1 Aggregates and Constraints

1. Consider the following choice rule

   ```
   {a(1); b(2)} 1.
   ```

   This means that our models must consist of *at most* one of these choices. We can rewrite the constraint choice rule as

   ```
   {a(1); b(2)}.
   :- a(1), b(2).
   ```

   Propose an equivalent transformation for the rule

   ```
   1 {a(1); b(2)}.
   ```

2. Let k and n be positive integers. Consider the following program P.

   ```
   k {q(1); q(2); ...; q(n)} k.
   ```

   How many models should we expect in P?

   Now consider the following program Q.

   ```
   {q(1); q(2); ...; q(n)} n.
   ```

   How many models should we expect in Q? If we remove the integrity constraint in Q, does that affect the number of models we obtain?

3. Consider the following program.

   ```
   person(tom).  person(mary).
   salary(tom, 500).  salary(mary, 600).
   ```

   Write an aggregate function to compute the sum of Tom and Mary's salary and return it with the predicate `total`.

   Note that you can show only the total salary with `#show total/1.`.

4. Consider the following program R.

   ```
   {in(1..n)} = k.
   ```

   This returns the set of single-value predicates `in(X)` of size k. How can we use the aggregate function `#count {...}` to return the same set of models as R?

5. Consider the following choice rule with constraints.

   ```
   a {p(1..n)} b.
   ```

   Use the `#count` aggregate to find an equivalent transformation without cardinality constraints.

2 Optimization statements

1. Consider the following program which is the optimization version of the scheduling problem.

```
% Find the largest possible number of pairwise disjoint
% members of a given list of finite sets.

% input:  for a list S_1, ..., S_n of sets, its length n and
% the set s/2 of pairs X, I such that X is in S_I
```

   Write a program to find the largest possible number of pairwise disjoint members of a given list of finite set.

2. Consider the following program which is the optimization version of the clique problem.

```
% Find the largest clique.

% input:  set vertex/1 of vertices of a graph G;
% set edge/2 of edges of G
```

   Write a program to find the largest size of the clique.

3. Consider the following program which is the optimization version of subset sum.

```
% Among the subsets of a given set of numbers for which
% the sum doesn't exceed the given upper bound find the
% one for which this sum is maximal.

% input :  a set number/1 of positive integers; a positive
% integer.
```

   Write a program to find the maximal sum.