## Topic Facts, Rules, and Queries

Que-What is logic programming ?Explain Facts,Rules,Queries along with an example

**Exercise**  1.1 Which of the following sequences of characters are atoms, which are variables, and which are neither?

1. vINCENT
2. Footmassage
3. variable23
4. Variable2000
5. big\_kahuna\_burger
6. ’big  kahuna  burger’
7. big  kahuna  burger
8. ’Jules’
9. \_Jules
10. ’\_Jules’

**Exercise**  1.2 Which of the following sequences of characters are atoms, which are variables, which are complex terms, and which are not terms at all? Give the functor and arity of each complex term.

1. loves(Vincent,mia)
2. ’loves(Vincent,mia)’
3. Butch(boxer)
4. boxer(Butch)
5. and(big(burger),kahuna(burger))
6. and(big(X),kahuna(X))
7. \_and(big(X),kahuna(X))
8. (Butch  kills  Vincent)
9. kills(Butch  Vincent)
10. kills(Butch,Vincent

**Exercise**  1.3 How many facts, rules, clauses, and predicates are there in the following knowledge base? What are the heads of the rules, and what are the goals they contain?

   woman(vincent).  
   woman(mia).  
   man(jules).  
   person(X):-  man(X);  woman(X).  
   loves(X,Y):-  father(X,Y).  
   father(Y,Z):-  man(Y),  son(Z,Y).  
   father(Y,Z):-  man(Y),  daughter(Z,Y).

**Exercise**  1.4 Represent the following in Prolog:

1. Butch is a killer.
2. Mia and Marsellus are married.
3. Zed is dead.
4. Marsellus kills everyone who gives Mia a footmassage.
5. Mia loves everyone who is a good dancer.
6. Jules eats anything that is nutritious or tasty.

**Exercise**  1.5 Suppose we are working with the following knowledge base:

   wizard(ron).  
   hasWand(harry).  
   quidditchPlayer(harry).  
   wizard(X):-  hasBroom(X),  hasWand(X).  
   hasBroom(X):-  quidditchPlayer(X).

How does Prolog respond to the following queries?

1. wizard(ron).
2. witch(ron).
3. wizard(hermione).
4. witch(hermione).
5. wizard(harry).
6. wizard(Y).
7. witch(Y).

## Topic Unification

**Exercise**  2.1 Which of the following pairs of terms unify? Where relevant, give the variable instantiations that lead to successful unification.

1. bread  =  bread
2. ’Bread’  =  bread
3. ’bread’  =  bread
4. Bread  =  bread
5. bread  =  sausage
6. food(bread)  =  bread
7. food(bread)  =  X
8. food(X)  =  food(bread)
9. food(bread,X)  =  food(Y,sausage)
10. food(bread,X,beer)  =  food(Y,sausage,X)
11. food(bread,X,beer)  =  food(Y,kahuna\_burger)
12. food(X)  =  X
13. meal(food(bread),drink(beer))  =  meal(X,Y)
14. meal(food(bread),X)  =  meal(X,drink(beer))

**Exercise**  2.2 We are working with the following knowledge base:

   house\_elf(dobby).  
   witch(hermione).  
   witch(’McGonagall’).  
   witch(rita\_skeeter).  
   magic(X):-  house\_elf(X).  
   magic(X):-  wizard(X).  
   magic(X):-  witch(X).

**Exercise**  2.3 Which of the following queries are satisfied? Where relevant, give all the variable instantiations that lead to success.

1. ?-  magic(house\_elf).
2. ?-  wizard(harry).
3. ?-  magic(wizard).
4. ?-  magic(’McGonagall’).
5. ?-  magic(Hermione).
6. word(determiner,a).  
      word(determiner,every).  
      word(noun,criminal).  
      word(noun,’big  kahuna  burger’).  
      word(verb,eats).  
      word(verb,likes).  
        
      sentence(Word1,Word2,Word3,Word4,Word5):-  
            word(determiner,Word1),  
            word(noun,Word2),  
            word(verb,Word3),  
            word(determiner,Word4),  
            word(noun,Word5).

What query do you have to pose in order to find out which sentences the grammar can generate? List all sentences that this grammar can generate in the order that Prolog will generate them in.

## Topic Recursion

**Exercise**  3.1 In the text, we discussed the predicate

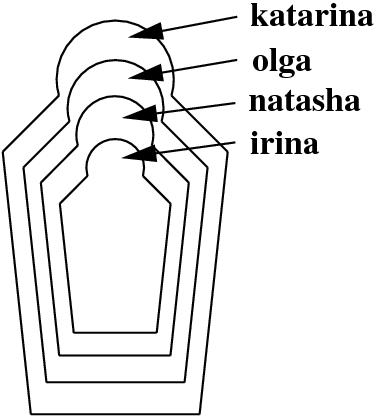
   descend(X,Y)  :-  child(X,Y).  
   descend(X,Y)  :-  child(X,Z),  
                                     descend(Z,Y).

Suppose we reformulated this predicate as follows:

   descend(X,Y)  :-  child(X,Y).  
   descend(X,Y)  :-  descend(X,Z),  
                                     descend(Z,Y).

Would this be problematic?

**Exercise**  3.2 Do you know these wooden Russian dolls (Matryoshka dolls) where the smaller ones are contained in bigger ones? Here is a schematic picture:



First, write a knowledge base using the predicate directlyIn/2 which encodes which doll is directly contained in which other doll. Then, define a recursive predicate in/2 , that tells us which doll is (directly or indirectly) contained in which other dolls. For example, the query in(katarina,natasha) should evaluate to true, while in(olga,  katarina) should fail.

## Topic Lists

**Exercise**  4.1 How does Prolog respond to the following queries?

1. [a,b,c,d]  =  [a,[b,c,d]].
2. [a,b,c,d]  =  [a|[b,c,d]].
3. [a,b,c,d]  =  [a,b,[c,d]].
4. [a,b,c,d]  =  [a,b|[c,d]].
5. [a,b,c,d]  =  [a,b,c,[d]].
6. [a,b,c,d]  =  [a,b,c|[d]].
7. [a,b,c,d]  =  [a,b,c,d,[]].
8. [a,b,c,d]  =  [a,b,c,d|[]].
9. []  =  \_.
10. []  =  [\_].
11. []  =  [\_|[]].

**Exercise**  4.2 Which of the following are syntactically correct lists? If the representation is correct, how many elements does the list have?

1. [1|[2,3,4]]
2. [1,2,3|[]]
3. [1|2,3,4]
4. [1|[2|[3|[4]]]]
5. [1,2,3,4|[]]
6. [[]|[]]
7. [[1,2]|4]
8. [[1,2],[3,4]|[5,6,7]]

**Exercise**  4.3 Write a predicate second(X,List) which checks whether X is the second element of List .

**Exercise**  4.4 Write a predicate swap12(List1,List2) which checks whether List1 is identical to List2 , except that the first two elements are exchanged.

**Exercise**  4.5 Suppose we are given a knowledge base with the following facts:

   tran(eins,one).  
   tran(zwei,two).  
   tran(drei,three).  
   tran(vier,four).  
   tran(fuenf,five).  
   tran(sechs,six).  
   tran(sieben,seven).  
   tran(acht,eight).  
   tran(neun,nine).

Write a predicate listtran(G,E) which translates a list of German number words to the corresponding list of English number words. For example:

   listtran([eins,neun,zwei],X).

should give:

   X  =  [one,nine,two].

Your program should also work in the other direction. For example, if you give it the query

   listtran(X,[one,seven,six,two]).

it should return:

   X  =  [eins,sieben,sechs,zwei].

**Exercise**  4.6 Write a predicate twice(In,Out) whose left argument is a list, and whose right argument is a list consisting of every element in the left list written twice. For example, the query

   twice([a,4,buggle],X).

should return

   X  =  [a,a,4,4,buggle,buggle]).

And the query

   twice([1,2,1,1],X).

should return

   X  =  [1,1,2,2,1,1,1,1].

(Hint: to answer this question, first ask yourself “What should happen when the first argument is the empty list?”. That’s the base case. For non-empty lists, think about what you should do with the head, and use recursion to handle the tail.)

## Topic Arithmetic

**Exercise**  5.1 How does Prolog respond to the following queries?

1. X  =  3\*4.
2. X  is  3\*4.
3. 4  is  X.
4. X  =  Y.
5. 3  is  1+2.
6. 3  is  +(1,2).
7. 3  is  X+2.
8. X  is  1+2.
9. 1+2  is  1+2.
10. is(X,+(1,2)).
11. 3+2  =  +(3,2).
12. \*(7,5)  =  7\*5.
13. \*(7,+(3,2))  =  7\*(3+2).
14. \*(7,(3+2))  =  7\*(3+2).
15. 7\*3+2  =  \*(7,+(3,2)).
16. \*(7,(3+2))  =  7\*(+(3,2)).

**Exercise**  5.2 Write a predicate addone/2 whose first argument is a list of integers, and whose second argument is the list of integers obtained by adding 1 to each integer in the first list. For example, the query

   ?-  addone([1,2,7,2],X).

should give

  X  =  [2,3,8,3].

## Topic Database Manipulation

**Exercise**  11.1 Suppose we start with an empty database. We then give the command:

   assert(q(a,b)),  assertz(q(1,2)),  asserta(q(foo,blug)).

What does the database now contain?

We then give the command:

   retract(q(1,2)),  assertz(  (p(X)  :-    h(X))  ).

What does the database now contain?

We then give the command:

   retractall(q(\_,\_)).

What does the database now contain?

## Topic Prolog Advantages and Disadvantages

**Advantages :**  
**1.**Easy to build database. Doesn’t need a lot of programming effort.  
**2.**Pattern matching is easy. Search is recursion based.  
**3.**It has built in list handling. Makes it easier to play with any algorithm involving lists.

**Disadvantages :**  
**1.**LISP (another logic programming language) dominates over prolog with respect to I/O features.  
**2.**Sometimes input and output is not easy.

## 