ФГБОУ ВПО

Уфимский государственный авиационный технический университет

Кафедра ВМиК

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К РАСЧЕТНО-ГРАФИЧЕСКОЙ РАБОТЕ**

по дисциплине «Функционально-логическое программирование»

вариант 10

Выполнила: студент гр. ПРО-211

                         Кабирова Л.Р.

Проверила: Усманова А.Р.

Уфа  2017

Задание 1:

Доказать с помощью аксиомы структурной индукции справедливость одного из приведенных ниже свойств согласно своему варианту (номеру в списочном составе группы): reverse (reverse xs) = xs.

Начнем с описания функции reverse:

reverse :: [a] -> [a]

reverse [] = []

reverse (x:xs) = reverse xs ++ [x]

Исходя из этого определения докажем, что reverse (reverse xs) = xs. Для упрощенного доказательства воспользуемся несколькими утверждениями.

Утверждение 1:

reverse([x]) = [x]

Доказательство утверждения 1:

reverse([x]) = reverse([]) ++ [x] = [] ++ [x] = [x]

Утверждение 2:

Для любых списков xs и ys

reverse(xs ++ ys) = reverse(ys)++ reverse(xs)

Доказательство утверждения 2:

1. reverse([]++ys)= reverse(ys) = reverse(ys)++[] = reverse(ys)++reverse([])

2. Пусть reverse(xs++ys) = reverse(ys)++ reverse(xs). Тогда reverse([x|xs] ++ ys) = reverse([x|xs++ys] ) = reverse(xs ++ ys) ++ [x] = reverse(ys)++ reverse(xs) ++[x] reverse(ys) ++ reverse([x|xs]) = reverse(ys) ++ reverse(xs) ++ [x] Ч.Т.Д.

Теперь можно переходить к доказательству исходного утверждения.

Доказательство:

1. reverse(reverse([])) = reverse([]) = []

2. Пусть reverse(reverse(xs)) = xs. Тогда reverse(reverse([x|xs]) = reverse(reverse(xs)++[x]) = reverse([x]) ++ reverse(reverse(xs)) = [x] ++ reverse(reverse(xs)) = [x] ++ xs = [x|xs] Ч.Т.Д.

Итак, мы доказали, что reverse (reverse xs) = xs.

Задание 2:

Написать собственную экспертную систему по примеру приведенной ниже. По возможности добавить механизм вывода. По возможности создать исполняемое приложение.

Исходный код:

domains

str=string.

int=integer.

intlist=integer\*.

predicates

nondeterm rule(int,str,str,intlist).

nondeterm complete(int).

nondeterm recognition(str).

nondeterm discover(intlist).

nondeterm ask(int).

facts

fact(int,str).

clauses

//база правил экспертной системы

rule(1,"блюдо","первое",[1,2]).

rule(2,"блюдо","фастфуд",[1,3]).

rule(3,"блюдо","второе",[1]).

rule(4,"блюдо","салат",[4]).

rule(5,"блюдо","десерт",[5]).

rule(6,"первое","суп щавелевый с галушками",[6,10,12,18,27]).

rule(7,"первое","рассольник",[6,12,16,27]).

rule(8,"первое","борщ",[6,12,14]).

rule(9,"первое","куриный суп",[7,11,12]).

rule(10,"первое","суп с фрикадельками",[8,11,12]).

rule(11,"первое","молочный суп",[11,26]).

rule(12,"первое","уха",[9,12]).

rule(13,"второе","пюре с котлетой",[8,12,29]).

rule(14,"второе","голубцы",[8,13,14]).

rule(15,"второе","плов",[6,13]).

rule(16,"второе","манты",[8,10,28]).

rule(17,"второе","пельмени",[8,10]).

rule(18,"второе","мясо по-французски",[6,12,20,22]).

rule(19,"второе","рыба фаршированная",[9,24]).

rule(20,"фастфуд","пицца",[10,15,17,20,21]).

rule(21,"фастфуд","бутерброд",[10,17,20,22]).

rule(22,"салат","зимний",[6,12,16,18,22,29]).

rule(23,"салат","греческий",[15,16,20,21,22]).

rule(24,"салат","хе",[6,19,27]).

rule(25,"салат","красная шапочка",[12,18,20,22,23]).

rule(26,"салат","летний",[15,16,22]).

rule(27,"салат","нежность",[7,12,16,18,20,22]).

rule(28,"десерт","блин сладкий",[10,24,26]).

rule(29,"десерт","блин",[10,26]).

rule(30,"десерт","шоколадные конфеты",[24,25]).

rule(31,"десерт","торт",[10,24]).

rule(32,"десерт","мороженое",[26,24,30]).

rule(33,"десерт","мороженое",[26,25,30]).

rule(34,"десерт","мороженое",[26,30]).

rule(35,"десерт","молочный коктейль",[26]).

ask(X):-fact(X,"y"),!.

ask(X):-fact(X,"n"),!,fail.

//экспертная система

ask(1):-write("блюдо должно подаваться горячим/теплым ?"),!,complete(1).

ask(2):-write("блюдо жидкое?"),!,complete(2).

ask(3):-write("блюдо быстрого приготовления?"),!,complete(3).

ask(4):-write("блюдо легкое?"),!,complete(4).

ask(5):-write("блюдо сладкое?"),!,complete(5).

ask(6):-write("из мяса?"),!,complete(6).

ask(7):-write("из курицы?"),!,complete(7).

ask(8):-write("из мясного фарша?"),!,complete(8).

ask(9):-write("из рыбы?"),!,complete(9).

ask(10):-write("с тестом / из теста?"),!,complete(10).

ask(11):-write("с лапшой?"),!,complete(11).

ask(12):-write("с картофелем?"),!,complete(12).

ask(13):-write("с рисом?"),!,complete(13).

ask(14):-write("с капустой?"),!,complete(14).

ask(15):-write("с помидорами?"),!,complete(15).

ask(16):-write("с солеными/свежими огурцами?"),!,complete(16).

ask(17):-write("с колбасой?"),!,complete(17).

ask(18):-write("с яйцом?"),!,complete(18).

ask(19):-write("с морковью?"),!,complete(19).

ask(20):-write("с сыром?"),!,complete(20).

ask(21):-write("с оливками?"),!,complete(21).

ask(22):-write("с майонезом?"),!,complete(22).

ask(23):-write("с гранатами?"),!,complete(23).

ask(24):-write("с начинкой?"),!,complete(24).

ask(25):-write("с шоколадом?"),!,complete(25).

ask(26):-write("из молока?"),!,complete(26).

ask(27):-write("кисловатое?"),!,complete(27).

ask(28):-write("на пару?"),!,complete(28).

ask(29):-write("традиционное?"),!,complete(29).

ask(30):-write("очень холодное?"),!,complete(30).

recognition(X):- rule(N, X, Y, Z), discover(Z), !,

write("       это более всего похоже на ", X, " - ", Y, " (правило ", N,")"), nl,

recognition(Y).

recognition("блюдо"):- write("это блюдо системе неизвестно!"),nl,!.

recognition("первое"):- write("это блюдо системе неизвестно, но его состав очень оригинален!"),nl,!.

recognition("второе"):- write("это блюдо системе неизвестно, но его состав очень оригинален!"),nl,!.

recognition("фастфуд"):- write("это блюдо системе неизвестно!"),nl,!.

recognition("салат"):- write("это блюдо системе неизвестно, но его состав очень оригинален!"),nl,!.

recognition("десерт"):- write("это блюдо системе неизвестно!"),nl,!.

recognition(\_).

discover([]).

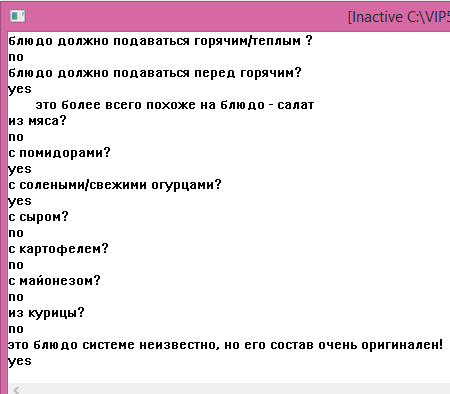
discover([X|Y]):- ask(X), discover(Y).

complete(X):- nl, readln(Y), assert(fact(X, Y)), Y="y".

goal

retractall(\_), recognition("блюдо").

Пример работы программы:



Вывод:

в ходе данной расчетно-графической работы были получены навыки «строительства» индуктивных рассуждений, а также была создана экспертная система в среде Visual Prolog.

cond(1,"Длина тела до 30 см")

cond(2,"Длина хвоста до 3 см")

cond(3,"Вес около 0,8 кг")

cond(4,"Иглы не более 3 см")

cond(5,"Имеет бурый окрас")

cond(6,"Имеет длинные уши")

cond(7,"Длина тела от 12 до 27 см")

cond(8,"Иглы от 1.7 до 1.9 см")

cond(9,"Предпочитает сухие степи и пустыни")

cond(10,"Ареал обитания охватывает Африку и Азию")

cond(11,"Питается мелкими птицами")

cond(12,"Может вырастать до 35 см в длину")

cond(13,"Вес может достигать 1.2 кг")

cond(14,"Распространен в Австрии и Германии")

cond(15,"Живет на приусадебных участках")

rule("Европейский ёж (лат. Erinaceus europaeus)",[1,2,3,4,5,15])

rule("Ушастый ёж (лат. Hemiechinus auritus)",[6,7,8,9,10,11])

rule("Восточноевропейский ёж (лат. Erinaceus concolor)",[12,13,14,15])

rule("Огромный ёж",[15,14,13,5,4,3,11,10,9,8,7,6,1])

rule("Егор обыкновенный",[14,11,9,7,4,3,2,1])