Exercises

Answer Set Solving in Practice

Exercise 1 (Answer Set Programming by Examples)

You can check your answers running `clingo` with the program files in the directory `exercise01`. For example, if the file is named `example.lp`, the command `clingo example.lp 0` will compute all stable models.

Exercise 1.1 (Positive Programs)

Follow some programs and their stable models:

```
a.
b.
--------
{ a, b }
```

```
a.
b :- a.
--------
{ a, b }
```

```
b :- a.
a.
--------
{ a, b }
```

```
b :- a, c.
a.
----------
{ a }
```

```
b :- a, c.
a.
c :- a.
-----------
{ a, b, c }
```

```
a :- b.
b :- a.
-------
{ }
```

```
a :- b.
b :- a.
a.
--------
{ a, b }
```

```
a :- b.
b :- a.
a :- a.
-------
{ }
```

```
a :- b.
b :- a.
a :- c.
c.
-----------
{ a, b, c }
```

```
a :- b.
b :- a, b.
a :- c.
c.
----------
{ a, c }
```

Find the stable models of the next positive programs:

```
a.
b.
c.
--------
```

```
a.
b :- a, c.
c.
----------
```

```
b :- a, c.
a.
c.
----------
```

```
b :- a, c.
a.
d :- b.
----------
```

```
b :- a, c.
a.
c :- a.
d :- a, b, c.
-------------
```

```
a :- b.
b :- a, c.
c.
----------
```

```
a :- b.
b :- a, c.
a.
c.
----------
```

```
a :- b.
b :- a.
a :- a, c.
c.
----------
```

```
a :- b.
b :- a, c.
a :- c, d.
c.
d :- c.
----------
```

```
a :- b.
b :- a, b.
a :- c.
c.
d :- a, c.
b :- d.
----------
```

Exercise 1.2 (Negation)

Follow some programs and their stable models:

```
a :- not b.
----------
{ a }
```

```
a :- not b.
b.
----------
{ b }
```

```
a :- not b.
b :- not c.
----------
{ b }
```

```
a :- not b.
b :- not c.
c :- d.
d.
----------
{ a, c, d }
```

```
a :- not b.
b :- not c.
c :- d.
d :- c.
------------
{ b }
```

```
a :- not b.
b :- not c.
d :- a.
e :- b.
---------
{ b, e }
```

```
a :- not b.
b :- not c.
a.
b :- a.
---------
{ a, b }
```

```
a :- not b.
b :- not c.
c :- not d.
d.
---------
{ b, d }
```

Find the stable models of the next programs:

```
a :- not b, c.
c.
--------------
```

```
a :- not b, c.
b.
c.
--------------
```

```
a :- not b, d.
b :- not c.
d.
--------------
```

```
a :- not b, c.
b :- not c.
c :- d.
d :- not e.
--------------
```

```
a :- not b.
b :- not d.
c :- d, not e.
d :- c, not e.
--------------
```

```
a :- not b.
b :- not c.
d :- a.
e :- not c.
-----------
```

```
a :- not b.
b :- not c.
a :- not d.
b :- e.
e.
-----------
```

```
a :- d.
a :- not b.
b :- not c, d.
c :- not d, b.
d.
--------------
```

Exercise 1.3 (Choice Rules)

Follow some programs and their stable models:

```
{ a }.
------
{ }
{ a }
```

```
0 { a } 1.
----------
{ }
{ a }
```

```
1 { a }.
--------
{ a }
```

```
2 { a }.
--------
```

```
{ a }.
b :- not a.
c :- a.
------
{ b }
{ a, c }
```

```
{ a }.
b :- not a.
b :- a.
----------
{ b }
{ a, b }
```

```
a.
{ b } :- a.
--------
{ a }
{ a, b }
```

```
{ a }.
{ b } :- a.
--------
{ }
{ a }
{ a, b }
```

Find the stable models of the next programs:

```
{ a }.
b :- not a.
c :- a, not d.
e :- b, c.
------------
```

```
{ a }.
b :- not a, c.
b :- a, d.
c.
------------
```

```
a.
b :- a.
{c} :- b, not d.
------------
```

```
{ a }.
b :- not a.
{ c } :- b.
------------
```

Follow some programs and their stable models:

```
{ a; b }.
------------
{ }
{ a }
{ b }
{ a, b }
```

```
1 { a; b }.
------------
{ a }
{ b }
{ a, b }
```

```
{ a; b } 1.
------------
{ }
{ a }
{ b }
```

```
1 { a; b } 1.
------------
{ a }
{ b }
```

```
2 { a; b }.
------------
{ a, b }
```

```
0 { a; b } 2.
------------
{ }
{ a }
{ b }
{ a, b }
```

```
3 { a; b }.
------------
```

```
{ a; b }.
c :- not a, not b.
d :- not a, b.
e :- a, not b.
f :- a, b.
------------
{ c }
{ b, d }
{ a, e }
{ a, b, f }
```

```
1 { a; b } 1.
c :- a.
d :- b.
------------
{ a, c }
{ b, d }
```

```
1 { a; b } 1.
c :- a.
d :- b.
c :- d.
d :- c.
------------
{ a, c, d }
{ b, c, d }
```

```
{ a }.
1 { b; c } 1 :- a.
d :- a, c.
------------
{ }
{ a, b }
{ a, c, d }
```

Find the stable models of the next programs:

```
{ a; b }.
c :- not a.
d :- b.
------------
```

```
1 { a; b } 1.
c :- not b.
d :- not a.
e :- c, not d.
------------
```

```
1 { a; b } 1.
c :- a.
d :- b.
c :- d, not a.
d :- c, not b.
------------
```

```
{ a }.
b :- a.
1 { c; d } 1 :- b.
e :- a, c.
------------
```

Exercise 1.4 (Constraints)

Follow some programs and their stable models:

```
:- not a.
---------
```

```
:- b.
----------
{ }
```

```
b :- not c.
:- b.
--------
```

```
b :- not c.
c.
:- b.
--------
{ c }
```

```
b.
:- b.
:- c.
------
```

```
b :- not d.
d.
:- b.
:- c.
----------
{ d }
```

```
c :- d.
d :- c.
:- c.
:- not d.
--------
```

```
a :- c.
b :- not c.
c.
:- not a, not b.
--------
{ a, c }
```

Find the stable models of the next programs:

```
d.
:- not a, d.
------------
```

```
b :- c.
:- b.
------------
```

```
b :- not c, d.
d.
:- b.
------------
```

```
b :- not c, d.
d.
:- b.
------------
```

```
c :- d.
d.
:- b.
:- c.
------------
```

```
b :- d.
c :- d.
:- b.
:- c.
------------
```

```
c :- d.
d :- c.
:- not c.
:- d.
------------
```

```
a :- not b.
b :- not c.
:- not a, not c.
------------
```

Exercise 1.5 (Programming with choice rules and constraints)

Follow some programs and their stable models:

```
{ a }.
:- not a.
----------
{ a }
```

```
{ a }.
:- not b, not a.
:- not b, a.
--------------
```

```
{ a }.
c :- a.
:- not c.
--------------
{ a, c }
```

```
{ a }.
c :- a.
d :- not a.
:- c.
:- d.
----------
```

```
1 { a; b }.
c :- a.
d :- b.
:- c, not e.
------
{ b, d }
```

```
{ a; b } 1.
c :- not a, not b.
d :- b.
:- not c, not d.
------
{ c }
{ b, d }
```

```
1 { a; b } 1.
c :- a, not b.
d :- not a, b.
:- c, d.
------
{ a, c }
{ b, d }
```

```
{ a }.
1 { b; c } 1.
:- not a, b.
:- a, c.
--------
{ c }
{ a, b }
```

Find the stable models of the next programs:

```
{ a }.
:- b.
:- not a.
----------
```

```
{ a }.
b.
:- b, not a.
:- b, a.
----------
```

```
{ a }.
d :- c.
c :- not a.
:- not d.
----------
```

```
{ a }.
d :- not a.
c :-  a.
e :- d.
:- not e.
----------
```

```
1 { a; b }.
c :- not a.
d :-  not b.
e :- a, b.
:- not c, not e.
----------
```

```
{ a; b } 1.
c :- not a.
d :- not b.
:- c, d.
----------
```

```
1 { a; b } 1.
c :-  a.
d :- b.
:- c.
:- d.
----------
```

```
{ a }.
1 { b; c } 1 :- a.
d :- a, b.
:- not d.
------------
```

Exercise 1.6 (Cardinality rules)

Follow some programs and their stable models:

```
{ a; b }.
:- 1 { a;b } 1.
----------
{ }
{ a, b }
```

```
1 { a; b }.
c :- 1 { a;b } 1.
:- not c.
----------
{ a, c }
{ b, c }
```

```
1 { a; b }.
c :- a.
d :- b.
:- { c;d } 1.
----------
{ a, b, c, d }
```

```
1 { a; b }.
{ c } :- { a;b } 1.
------------
{ a }
{ a, c }
{ b }
{ b, c }
{ a, b }
```

```
1 { a; b }.
a :- 1{ a; b }.
b :- 1{ a; b }.
-----------
{ a, b }
```

```
1{ a; b }1.
1{ c; d } :- a.
:- 3 { a;b;c;d }.
-----------
{ a, c }
{ a, d }
{ b }
```

```
{ a; b } 1.
c :- a.
:- 2 { a;b;c }.
------------
{ }
{ b }
```

```
1 { a; b }.
c :- a.
d :- b.
:- 2 { a;b;c;d } 2.
-------------
{ a, b, c, d }
```

Find the stable models of the next programs:

```
{ a; b }.
c.
:- 2 { a;b;c } 2.
------------
```

```
1 { a; b }.
c :- 1 {a;b} 1.
:- c.
------------
```

```
1 { a; b }.
a :- a.
d :- b.
:- 2 { c; d }.
------------
```

```
1 { a; b }.
{ c } :- 1 { a;b } 1.
-------------
```

```
1 { a;b;c }.
a :- 1 { a;b;c }.
b :- 1 { a;b;c }.
c :- 1 { a;b;c }.
-----------
```

```
1 { a;b } 1.
1 { c;d } 1 :- a.
:- 2 { a;b;c;d }.
 ------------
```

```
{ a; b } 1.
c :- a.
:- { a;b;c } 1.
------------
```

```
1 { a; b }.
c :- a.
c :- b.
:- 2 { a;b;c }.
------------
```

Exercise 1.7 (Aggregates)

Follow some programs and their stable models:

```
a. b. c.

a(V) :- V = #sum{ 1: a;     2: b;     5: c     }.
b(V) :- V = #sum{ 1: a;     2: b;     5: not c }.
c(V) :- V = #sum{ 1: a;     2: not b; 5: not c }.
d(V) :- V = #sum{ 1: not a; 2: not b; 5: not c }.

e(V) :- V = #sum{ 1: a;     1: b;     1: c     }.
f(V) :- V = #sum{ 1: a;     1: b;     1: not c }.
g(V) :- V = #sum{ 1: a;     1: not b; 1: not c }.
h(V) :- V = #sum{ 1: not a; 1: not b; 1: not c }.

i(V) :- V = #sum{ 1: a;     1: b;     5: c     }.
j(V) :- V = #sum{ 1: a;     5: b;     5: c     }.
k(V) :- V = #sum{ 5: a;     5: b;     5: c     }.

l(V) :- V = #sum{ 1,x: a;     1,y: b;     1,z: c     }.
m(V) :- V = #sum{ 1,x: a;     1,y: b;     1,z: not c }.
n(V) :- V = #sum{ 1,x: a;     1,y: not b; 1,z: not c }.
o(V) :- V = #sum{ 1,x: not a; 1,y: not b; 1,z: not c }.

p(V) :- V = #sum{ 1,x: a;     1,y: b;     5,z: c     }.
q(V) :- V = #sum{ 1,x: a;     5,y: b;     5,z: c     }.
r(V) :- V = #sum{ 5,x: a;     5,y: b;     5,z: c     }.

s(V) :- V = #sum{ 1,x: a;     1,x: b;     1,x: c     }.
t(V) :- V = #sum{ 1,x: a;     1,x: b;     1,x: not c }.
u(V) :- V = #sum{ 1,x: a;     1,x: not b; 1,x: not c }.
v(V) :- V = #sum{ 1,x: not a; 1,x: not b; 1,x: not c }.

w(V) :- V = #sum{ 1,x: a;     1,x: b;     5,x: c     }.
x(V) :- V = #sum{ 1,x: a;     5,x: b;     5,x: c     }.
y(V) :- V = #sum{ 5,x: a;     5,x: b;     5,x: c     }.

--------------------------------------
{ a, b, c,
  a(8), b(3), c(1), d(0),
  e(1), f(1), g(1), h(0),
  i(6), j(6), k(5),
  l(3), m(2), n(1), o(0),
  p(7), q(11), r(15),
  s(1), t(1), u(1), v(0),
  w(6), x(6), y(5) }
```

```
{ a; b }.
:- 1 #sum{ 1,x: a; 1,y: b }.
----------------------------
{ }
```

```
{ a; b }.
:- 2 #sum{ 1,x: a; 1,y: b }.
----------------------------
{ }
{ a }
{ b }
```

```
{ a; b }.
:- #sum{ 1,x: a; 1,x: b } 1.
----------------------------
```

```
{ a; b }.
:- #sum{ 1: a; 1: b } 1.
------------------------
```

```
{ a; b }.
:- -1 #sum{ -1,x: a; -1,y: b }.
-------------------------------
{ a, b }
```

```
{ a; b }.
:- 1 #sum{ 1,x: not a; 1,y: not b }.
-------------------------------------
{ a, b }
```

```
{ a; b }.
c :- 1 #sum{ 1,x: a; 1,y: b }.
:- 2 #sum{ 1,x: a; 1,y: c }.
-----------------------------
{ }
{ b, c }
```

```
{ a; b }.
c :- 2 #sum{ 1: a; 2: b }.
:- 3 #sum{ 1: a; 2: c }.
-------------------------
{ }
{ a }
{ b, c }
```

Find the stable models of the next programs:

```
{ a; b }.
c :- a.
:- 1 #sum{ 1,x: c; 1,y: b }.
---------------------------
```

```
{ a; b }.
:- 2 #sum{ 1,x: a; 10,y: b }.
-----------------------------
```

```
{ a; b }.
c :- a.
d :- b.
:- #sum{ 1,x: c; 1,x: d } 1.
---------------------------
```

```
{ a; b }.
c :- a.
d :- b.
:- #sum{ 1: c; 1: d } 1.
---------------------------
```

```
{ a; b }.
:- 0 #sum{ -1,x: a; -1,y: b }.
------------------------------
```

```
{ a; b; c}.
:- 1 #sum{ 1,x: not a;
           1,y: not b;
           1,z: not c}.
----------------------
```

```
{ a; b }.
c :- 2 #sum{ 1,x: a; 1,y: b }.
:- 2 #sum{ 2,x: a; 1,y: c }.
------------------------------
```

```
{ a; b }.
c :- 3 #sum{ 1: a; 2: b }.
:- 2 #sum{ 1: c; 2: b }.
-------------------------
```