mai ușor varianta dorită.

/\*\* functii ajutatoare \*/

durata(Y,Z,O) :- mancare(Y,\_,\_,X,\_,\_,Z,\_,\_), 20 < X, X < 40, O =='scurta'.

durata(Y,Z,O) :- mancare(Y,\_,\_,X,\_,\_,Z,\_,\_), 41 < X, X < 90, O =='medie'.

durata(Y,Z,O) :- mancare(Y,\_,\_,X,\_,\_,Z,\_,\_), 91< X, O =='lunga'.

anulCrearii(Y,Z,O) :- mancare(Y,X,\_,\_,\_,\_,Z,\_,\_), X < 2010, O =='veche'.

anulCrearii(Y,Z,O) :- mancare(Y,X,\_,\_,\_,\_,Z,\_,\_), 2011 < X, X < 2018, O =='mediu'.

anulCrearii(Y,Z,O) :- mancare(Y,X,\_,\_,\_,\_,Z,\_,\_), 2019 < X, O =='noua'.

pret(Y,Z,O) :- mancare(Y,\_,\_,\_,X,\_,Z,\_,\_), X < 50, O =='ieftin'.

pret(Y,Z,O) :- mancare(Y,\_,\_,\_,X,\_,Z,\_,\_), 51 < X, X < 150, O =='mediu'.

pret(Y,Z,O) :- mancare(Y,\_,\_,\_,X,\_,Z,\_,\_), 151 < X, O =='scump'.

alergie(Y,Z,O) :- mancare(Y,\_,\_,\_,\_,\_,Z,X,\_), X \= O.

extrageRetete(Y, Z, O1,O2,O3,O4,O5,O6) :- mainFunction(Y,Z,O1,O2,O3,O4,O5,O6).

extrageRetete(Y, Z, O1,O2,O3,O4,O5,O6) :- mainFunction(Y,Z,O1,O2,O3,O4,O5).

extrageRetete(Y, Z, O1,O2,O3,O4,O5,O6) :- mainFunction(Y,Z,O1,O2,O3,O4).

extrageRetete(Y ,Z, O1,O2,O3,O4,O5,O6) :- mainFunction(Y,Z,O1,O2,O3).

extrageRetete(Y,X,O1,O2,O3,O4,O5,O6) :- mainFunction(Y,Z,O1,O2).

extrageRetete(Y,Z,O1,O2,O3,O4,O5,O6).

mainFunction(Y,Z,O1,O2,O3,O4,O5,O6) :- mancare(Y,\_,\_,\_,\_,\_,Z,\_,O1), anulCrearii(Y,Z,O2), mancare(Y,\_,O3,\_,\_,\_,Z,\_,\_), alergie(Y,Z, O4), durata(Y,Z, O5), pret(Y,Z, O6), !.

mainFunction(Y,Z,O1,O2,O3,O4,O5) :- mancare(Y,\_,\_,\_,\_,\_,Z,\_,O1), anulCrearii(Y,Z,O2), mancare(Y,\_,O3,\_,\_,\_,Z,\_,\_), alergie(Y,Z, O4), durata(Y,Z, O5), !.

mainFunction(Y,Z,O1,O2,O3,O4) :- mancare(Y,\_,\_,\_,\_,\_,Z,\_,O1), anulCrearii(Y,Z,O2), mancare(Y,\_,O3,\_,\_,\_,Z,\_,\_), alergie(Y,Z, O4), !.

mainFunction(Y,Z,O1,O2,O3) :- mancare(Y,\_,\_,\_,\_,\_,Z,\_,O1), anulCrearii(Y,Z,O2), mancare(Y,\_,O3,\_,\_,\_,Z,\_,\_), !.

mainFunction(Y,Z,O1,O2) :- mancare(Y,\_,\_,\_,\_,\_,Z,\_,O1), anulCrearii(Y,Z,O2), !.

Funcția alergie permite să verificăm dacă tipul de alergie ales de utilizator este inclus sau nu în produsul respectiv și ulterior să se determine dacă rețeta respective poate fi recomandată sau nu.

În continuare începe apelarea funcțiilor și simularea unei interfețe prin care să fie acordate toate întrebările utilizatorului.

/\*\* SISTEM EXPERT \*/

start:-

write('Sistem expert - Recomandari de retete'),nl,

write('Va rog sa raspundeti la intrebarile de mai jos'),nl,

write('Ne va ajuta sa gasim retete in dependenta de preferintele dvs'),nl,nl,

/\*\* AICI INCEP INTREBARILE \*/

write('Care este numele dvs? '),

read(Name),nl,

write('Salut'),

write(Name),nl,

write('Care este sexul dvs? '),nl,

write('1. Feminin'),nl,

write('2. Masculin'),nl,

read(Sex),nl,

/\*\* Intrebare 1 - dispozitia pentru a incepe o reteta\*/

write('Care este dispozitia dvs? '),nl,

write('1. Trist'),nl,

write('2. Fericit'),nl,

write('3. Normal'),nl,

read(Ans1),nl,

/\*\* Intrebare 2 - anul adaugarii\*/

write('Ce tip de reteta dupa adugare doriti?'),nl,

write('1. Veche'),nl,

write('2. Mediu'),nl,

write('3. Noua'),nl,

read(Ans2),nl,

/\*\* Intrebare 3 - categorie reteta \*/

write('Ce categorie de reteta preferati?'),nl,

write('1. Supe'),nl,

write('2. Salate'),nl,

write('3. Desert'),nl,

write('4. Sandwich'),nl,

write('5. Bauturi'),nl,

write('6. Sos'),nl,

read(Ans3),nl,

/\*\* Intrebare 4 - alergii\*/

write('Aveti alergii la careva din produsele de mai jos?'),nl,

write('1. Peste'),nl,

write('2. Fasole'),nl,

write('3. Carne'),nl,

write('4. Morcov'),nl,

write('5. Creveti'),nl,

write('6. Pepeni'),nl,

write('7. Ciuperci'),nl,

write('8. Oua'),nl,

write('9. Ciocolata'),nl,

write('10. Lactate'),nl,

write('11. Mere'),nl,

write('12. Rosii'),nl,

write('13. Vin'),nl,

write('14. Citrice'),nl,

write('15. Mustar'),nl,

read(Ans4),nl,

/\*\* Intrebare 5 - durata \*/

write('Ce durata doresti sa aiba reteta?'),nl,

write('1. Scurta'),nl,

write('2. Medie'),nl,

write('3. Lunga'),nl,

read(Ans5),nl,

/\*\* Intrebare 6 - pret \*/

write('Ce pret doriti sa aiba reteta?'),nl,

write('1. Ieftin'),nl,

write('2. Mediu'),nl,

write('3. Scump'),nl,

read(Ans6),nl,

/\*\* QUESTION - YEAR PRODUCTION \*/

extrageRetete(Y, Z,Ans1,Ans2,Ans3,Ans4,Ans5,Ans6),

write('Trebuie sa incercati aceasta reteta:'),nl,

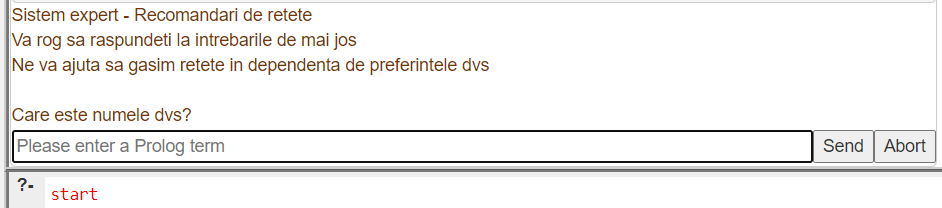
write(Y),nl,

write('Reteta o poti gasi pe acest link:'),nl,

write(Z),nl.

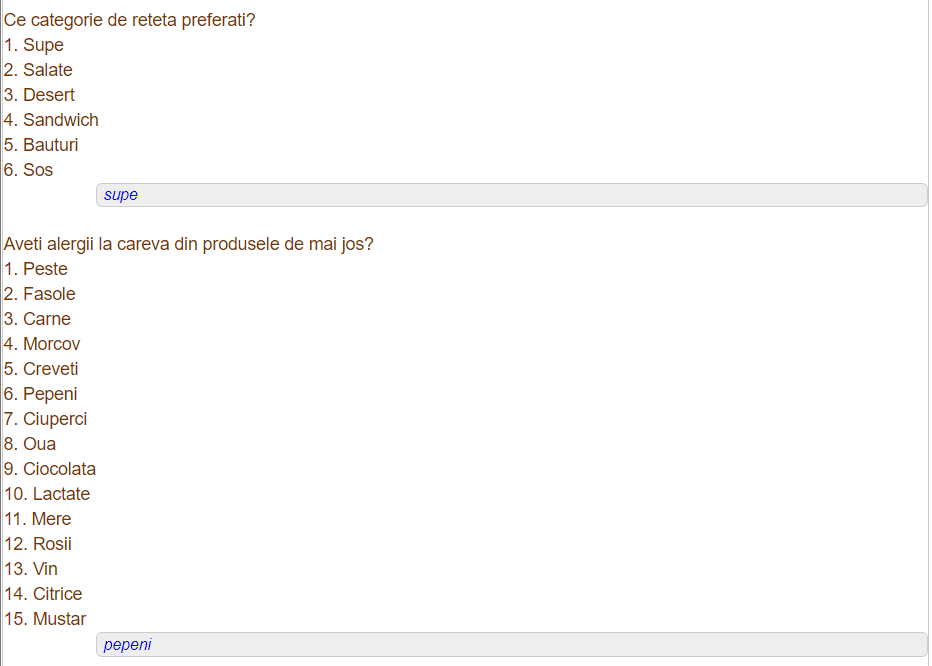
Putem observa blocurile cu fiecare din întrebări și variantele de răspuns propuse utilizatorului.

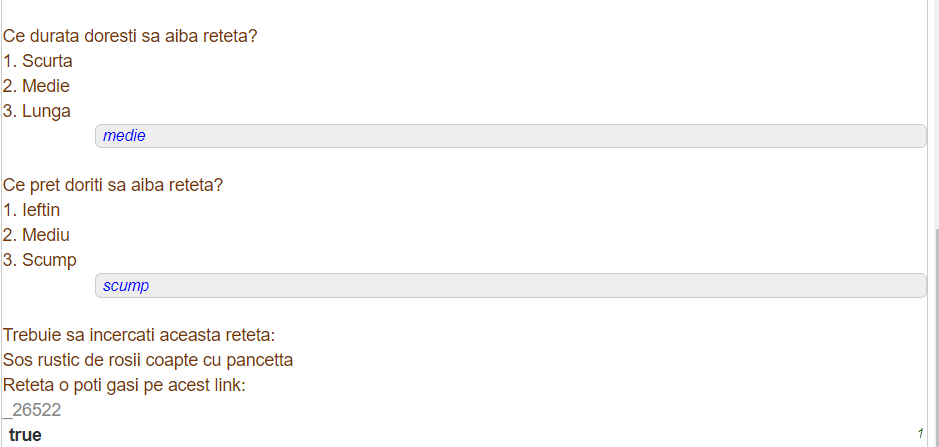
Lansăm programul utilizând comanda start.



Observăm replicile de întâmpinare. După întroducerea răspunsurilor vor urma următoarele întrebări.







# Concluzii

În urma realizării lucrării de an mi-am format deprinderi de lucru cu limbajul Prolog și am desprins mai multe repere teoretice utile pentru lucrul cu sistemele expert. Una dintre cele mai importante tematici însușite este arhitectuca unui system expert.

Arhitectura unui sistem expert cuprinde baza de cunoştinţe, baza de fapte, motorul de inferenţe, modulul de achiziţie a cunoştinţelor, modulul explicativ şi interfaţa cu utilizatorul.

*Baza de cunoştinţe* şi *baza de fapte* formează baza de date a sistemului expert.

Baza de cunoştinţe cuprinde date sub formă de cunoştinţe (reprezentate prin reguli, reţele semantice, reţele neuronale, modele hibride etc.) ale unui domeniu specializat. Aceste cunoştinţe sunt expresia entităţilor şi a relaţiilor dintre ele din universul real.

*Baza de fapte* include date despre problema de rezolvat (formularea problemei) şi fapte generate de procesul de raţionamente efectuat de motorul de inferenţe asupra bazei de cunoştinţe.

*Motorul de inferenţe* (inference engine) reprezintă partea de prelucrare a sistemului expert şi constituie subsistemul rezolutiv al acestuia

*Modulul de achiziţie* a cunoştinţelor permite conversia cunoştinţelor din forma asigurată de inginerul de cunoştinţe în forma internă de stocare pe suport magnetic specifică sistemului de calcul (reprezentarea cunoştinţelor). În plus, modulul de achiziţie a cunoştinţelor reprezintă şi o interfaţă de comunicare cu baza de date a sistemului expert (către interior) şi cu alte sisteme informatice (către exteriorul sistemului expert).

*Modulul explicativ* asigură prezentarea în limbaj natural a justificării raţionamentelor executate de motorul de inferenţe şi a întrebărilor adresate utilizatorului.

*Interfaţa cu utilizatorul* permite dialogul utilizatorului cu sistemul expert, pentru introducerea datelor de intrare şi afişarea rezultatelor pentru problema de rezolvat.

Baza de cunoştinţe, baza de fapte, modulul de achiziţie a cunoştinţelor şi modulul explicativ formează subsistemul cognitiv al sistemului expert, ca sistem bazat pe cunoştinţe, KBS (Knowledge-Based System).

Cunoştinţele realizează descrierea rezolvării unei clase de probleme. Faptele asigură descrierea aserţiunilor (afirmaţiilor) de instanţiere a unei probleme de rezolvat ce aparţine unei anumite clase de probleme.

# Bibliografie

* <http://iota.ee.tuiasi.ro/~mgavril/Simpe/L2.htm>
* <http://arheologie.ulbsibiu.ro/publicatii/carti/ai/1e.htm>
* <https://ro.warbletoncouncil.org/sistemas-expertos-11181>
* <https://www.utgjiu.ro/math/mbuneci/book/exp/cap01.pdf>
* <https://ro.qaz.wiki/wiki/Logic_programming>
* <https://ro.scribd.com/doc/37540910/Programare-Logica>
* <https://ro.wikipedia.org/wiki/Prolog>

# Anexe

## Codul sursă

/\*\* supe si ciorbe \*/

/\*\*nume, anul adaugarii, categorie, timp preparare, produs principal, link reteta, alergii, raiting\*\*/

mancare('Supa cu somon',2020,'supe',60,300,'peste','https://www.e-retete.ro/retete/supa-cu-somon','peste', 'fericit').

mancare('Supa de fasole cu pesto',2017,'supe',60,100,'fasole','https://www.e-retete.ro/retete/supa-de-fasole-cu-pesto','fasole', 'normal').

mancare('Supa de coada de vita',2016,'supe',60,100,'carne','https://www.e-retete.ro/retete/supa-de-coada-de-vita','carne', 'fericit').

mancare('Supa de pui cu legume si orez',2014,'supe',80,120,'carne','https://www.e-retete.ro/retete/supa-de-pui-cu-legume-si-orez','carne', 'normal').

mancare('Bors rosu',2011,'supe',90,150,'carne','https://www.e-retete.ro/retete/bors-rosu','carne', 'normal').

/\*\* salate \*/

mancare('Salata spaniola de rosii si ton',2018,'salate',30,80,'peste','https://www.e-retete.ro/retete/salata-spaniola-de-rosii-si-ton','peste', 'fericit').

mancare('Morcovi marinati',2017,'salate',50,50,'morcov','https://www.e-retete.ro/retete/morcovi-marinati','morcov', 'fericit').

mancare('Salsa de avocado si ananas',2016,'salate',90,300,'creveti','https://www.e-retete.ro/retete/salsa-de-avocado-si-ananas','creveti', 'fericit').

mancare('Cartofi noi cu macris',2013,'salate',70,50,'cartofi','https://www.e-retete.ro/retete/cartofi-noi-cu-macris','cartofi', 'normal').

mancare('Orez cu legume si susan',2013,'salate',70,90,'ciuperci','https://www.e-retete.ro/retete/orez-cu-legume-si-susan','ciuperci', 'normal').

/\*\* deserturi \*/

mancare('Pavlova cu zmeura si catina',2018,'desert',120,150,'smantana','https://www.e-retete.ro/retete/pavlova-cu-zmeura-si-catina','oua', 'fericit').

mancare('Brownie cu ciocolata',2018,'desert',120,120,'ciocolata','https://www.e-retete.ro/retete/brownie-cu-ciocolata','ciocolata', 'trist').

mancare('Brownie cu crema de ciocolata si cafea',2016,'desert',120,300,'ciocolata','https://www.e-retete.ro/retete/brownie-cu-crema-de-ciocolata-si-cafea','ciocolata', 'trist').

mancare('Tarte cu branza de capra si mere',2018,'desert',120,80,'lactate','https://www.e-retete.ro/retete/tarte-cu-branza-de-capra-si-mere','lactate', 'normal').

mancare('Prajitura cu mere',2018,'desert',90,70,'mere','https://www.e-retete.ro/retete/prajitura-cu-mere','mere', 'normal').

/\*\* sandwich \*/

mancare('Sandwich toast cu vinete si branza de capra',2018,'sandwich',30,50,'branza','https://www.e-retete.ro/retete/sandwich-toast-cu-vinete-si-branza-de-capra','lactate', 'normal').

mancare('Bruschetta cu ciuperci',2014,'sandwich',30,50,'ciuperci','https://www.e-retete.ro/retete/bruschetta-cu-ciuperci','ciuperci', 'normal').

mancare('Bruschetta cu rosii',2012,'sandwich',20,50,'rosii','https://www.e-retete.ro/retete/brownie-cu-crema-de-ciocolata-si-cafea','rosii', 'normal').

mancare('Muffuletta',2010,'sandwich',30,80,'carne','https://www.e-retete.ro/retete/muffuletta','carne', 'fericit').

mancare('Tartine cu rosii si branza feta (Dakos)',2008,'sandwich',30,50,'branza','https://www.e-retete.ro/retete/tartine-cu-rosii-si-branza-feta-dakos','lactate', 'fericit').

/\*\* bauturi \*/

mancare('Lichior de vin rosu',2017,'bauturi',30,50,'vin','https://www.e-retete.ro/retete/lichior-de-vin-rosu','vin', 'trist').

mancare('Suc de mere fiert',2014,'bauturi',50,50,'mere','https://www.e-retete.ro/retete/suc-de-mere-fiert','mere', 'fericit').

mancare('Limonada rapida',2014,'bauturi',20,50,'lamaie','https://www.e-retete.ro/retete/limonada-rapida','citrice', 'normal').

mancare('Ayran',2010,'bauturi',50,60,'lactate','https://www.e-retete.ro/retete/ayran','lactate', 'normal').

mancare('Lapte cu banane',2010,'bauturi',30,50,'lactate','https://www.e-retete.ro/retete/lapte-cu-banane','lactate', 'fericit').

/\*\* sosuri si salsa \*/

mancare('Sos rustic de rosii coapte cu pancetta',2020,'sos',50,40,'rosii','https://www.e-retete.ro/retete/sos-rustic-de-rosii-coapte-cu-pancetta','rosii', 'normal').

mancare('Pate de broccoli copt',2020,'sos',50,50,'broccoli','https://www.e-retete.ro/retete/pate-de-broccoli-copt','fara', 'normal').

mancare('Dressing de parmezan',2020,'sos',40,120,'cascaval','https://www.e-retete.ro/retete/dressing-de-parmezan','lactate', 'fericit').

mancare('BBQ Ketchup',2020,'sos',70,60,'rosii','https://www.e-retete.ro/retete/bbq-ketchup','rosii', 'normal').

mancare('Mustar de casa',2008,'sos',50,30,'mustar','https://www.e-retete.ro/retete/mustar-de-casa','mustar', 'fericit').

/\*\* functii ajutatoare \*/

durata(Y,Z,O) :- mancare(Y,\_,\_,X,\_,\_,Z,\_,\_), 20 < X, X < 40, O =='scurta'.

durata(Y,Z,O) :- mancare(Y,\_,\_,X,\_,\_,Z,\_,\_), 41 < X, X < 90, O =='medie'.

durata(Y,Z,O) :- mancare(Y,\_,\_,X,\_,\_,Z,\_,\_), 91< X, O =='lunga'.

anulCrearii(Y,Z,O) :- mancare(Y,X,\_,\_,\_,\_,Z,\_,\_), X < 2010, O =='veche'.

anulCrearii(Y,Z,O) :- mancare(Y,X,\_,\_,\_,\_,Z,\_,\_), 2011 < X, X < 2018, O =='mediu'.

anulCrearii(Y,Z,O) :- mancare(Y,X,\_,\_,\_,\_,Z,\_,\_), 2019 < X, O =='noua'.

pret(Y,Z,O) :- mancare(Y,\_,\_,\_,X,\_,Z,\_,\_), X < 50, O =='ieftin'.

pret(Y,Z,O) :- mancare(Y,\_,\_,\_,X,\_,Z,\_,\_), 51 < X, X < 150, O =='mediu'.

pret(Y,Z,O) :- mancare(Y,\_,\_,\_,X,\_,Z,\_,\_), 151 < X, O =='scump'.

alergie(Y,Z,O) :- mancare(Y,\_,\_,\_,\_,\_,Z,X,\_), X \= O.

extrageRetete(Y, Z, O1,O2,O3,O4,O5,O6) :- mainFunction(Y,Z,O1,O2,O3,O4,O5,O6).

extrageRetete(Y, Z, O1,O2,O3,O4,O5,O6) :- mainFunction(Y,Z,O1,O2,O3,O4,O5).

extrageRetete(Y, Z, O1,O2,O3,O4,O5,O6) :- mainFunction(Y,Z,O1,O2,O3,O4).

extrageRetete(Y ,Z, O1,O2,O3,O4,O5,O6) :- mainFunction(Y,Z,O1,O2,O3).

extrageRetete(Y,X,O1,O2,O3,O4,O5,O6) :- mainFunction(Y,Z,O1,O2).

extrageRetete(Y,Z,O1,O2,O3,O4,O5,O6).

mainFunction(Y,Z,O1,O2,O3,O4,O5,O6) :- mancare(Y,\_,\_,\_,\_,\_,Z,\_,O1), anulCrearii(Y,Z,O2), mancare(Y,\_,O3,\_,\_,\_,Z,\_,\_), alergie(Y,Z, O4), durata(Y,Z, O5), pret(Y,Z, O6), !.

mainFunction(Y,Z,O1,O2,O3,O4,O5) :- mancare(Y,\_,\_,\_,\_,\_,Z,\_,O1), anulCrearii(Y,Z,O2), mancare(Y,\_,O3,\_,\_,\_,Z,\_,\_), alergie(Y,Z, O4), durata(Y,Z, O5), !.

mainFunction(Y,Z,O1,O2,O3,O4) :- mancare(Y,\_,\_,\_,\_,\_,Z,\_,O1), anulCrearii(Y,Z,O2), mancare(Y,\_,O3,\_,\_,\_,Z,\_,\_), alergie(Y,Z, O4), !.

mainFunction(Y,Z,O1,O2,O3) :- mancare(Y,\_,\_,\_,\_,\_,Z,\_,O1), anulCrearii(Y,Z,O2), mancare(Y,\_,O3,\_,\_,\_,Z,\_,\_), !.

mainFunction(Y,Z,O1,O2) :- mancare(Y,\_,\_,\_,\_,\_,Z,\_,O1), anulCrearii(Y,Z,O2), !.

/\*\* SISTEM EXPERT \*/

start:-

write('Sistem expert - Recomandari de retete'),nl,

write('Va rog sa raspundeti la intrebarile de mai jos'),nl,

write('Ne va ajuta sa gasim retete in dependenta de preferintele dvs'),nl,nl,

/\*\* AICI INCEP INTREBARILE \*/

write('Care este numele dvs? '),

read(Name),nl,

write('Salut'),

write(Name),nl,

write('Care este sexul dvs? '),nl,

write('1. Feminin'),nl,

write('2. Masculin'),nl,

read(Sex),nl,

/\*\* Intrebare 1 - dispozitia pentru a incepe o reteta\*/

write('Care este dispozitia dvs? '),nl,

write('1. Trist'),nl,

write('2. Fericit'),nl,

write('3. Normal'),nl,

read(Ans1),nl,

/\*\* Intrebare 2 - anul adaugarii\*/

write('Ce tip de reteta dupa adugare doriti?'),nl,

write('1. Veche'),nl,

write('2. Mediu'),nl,

write('3. Noua'),nl,

read(Ans2),nl,

/\*\* Intrebare 3 - categorie reteta \*/

write('Ce categorie de reteta preferati?'),nl,

write('1. Supe'),nl,

write('2. Salate'),nl,

write('3. Desert'),nl,

write('4. Sandwich'),nl,

write('5. Bauturi'),nl,

write('6. Sos'),nl,

read(Ans3),nl,

/\*\* Intrebare 4 - alergii\*/

write('Aveti alergii la careva din produsele de mai jos?'),nl,

write('1. Peste'),nl,

write('2. Fasole'),nl,

write('3. Carne'),nl,

write('4. Morcov'),nl,

write('5. Creveti'),nl,

write('6. Pepeni'),nl,

write('7. Ciuperci'),nl,

write('8. Oua'),nl,

write('9. Ciocolata'),nl,

write('10. Lactate'),nl,

write('11. Mere'),nl,

write('12. Rosii'),nl,

write('13. Vin'),nl,

write('14. Citrice'),nl,

write('15. Mustar'),nl,

read(Ans4),nl,

/\*\* Intrebare 5 - durata \*/

write('Ce durata doresti sa aiba reteta?'),nl,

write('1. Scurta'),nl,

write('2. Medie'),nl,

write('3. Lunga'),nl,

read(Ans5),nl,

/\*\* Intrebare 6 - pret \*/

write('Ce pret doriti sa aiba reteta?'),nl,

write('1. Ieftin'),nl,

write('2. Mediu'),nl,

write('3. Scump'),nl,

read(Ans6),nl,

/\*\* QUESTION - YEAR PRODUCTION \*/

extrageRetete(Y, Z,Ans1,Ans2,Ans3,Ans4,Ans5,Ans6),

write('Trebuie sa incercati aceasta reteta:'),nl,

write(Y),nl,

write('Reteta o poti gasi pe acest link:'),nl,

write(Z),nl.

## Rezultat

