# Λύσεις για τις ασκήσεις του φυλλαδίου

## 1 Παραγοντικό

Υλοποίηση του παραγοντικού χωρίς tail recursion (χρησιμοποιούμε if-then-else διότι το παραγοντικό είναι συνάρτηση):

```
fact(N, F) :-
  ( N =:= 0 -> F = 1 ; N > 0, M is N-1, fact(M, F1), F is F1*N ).
```

και με tail recursion (προσέξτε ότι το βοηθητικό κατηγόρημα `fact/3` έχει διαφορετικό πλήθος ορισμάτων):

```
fact(N, F) :- fact(N, 1, F).

fact(N, P, F) :-
  ( N =:= 0 -> F = P ; N > 0, M is N-1, P1 is P*N, fact(M, P1, F) ).
```

## 2 Πύργοι του Hanoi

```
hanoi(0, _Source, _Target, _Auxiliary, []).
hanoi(N, Source, Target, Auxiliary, Moves) :-
  N > 0,
  M is N-1,
  hanoi(M, Source, Auxiliary, Target, Moves1),
  hanoi(M, Auxiliary, Target, Source, Moves2),
  append(Moves1, [move(Source, Target) | Moves2], Moves).
```

Η λύση για 3 δακτυλίους (μορφοποιημένη για να διαβάζεται ευκολότερα):

```
?- hanoi(3, left, right, middle, Moves).
Moves = [ move(left, right),
          move(left, middle),
          move(right, middle),
          move(left, right),
          move(middle, left),
          move(middle, right),
          move(left, right)
        ] .
```

# 3 Τρίλιζα

```
is_winning_pos([x,x,x]).
is_winning_pos([x,x,b]).
is_winning_pos([x,b,x]).
is_winning_pos([b,x,x]).

triple(L, T) :- member(T, L).
triple(L, T) :- transpose(L, LT), member(T, LT).
triple([[A,_,_],[_,B,_],[_,_,C]], [A,B,C]).
triple([[_,_,A],[_,B,_],[C,_,_]], [A,B,C]).

transpose([[A,B,C],[D,E,F],[G,H,I]], [[A,D,G],[B,E,H],[C,F,I]]).

winner(L) :- L = [_,_,_], triple(L, T), is_winning_pos(T).
```

# 4 Ζέβρα

```
zebra(Houses) :-
  Houses = [house(norwegian, _, _, _, _),
            house(_, blue, _, _, _),
            house(_, _, _, milk, _), _, _],
  member(house(englishman, red, _, _, _), Houses),
  member(house(spaniard, _, dog, _, _), Houses),
  member(house(_, green, _, coffee, _), Houses),
  member(house(ukranian, _, _, tea, _), Houses),
  right_of(house(_, ivory, _, _, _), house(_, green, _, _, _), Houses),
  member(house(_, _, snails, _, old_gold), Houses),
  member(house(_, yellow, _, _, kools), Houses),
  next_to(house(_, _, _, _, chesterfields), house(_, _, fox, _, _), Houses),
  next_to(house(_, _, _, _, kools), house(_, _, horse, _, _), Houses),
  member(house(_, _, _, orange_juice, lucky_strike), Houses),
  member(house(japanese, _, _, _, parliaments), Houses).

right_of(X, Y, [X, Y|_]).
right_of(X, Y, [_, X, Y|_]).
right_of(X, Y, [_, _, X, Y|_]).
right_of(X, Y, [_, _, _, X, Y]).

next_to(X, Y, Houses) :- right_of(X, Y, Houses).
next_to(X, Y, Houses) :- right_of(Y, X, Houses).
```