CSE 579
Week 4 Graded Assignment
Template for clingo Work

Problem 1

Input Program

```
% Each row has exactly one queen
{queen(R,1..n)}=1 :- R=1..n.
% No two queens are on the same column
:- queen(R1,C), queen(R2,C), R1!=R2.
% No two queens are on the same diagonal
:- queen(R1,C1), queen(R2,C2), R1!=R2, |R1-R2|=|C1-C2|.
% No queens in the 4*4=16 squares in the middle of the board.
:- queen(R,C), R>=3, R<=6, C>=3, C<=6.
```

Command Line

```
clingo p4.lp -c n=8 0
```

Output of clingo

```
clingo version 5.6.2
Reading from p4.lp
Solving...
Answer: 1
queen(5,7) queen(1,4) queen(2,6) queen(4,2) queen(3,8) queen(6,1) queen(7,3) queen(8,5)
Answer: 2
queen(2,3) queen(3,1) queen(6,8) queen(4,7) queen(1,5) queen(5,2) queen(7,6) queen(8,4)
Answer: 3
queen(2,4) queen(4,1) queen(5,8) queen(3,7) queen(1,6) queen(6,2) queen(7,5) queen(8,3)
Answer: 4
queen(6,7) queen(1,3) queen(2,5) queen(3,2) queen(4,8) queen(5,1) queen(8,6) queen(7,4)
SATISFIABLE

Models       : 4
Calls        : 1
Time         : 0.005s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.005s
```

Problem 2

Input Program

```
% Each row has exactly one queen
{queen(R,1..n)}=1 :- R=1..n.
% No two queens are on the same column
:- queen(R1,C), queen(R2,C), R1!=R2.
% No two queens are on the same diagonal
:- queen(R1,C1), queen(R2,C2), R1!=R2, |R1-R2|=|C1-C2|.
```

Command Line

You should write 10 command lines below, one for each value of n from the set {3, …, 12}.

```
clingo p4.lp -c n=3
clingo p4.lp -c n=4
clingo p4.lp -c n=5
clingo p4.lp -c n=6
clingo p4.lp -c n=7
clingo p4.lp -c n=8
clingo p4.lp -c n=9
clingo p4.lp -c n=10
clingo p4.lp -c n=11
clingo p4.lp -c n=12
```

Output of clingo

Since there are 10 outputs for 10 different values of n, please do not copy and paste the outputs of clingo for Problem 2 (and also for Problem 8 later).

Answer to Questions

Draw a table that lists the number of solutions and the times to compute all solutions. Use CPU time that clingo returns.

Value n	Number of solutions	time
3	0	0.002s
4	2	0.002s
5	10	0.003s
6	4	0.004s
7	40	0.005s
8	92	0.010s
9	352	0.030s
10	724	0.167s
11	2680	1.672s
12	14200	30.215s

Problem 3

Input Program

```
a(1,1,8).
a(2,3,3).
a(2,4,6).
a(3,2,7).
a(3,5,9).
a(3,7,2).
a(4,2,5).
a(4,6,7).
a(5,5,4).
a(5,6,5).
a(5,7,7).
a(6,4,1).
a(6,8,3).
a(7,3,1).
a(7,8,6).
a(7,9,8).
a(8,3,8).
a(8,4,5).
a(8,8,1).
a(9,2,9).
a(9,7,4).
% Each number 1..9 is assigned to one cell in each box
1 { a(X,Y,N):X=1..9,Y=1..9,X1<=X,X<=X1+2,Y1<=Y,Y<=Y1+2 } 1
            :- N=1..9, X1 = 3*(0..2)+1, Y1 = 3*(0..2)+1.
% no two different numbers given a row and a column
:- a(X,Y,N), a(X,Y,N1), N!=N1.
% no two different columns given a row and a number
:- a(X,Y,N), a(X,Y1,N), Y!=Y1.
% no two different rows given a column and a number
:- a(X,Y,N), a(X1,Y,N), X!=X1.
```

Command Line

```
clingo p4.lp 0
```

Output of clingo

```
clingo version 5.6.2
Reading from p4.lp
Solving...
Answer: 1
a(1,1,8) a(2,3,3) a(2,4,6) a(3,2,7) a(3,5,9) a(3,7,2) a(4,2,5) a(4,6,7) a(5,5,4) a(5,6,5) a(5,7,7)
a(6,4,1) a(6,8,3) a(7,3,1) a(7,8,6) a(7,9,8) a(8,3,8) a(8,4,5) a(8,8,1) a(9,2,9) a(9,7,4) a(4,1,1)
a(1,2,1) a(6,1,2) a(7,2,2) a(1,3,2) a(5,1,3) a(8,2,3) a(8,1,4) a(2,2,4) a(4,3,4) a(7,1,5) a(3,3,5)
a(3,1,6) a(5,2,6) a(9,3,6) a(9,1,7) a(6,3,7) a(6,2,8) a(2,1,9) a(5,3,9) a(9,5,1) a(3,6,1) a(4,4,2)
a(8,5,2) a(2,6,2) a(9,4,3) a(4,5,3) a(1,6,3) a(3,4,4) a(7,6,4) a(1,5,5) a(6,5,6) a(8,6,6) a(1,4,7)
a(7,5,7) a(5,4,8) a(2,5,8) a(9,6,8) a(7,4,9) a(6,6,9) a(2,7,1) a(5,9,1) a(5,8,2) a(9,9,2) a(7,7,3)
a(3,9,3) a(1,8,4) a(6,9,4) a(6,7,5) a(9,8,5) a(2,9,5) a(1,7,6) a(4,9,6) a(2,8,7) a(8,9,7) a(4,7,8)
a(3,8,8) a(8,7,9) a(4,8,9) a(1,9,9)
SATISFIABLE
Models      : 1
Calls       : 1
Time        : 0.022s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time    : 0.021s
```

Problem 4

Input Program

```
% the facts to represent the given numbers are provided below

a(1,1,9). a(1,2,14). a(1,6,3). a(1,8,5). a(1,9,15). a(1,11,2). a(1,15,7). a(1,16,1).
a(2,1,6). a(2,2,12). a(2,6,14). a(2,11,10). a(2,15,5). a(2,16,11).
a(3,1,4). a(3,4,7). a(3,5,6). a(3,8,13). a(3,9,16). a(3,12,1). a(3,13,2). a(3,16,9).
a(4,2,15). a(4,3,16). a(4,5,9). a(4,6,7). a(4,11,11). a(4,12,6). a(4,14,3). a(4,15,14).
a(5,2,7). a(5,3,15). a(5,14,2). a(5,15,16).
a(6,1,5). a(6,3,13). a(6,5,14). a(6,7,15). a(6,10,10). a(6,12,3). a(6,14,1). a(6,16,8).
a(7,2,8). a(7,4,10). a(7,6,9). a(7,7,4). a(7,8,11). a(7,9,13). a(7,10,6). a(7,11,15). a(7,13,14).
a(7,15,3).
a(8,1,16). a(8,5,5). a(8,7,3). a(8,10,14). a(8,12,9). a(8,16,6).
a(9,1,15). a(9,5,16). a(9,7,10). a(9,10,9). a(9,12,13). a(9,16,14).
a(10,2,9). a(10,4,6). a(10,6,5). a(10,7,13). a(10,8,3). a(10,9,1). a(10,10,15). a(10,11,4).
a(10,13,7). a(10,15,12).
a(11,1,2). a(11,3,8). a(11,5,15). a(11,7,14). a(11,10,16). a(11,12,12). a(11,14,5).
a(11,16,13).
a(12,2,13). a(12,3,12). a(12,14,9). a(12,15,11).
a(13,2,5). a(13,3,3). a(13,5,2). a(13,6,16). a(13,11,13). a(13,12,10). a(13,14,12). a(13,15,9).
a(14,1,8). a(14,4,4). a(14,5,12). a(14,8,1). a(14,9,6). a(14,12,7). a(14,13,15). a(14,16,3).
a(15,1,10). a(15,2,1). a(15,6,15). a(15,11,16). a(15,15,6). a(15,16,2).
a(16,1,11). a(16,2,2). a(16,6,8). a(16,8,14). a(16,9,3). a(16,11,1). a(16,15,10). a(16,16,7).

% write the remaining part of the clingo program below
% range of values
value(1..16).
% 16x16 grid
cell(1..16, 1..16).
% each cell in a grid must be assigned exactly one value
1 { a(X,Y,N) : value(N) } 1 :- cell(X,Y).
% no two different numbers given a row and a column
:- a(X,Y,N), a(X,Y,N1), N!=N1.
% no two different columns given a row and a number
:- a(X,Y,N), a(X,Y1,N), Y!=Y1.
% no two different rows given a column and a number
:- a(X,Y,N), a(X1,Y,N), X!=X1.
% no two cells in the same 4x4 box can have the same value
:- a(X,Y,N), a(X1,Y1,N), (X-1)/4 == (X1-1)/4, (Y-1)/4 == (Y1-1)/4, X != X1, Y != Y1.
% print solution
#show a/3.
```

Command Line

```
clingo p4.lp 0
```

Output of clingo

```
clingo version 5.6.2
Reading from p4.lp
Solving...
Answer: 1
a(1,1,9) a(1,2,14) a(1,6,3) a(1,8,5) a(1,9,15) a(1,11,2) a(1,15,7) a(1,16,1) a(2,1,6) a(2,2,12)
a(2,6,14) a(2,11,10) a(2,15,5) a(2,16,11) a(3,1,4) a(3,4,7) a(3,5,6) a(3,8,13) a(3,9,16)
```

```
a(3,12,1) a(3,13,2) a(3,16,9) a(4,2,15) a(4,3,16) a(4,5,9) a(4,6,7) a(4,11,11) a(4,12,6)
a(4,14,3) a(4,15,14) a(5,2,7) a(5,3,15) a(5,14,2) a(5,15,16) a(6,1,5) a(6,3,13) a(6,5,14)
a(6,7,15) a(6,10,10) a(6,12,3) a(6,14,1) a(6,16,8) a(7,2,8) a(7,4,10) a(7,6,9) a(7,7,4)
a(7,8,11) a(7,9,13) a(7,10,6) a(7,11,15) a(7,13,14) a(7,15,3) a(8,1,16) a(8,5,5) a(8,7,3)
a(8,10,14) a(8,12,9) a(8,16,6) a(9,1,15) a(9,5,16) a(9,7,10) a(9,10,9) a(9,12,13) a(9,16,14)
a(10,2,9) a(10,4,6) a(10,6,5) a(10,7,13) a(10,8,3) a(10,9,1) a(10,10,15) a(10,11,4)
a(10,13,7) a(10,15,12) a(11,1,2) a(11,3,8) a(11,5,15) a(11,7,14) a(11,10,16) a(11,12,12)
a(11,14,5) a(11,16,13) a(12,2,13) a(12,3,12) a(12,14,9) a(12,15,11) a(13,2,5) a(13,3,3)
a(13,5,2) a(13,6,16) a(13,11,13) a(13,12,10) a(13,14,12) a(13,15,9) a(14,1,8) a(14,4,4)
a(14,5,12) a(14,8,1) a(14,9,6) a(14,12,7) a(14,13,15) a(14,16,3) a(15,1,10) a(15,2,1)
a(15,6,15) a(15,11,16) a(15,15,6) a(15,16,2) a(16,1,11) a(16,2,2) a(16,6,8) a(16,8,14)
a(16,9,3) a(16,11,1) a(16,15,10) a(16,16,7) a(2,3,1) a(4,4,2) a(2,4,3) a(3,3,5) a(3,2,10)
a(4,1,13) a(1,3,11) a(1,4,8) a(4,7,1) a(2,7,2) a(4,8,8) a(1,5,10) a(3,6,11) a(3,7,12) a(2,8,15)
a(2,5,4) a(1,7,16) a(3,10,3) a(2,10,7) a(2,12,8) a(4,10,12) a(3,11,14) a(4,9,5) a(2,9,9)
a(1,10,13) a(1,12,4) a(3,15,8) a(4,16,10) a(1,13,12) a(2,14,13) a(3,14,15) a(4,13,4)
a(1,14,6) a(2,13,16) a(8,4,1) a(7,3,2) a(5,1,3) a(8,3,4) a(6,2,6) a(6,4,9) a(8,2,11) a(7,1,12)
a(5,4,14) a(8,6,2) a(8,8,7) a(5,5,8) a(6,6,12) a(6,8,16) a(7,5,1) a(5,6,13) a(5,7,6) a(5,8,10)
a(5,9,4) a(6,11,7) a(8,11,12) a(7,12,16) a(5,10,1) a(6,9,2) a(8,9,8) a(5,11,5) a(5,12,11)
a(6,15,4) a(7,16,5) a(7,14,7) a(5,13,9) a(8,14,10) a(8,15,15) a(6,13,11) a(8,13,13)
a(5,16,12) a(11,2,4) a(10,3,10) a(11,4,11) a(12,4,16) a(12,1,1) a(9,2,3) a(10,1,14) a(9,3,7)
a(9,4,5) a(12,8,2) a(12,6,4) a(11,6,6) a(12,7,8) a(11,8,9) a(9,6,1) a(12,5,7) a(10,5,11)
a(9,8,12) a(10,12,2) a(11,11,3) a(12,10,5) a(12,11,6) a(9,9,11) a(12,12,14) a(11,9,7)
a(12,9,10) a(9,11,8) a(11,15,1) a(9,13,6) a(10,14,8) a(12,16,15) a(10,16,16) a(12,13,3)
a(9,14,4) a(11,13,10) a(9,15,2) a(16,3,6) a(13,1,7) a(15,3,9) a(16,4,12) a(15,4,13)
a(14,3,14) a(14,2,16) a(13,4,15) a(15,8,4) a(14,7,5) a(15,7,7) a(16,7,9) a(14,6,10) a(15,5,3)
a(16,5,13) a(13,7,11) a(13,8,6) a(14,10,2) a(16,10,4) a(15,12,5) a(14,11,9) a(15,10,11)
a(13,9,14) a(16,12,15) a(13,10,8) a(15,9,12) a(13,13,1) a(14,14,11) a(14,15,13) a(15,14,14)
a(16,14,16) a(16,13,5) a(15,13,8) a(13,16,4)
SATISFIABLE

Models       : 1
Calls        : 1
Time         : 0.164s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.163s
```

Problem 5

Input Program

```
% the facts to represent the given numbers are provided below

a(1,3,7). a(1,7,8).
a(2,2,2). a(2,8,4).
a(3,1,8). a(3,3,4). a(3,5,2). a(3,7,5). a(3,9,1).
a(4,5,7).
a(5,3,8). a(5,4,3). a(5,5,6). a(5,6,4). a(5,7,2).
a(6,5,9).
a(7,1,3). a(7,3,2). a(7,5,8). a(7,7,7). a(7,9,4).
a(8,2,7). a(8,8,8).
a(9,3,6). a(9,7,9).

% write the remaining part of the clingo program below
% Each number 1..9 is assigned to one cell in each box
1 { a(X,Y,N):X=1..9,Y=1..9,X1<=X,X<=X1+2,Y1<=Y,Y<=Y1+2 } 1
            :- N=1..9, X1 = 3*(0..2)+1, Y1 = 3*(0..2)+1.
% no two different numbers given a row and a column
:- a(X,Y,N), a(X,Y,N1), N!=N1.
% no two different columns given a row and a number
:- a(X,Y,N), a(X,Y1,N), Y!=Y1.
% no two different rows given a column and a number
:- a(X,Y,N), a(X1,Y,N), X!=X1.
% no two numbers at same color
:- a(X,Y,N), a(X1,Y1,N),
    X\3 == X1\3, Y\3 == Y1\3,
    1{X != X1; Y != Y1}.
```

Command Line

```
clingo p4.lp 0
```

Output of clingo

```
clingo version 5.6.2
Reading from p4.lp
Solving...
Answer: 1
a(1,3,7) a(1,7,8) a(2,2,2) a(2,8,4) a(3,1,8) a(3,3,4) a(3,5,2) a(3,7,5) a(3,9,1) a(4,5,7) a(5,3,8)
a(5,4,3) a(5,5,6) a(5,6,4) a(5,7,2) a(6,5,9) a(7,1,3) a(7,3,2) a(7,5,8) a(7,7,7) a(7,9,4) a(8,2,7)
a(8,8,8) a(9,3,6) a(9,7,9) a(4,3,1) a(4,6,8) a(4,9,6) a(7,6,5) a(4,1,2) a(4,4,5) a(4,7,4) a(7,4,9)
a(5,2,9) a(5,8,1) a(8,5,3) a(6,1,6) a(6,4,1) a(6,7,3) a(9,1,4) a(9,4,2) a(6,3,5) a(6,6,2) a(6,9,8)
a(9,6,7) a(9,9,3) a(6,2,4) a(6,8,7) a(9,2,8) a(9,5,1) a(9,8,5) a(1,2,5) a(1,5,4) a(1,8,2) a(7,2,1)
a(7,8,6) a(2,3,3) a(2,6,1) a(2,9,7) a(8,3,9) a(8,6,6) a(8,9,2) a(2,1,9) a(2,4,8) a(2,7,6) a(8,1,5)
a(8,4,4) a(8,7,1) a(2,5,5) a(3,2,6) a(3,8,3) a(1,1,1) a(1,4,6) a(1,6,3) a(1,9,9) a(4,2,3) a(4,8,9)
a(3,6,9) a(3,4,7) a(5,1,7) a(5,9,5)
SATISFIABLE

Models       : 1
Calls        : 1
Time         : 0.037s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.037s
```

Problem 6

Input Program

```
% the facts to represent the given numbers are provided below

a(1,1,3). a(1,9,4).
a(2,4,6). a(2,6,9).
a(3,3,6). a(3,7,9).
a(4,2,8). a(4,4,3). a(4,6,2). a(4,8,6).
a(5,5,7).
a(6,2,1). a(6,4,8). a(6,6,5). a(6,8,7).
a(7,3,7). a(7,7,8).
a(8,4,7). a(8,6,8).
a(9,1,9). a(9,9,7).

% write the remaining part of the clingo program below
% Each number 1..9 is assigned to one cell in each box
1 { a(X,Y,N):X=1..9,Y=1..9,X1<=X,X<=X1+2,Y1<=Y,Y<=Y1+2 } 1
            :- N=1..9, X1 = 3*(0..2)+1, Y1 = 3*(0..2)+1.
% no two different numbers given a row and a column
:- a(X,Y,N), a(X,Y,N1), N!=N1.
% no two different columns given a row and a number
:- a(X,Y,N), a(X,Y1,N), Y!=Y1.
% no two different rows given a column and a number
:- a(X,Y,N), a(X1,Y,N), X!=X1.
% solve anti-knight condition
:- a(R,C,N), a(R1,C1,N), |R1-R|+|C1-C|==3.
```

Command Line

```
clingo p4.lp 0
```

Output of clingo

```
clingo version 5.6.2
Reading from p4.lp
Solving...
Answer: 1
a(1,1,3) a(1,9,4) a(2,4,6) a(2,6,9) a(3,3,6) a(3,7,9) a(4,2,8) a(4,4,3) a(4,6,2) a(4,8,6) a(5,5,7)
a(6,2,1) a(6,4,8) a(6,6,5) a(6,8,7) a(7,3,7) a(7,7,8) a(8,4,7) a(8,6,8) a(9,1,9) a(9,9,7) a(1,3,1)
a(3,6,1) a(4,5,1) a(1,5,2) a(2,2,2) a(6,1,2) a(3,5,3) a(5,3,3) a(2,1,4) a(3,4,4) a(6,3,4) a(2,3,5)
a(1,4,5) a(5,2,5) a(5,1,6) a(3,2,7) a(4,1,7) a(1,6,7) a(3,1,8) a(2,5,8) a(1,2,9) a(4,3,9) a(5,4,9)
a(2,9,1) a(3,9,2) a(5,7,2) a(2,8,3) a(5,6,4) a(4,7,4) a(3,8,5) a(1,7,6) a(6,5,6) a(2,7,7) a(1,8,8)
a(5,8,1) a(6,7,3) a(4,9,5) a(5,9,8) a(6,9,9) a(7,1,1) a(8,3,2) a(7,4,2) a(9,2,3) a(7,2,4) a(8,1,5)
a(7,5,5) a(8,2,6) a(9,3,8) a(9,4,1) a(8,7,1) a(7,6,3) a(9,5,4) a(9,6,6) a(8,5,9) a(7,8,9) a(9,8,2)
a(8,9,3) a(8,8,4) a(9,7,5) a(7,9,6)
SATISFIABLE

Models       : 1
Calls        : 1
Time         : 0.028s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.028s
```

Problem 7

Input Program

```
% In general, it would be time-consuming to enumerate a lot of similar facts in a domain
% thus we always use rules to help us generate those facts.
% In problem 7, it’s difficult to enumerate all greater-than relations without a mistake
% thus we provide some rules that you can start from.

% =================================================================

% define the directions of the greater-than symbols on each row
left_right(1, l, l, r, l, l, l).
left_right(2, l, r, l, r, l, l).
left_right(3, r, l, r, r, r, l).
left_right(4, r, l, l, l, r, l).
left_right(5, l, r, r, l, r, r).
left_right(6, l, r, l, l, l, r).
left_right(7, r, l, r, l, l, r).
left_right(8, r, r, l, r, l, l).
left_right(9, l, r, l, r, l, l).

up_down(1, u, u, d, u, u, d, d, d, d).
up_down(2, u, d, d, d, d, d, u, u, d).
up_down(4, d, u, d, u, u, d, u, u, u).
up_down(5, u, d, u, d, d, u, u, u, u).
up_down(7, d, u, d, d, u, d, d, u, u).
up_down(8, d, u, u, u, u, d, d, d, d).

% define the effect of the greater-than symbols on left-right directions
N1 < N2 :- left_right(R, D, _, _, _, _, _), D=l, a(R,1,N1), a(R,2,N2).
N1 < N2 :- left_right(R, _, D, _, _, _, _), D=l, a(R,2,N1), a(R,3,N2).
N1 < N2 :- left_right(R, _, _, D, _, _, _), D=l, a(R,4,N1), a(R,5,N2).
N1 < N2 :- left_right(R, _, _, _, D, _, _), D=l, a(R,5,N1), a(R,6,N2).
N1 < N2 :- left_right(R, _, _, _, _, D, _), D=l, a(R,7,N1), a(R,8,N2).
N1 < N2 :- left_right(R, _, _, _, _, _, D), D=l, a(R,8,N1), a(R,9,N2).

N1 > N2 :- left_right(R, D, _, _, _, _, _), D=r, a(R,1,N1), a(R,2,N2).
N1 > N2 :- left_right(R, _, D, _, _, _, _), D=r, a(R,2,N1), a(R,3,N2).
N1 > N2 :- left_right(R, _, _, D, _, _, _), D=r, a(R,4,N1), a(R,5,N2).
N1 > N2 :- left_right(R, _, _, _, D, _, _), D=r, a(R,5,N1), a(R,6,N2).
N1 > N2 :- left_right(R, _, _, _, _, D, _), D=r, a(R,7,N1), a(R,8,N2).
N1 > N2 :- left_right(R, _, _, _, _, _, D), D=r, a(R,8,N1), a(R,9,N2).

% you need to write down the remaining program below. For example, the first rule
% should “define the effect of the greater-than symbols on up-down directions”
% Each number 1..9 is assigned to one cell in each box
1 { a(X,Y,N):X=1..9,Y=1..9,X1<=X,X<=X1+2,Y1<=Y,Y<=Y1+2 } 1
            :- N=1..9, X1 = 3*(0..2)+1, Y1 = 3*(0..2)+1.
% no two different numbers given a row and a column
:- a(X,Y,N), a(X,Y,N1), N!=N1.
% no two different columns given a row and a number
:- a(X,Y,N), a(X,Y1,N), Y!=Y1.
% no two different rows given a column and a number
```

```
:- a(X,Y,N), a(X1,Y,N), X!=X1.
% solve greater-than sudoku condition
:- a(X,Y,N), a(X1,Y1,N1), gt(X,Y,X1,Y1), N <= N1.
```

Command Line

```
clingo p4.lp
```

Output of clingo

```
clingo version 5.6.2
Reading from p4.lp
p4.lp:48:27-40: info: atom does not occur in any rule head:
  gt(X,Y,X1,Y1)

Solving...
Answer: 1
left_right(1,l,l,r,l,l,l) left_right(2,l,r,l,r,l,l) left_right(3,r,l,r,r,r,l) left_right(4,r,l,l,l,r,l)
left_right(5,l,r,r,l,r,r) left_right(6,l,r,l,l,l,r) left_right(7,r,l,r,l,l,r) left_right(8,r,r,l,r,l,l)
left_right(9,l,r,l,r,l,l) up_down(1,u,u,d,u,u,d,d,d,d) up_down(2,u,d,d,d,d,d,u,u,d)
up_down(4,d,u,d,u,u,d,u,u,u) up_down(5,u,d,u,d,d,u,u,u,u) up_down(7,d,u,d,d,u,d,d,u,u)
up_down(8,d,u,u,u,u,d,d,d,d) a(2,1,1) a(4,2,1) a(8,3,1) a(6,1,2) a(8,2,2) a(2,3,2) a(9,1,3)
a(3,2,3) a(5,3,3) a(3,1,4) a(7,2,4) a(6,3,4) a(7,1,5) a(6,2,5) a(3,3,5) a(5,1,6) a(2,2,6) a(7,3,6)
a(1,1,7) a(5,2,7) a(9,3,7) a(8,1,8) a(1,2,8) a(4,3,8) a(4,1,9) a(9,2,9) a(1,3,9) a(6,4,1) a(1,5,1)
a(9,6,1) a(9,4,2) a(5,5,2) a(3,6,2) a(4,4,3) a(7,5,3) a(2,6,3) a(1,4,4) a(9,5,4) a(5,6,4) a(2,4,5)
a(4,5,5) a(8,6,5) a(8,4,6) a(6,5,6) a(1,6,6) a(7,4,7) a(3,5,7) a(4,6,7) a(5,4,8) a(2,5,8) a(7,6,8)
a(3,4,9) a(8,5,9) a(6,6,9) a(7,7,1) a(3,8,1) a(5,9,1) a(1,7,2) a(4,8,2) a(7,9,2) a(8,7,3) a(1,8,3)
a(6,9,3) a(2,7,4) a(8,8,4) a(4,9,4) a(9,7,5) a(5,8,5) a(1,9,5) a(4,7,6) a(9,8,6) a(3,9,6) a(6,7,7)
a(2,8,7) a(8,9,7) a(3,7,8) a(6,8,8) a(9,9,8) a(5,7,9) a(7,8,9) a(2,9,9)
SATISFIABLE

Models       : 1+
Calls        : 1
Time         : 0.062s (Solving: 0.03s 1st Model: 0.03s Unsat: 0.00s)
CPU Time     : 0.061s
```

Problem 8

Input Program

```
% range of squares
size(n).
square(1..n).

% each square can hold at most one bishop
0 { bishop(X,Y) : square(X), square(Y) } 1.

% no two bishops can attack each other
:- bishop(X1,Y1), bishop(X2,Y2), X1-Y1 == X2-Y2.
:- bishop(X1,Y1), bishop(X2,Y2), X1+Y1 == X2+Y2.

% maximize the number of bishops
#maximize { 1, bishop(X,Y) : square(X), square(Y) }.

% show the bishops
#show bishop/2.
```

Command Line

You should write 6 command lines below, one for each value of n from the set {3, …, 8}.
Hint 1: you do not need to find all stable models.
Hint 2: the value of k (which denotes the number of bishops) in each command line should be the biggest value that you can assign to have at least one stable model.

Output of clingo

Since there are 6 outputs for 6 different values of n, please do not copy and paste the outputs of clingo for Problem 8 (and also for Problem 2 earlier).

Answer to Questions

Draw a table that lists the maximum value of bishops when the chessboard is n by n, where n is 3, 4, 5, 6, 7, 8. Infer the general function f(n) for n >= 3 that returns the maximum value of bishops.

Value of n	f(n)
3	4
4	6
5	8
6	10
7	12
8	14

f(n) = 2*(n-1)

Problem 9

Input Program

```
% Partition {1,..,n} into k sum-free sets
% input: positive integers n, k.
% partition set {1,...,n} into k subsets
% in(I,S) means number I is in set S

{in(I,1..k)} = 1 :- I=1..n.
% these subsets are sum-free.
:- in(I,S), in(J,S), in(I+J,S).
```

Command Line

You should write 4 command lines below, one for each value of k from the set {1, 2, 3, 4}.
Hint 1: you do not need to find all stable models.
Hint 2: the value of n in each command line should be the biggest value that you can assign to have at least one stable model.

```
cat p4.lp | clingo -c k=1 -c n=1
cat p4.lp | clingo -c k=2 -c n=4
cat p4.lp | clingo -c k=3 -c n=13
cat p4.lp | clingo -c k=4 -c n=44
```

Output of clingo

There are 4 outputs for 4 different values of k. Please copy and paste the outputs for k in {1, 2, 3} in the following table.

Value of k | **Output**

1

```
clingo version 5.6.2
Reading from stdin
Solving...
Answer: 1
in(1,1)
SATISFIABLE
```

2

```
clingo version 5.6.2
Reading from stdin
Solving...
Answer: 1
in(1,1) in(2,2) in(3,2) in(4,1)
SATISFIABLE
```

3

```
clingo version 5.6.2
Reading from stdin
Solving...
Answer: 1
in(1,1) in(2,2) in(3,2) in(4,1) in(5,3) in(6,3) in(7,1) in(8,3) in(9,3) in(10,1) in(11,2)
in(12,2) in(13,1)
SATISFIABLE
```

Answer to Questions

Fill in the values accordingly.

Exact value of A(1)	1
Exact value of A(2)	4
Exact value of A(3)	13
Largest lower bound for A(4)	44

		Note: it would take longer time when you increase the value of n. Thus, you may stop increasing the value of n when your program does not terminate within 10 minutes and submit the last trial of n.		