```
% Task: Partition {1,..., n} into r sum-free sets

% Input: positive integers n, r

1{in(I, 1..r)}1 :- I = 1..n.
% achieved: set { 1,...,n} partitioned into subsets
{I:in(I,1)}, ..., { I:in(I,r)}

:- in(I,X), in(J,X), in(I+J,X).
% Achieve these subsets are sum-free
```

```
% Task: Find independent sets of vertices of size n

% Input: set node/1 of vertices of a graph G;
% set edge/2 of edges of G, positive integer n.

n {in(X) : node(X)}n.
% achieved : in/1 is a set consisting of n vertices

:- in(X), in(Y), edge(X,Y).
% achieved: in/1 has no pairs of adjacent vertices

# show in/1.
```

```
% Task: Find a clique of size n

% Input: set node/1 of vertices of a graph G;
% set edge/2 of edges of G, positive integer n.

n {in(X) : node(X)}n.
% achieved : in/1 is a set consisting of n vertices

:- in(X), in(Y), not edge(X,Y).
% achieved: in/1 has no pairs of non-adjacent vertices

# show in/1.
```

```
%Task: Find the number of edges in a graph G and degrees of vertices

% Input: set of nodes/1 of vertices of a graph G; set
edge/2 of edges of G.

adjacent(X,Y) :- edge(X,Y).
adjacent(X,Y) :- edge(Y,X).
% achieved: adjacent (X,Y) iff X,Y are adjacent in the
graph.

number_of_edges(N/2) :- N = #count{X,Y : adjacent(X,Y)}.
% achieved: number_of_edges(N) if and only if N is the number of edges of
G.

degree(X,D) :- vertex(X), D = #count{Y: adjacent(X,Y)}.
% achieved:  degree(X,D) if and only if D is the degree of the vertex X of
G.

#show degree/2. #show number_of_edges/1.
```

```
% Task: Find largest independent sets of vertices

% Input: set node/1 of vertices of a graph G;
                        % set edge /2 of edges of G

{in(X)} :- node(X).
% achieved: in/1 is a set consisting of n vertices of G.

:- in(X), in(Y), edge(X,Y).
% achieved: in/1 has no pairs of adjacent vertices.

#maximize{1,X : in(X)}.
% achieved: the number of elements of in/1 is maximal

#show in/1.
```

```
% Task: Calculate the number of classes taught on each of the five
floors.

% Input: set where/2 of all pairs (C,I) such that
                      % class C is taught on the I-th floor.

howmany(I,N) :- N = #count{C : where(C,I)}, I=1..5.
% achieved: howmany(I,N) if and only if the number of classes %
taught on the I-th floor is N.

#show howmany/2.
```