Последняя лекция – пересдачи, досдачи

Дерево описывается структурой, доменом, может быть не бинарным в прологе, ибо можно использовать вариантные домены, может получиться логически строго организованное дерево.

Списки можно организовывать по отдельным фактам, которые распределены в разных местах, но структурированы как-то (списки, деревья), все зависит от программиста.

Лисп – обработка списков, избран для создания системы искусственного интеллекта, если фиксируют знания как в прологе. То есть и лисп, и пролог – инструменты.

Объединение двух списков:

//Вопрос про эффективное присоединение списков короткого и длинного.

append ( [ ] , L, L)

append ( [ H1 | T1 ], L2, [ H1 | T3 ]) :- append ( T1, L2, T3) //Т3 – хвост третьего списка

Удалить все вхождения числа в списке

1. delete\_all ( \_, [ ], [ ] ).
2. delete\_all ( X, [ X | L ], L1 ) :- delete\_all ( X, L, L1).
3. delete\_all (X, [ Y | L ], [ Y | L1] ) :- X <> Y , delete\_all ( X, L, L1).

TP: delete\_all (2; [2, 3, 2, 4], R)  
шаг 1 ТЦ: delete\_all (2, [2, 3, 2, 4], R)  
 ПРII {2 = X1, [3, 2, 4] = L1, R = L11} //ПР2, 2 римская  
 ТР: delete\_all (2, [3, 2, 4], L11)  
шаг 2 ТЦ: delete\_all (2, [3, 2, 4], L11)  
 ПРII {2 = X2, 3 = Y2, [2, 4] = L2, L11 = L12}  
 ТР: X<>Y <-> 2<>3  
 delete\_all (2, [2, 4], L11)  
шаг 3 ТЦ: 2 <> 3  
 успех  
 ТР: delete\_all (2, [2, 4], L11)  
шаг 4 ТЦ: delete\_all (2, [2, 4], L11)  
 ПРII {2 = X, [4] = L, L11 = L14}  
 ТР: delete\_all (2, [4], L14)  
шаг 5 ТЦ: delete\_all (2, [4], L14)  
 ПРIII: {2 = X5, 4 = Y5, [] = L5, L14 = L15}  
 TP: X<> Y <-> 2 <> 4  
 delete(2, [ ], L15)  
шаг 6 ТЦ: delete\_all(2, [ ], L15)  
 ПРI: {L15 = [ ] }  
 ТР: пусто, успех ---🡪 R = [3, 4]