**CSE 579**

**Programming Assignment 1**

**Template for clingo Work**

Problem 1

|  |  |
| --- | --- |
| Input  Program | % Define the size of the chessboard  size(8).  % Generate positions for queens ensuring they are not in the middle 4x4 square  { queen(X,Y) : X = 1..8, Y = 1..8, not middle(X,Y) } = 8.  % Define the middle 4x4 squares  middle(3..6,3..6).  % Constraint: No two queens on the same row  :- queen(X,Y1), queen(X,Y2), Y1 != Y2.  % Constraint: No two queens on the same column  :- queen(X1,Y), queen(X2,Y), X1 != X2.  % Constraint: No two queens on the same diagonal  :- queen(X1,Y1), queen(X2,Y2), X1 != X2, abs(X1-X2) = abs(Y1-Y2).  % Display the solution.  #show queen/2. |
| Command  Line | clingo ./program\_1.txt 0 |
| Output  of clingo |  |

Problem 2

|  |  |
| --- | --- |
| Input  Program | #const n = 10.  % Define the cells of the chessboard in which the Queens can be placed.  {queen\_cell(R, 1..n)} = 1 :- R = 1..n.  % 2 queens not in the similar column  :- queen\_cell(R1, C), queen\_cell(R2, C), R1 != R2.  % queens not in diagonal  :- queen\_cell(R1, C1), queen\_cell(R2, C2), R1 != R2, |R1 - R2| = |C1 - C2|. |
| Command  Line | clingo -c n=3 p2.txt 0  clingo -c n=4 p2.txt 0  clingo -c n=5 p2.txt 0  clingo -c n=6 p2.txt 0  clingo -c n=7 p2.txt 0  clingo -c n=8 p2.txt 0  clingo -c n=9 p2.txt 0  clingo -c n=10 p2.txt 0  clingo -c n=11 p2.txt 0  clingo -c n=12 p2.txt 0 |
| Output  of clingo | - |
| 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 (in sec) | | 3 | 0 | 0.001 | | 4 | 2 | 0.001 | | 5 | 10 | 0.001 | | 6 | 4 | 0.001 | | 7 | 40 | 0.002 | | 8 | 92 | 0.004 | | 9 | 352 | 0.02 | | 10 | 724 | 0.1 | | 11 | 2680 | 0.79 | | 12 | 14200 | 9.25 | |

Problem 3

|  |  |
| --- | --- |
| Input  Program | % Define the Sudoku board.  1 {cell(R, C, V): V = 1..9} 1 :- R = 1..9, C = 1..9.  % Given initial values on the Sudoku board.  cell(1,1,1). cell(1,6,7). cell(1,8,9).  cell(2,2,3). cell(2,5,2). cell(2,9,8).  cell(3,3,9). cell(3,4,6). cell(3,7,5).  cell(4,3,5). cell(4,4,3). cell(4,7,9).  cell(5,2,1). cell(5,5,8). cell(5,9,2).  cell(6,1,6). cell(6,6,4).  cell(7,1,3). cell(7,8,1).  cell(8,2,4). cell(8,9,7).  cell(9,3,7). cell(9,7,3).  % No two cells on the same column can have the same value.  :- cell(R1, C, V), cell(R2, C, V), R1 != R2.  % No two cells on the same row can have the same value.  :- cell(R, C1, V), cell(R, C2 ,V), C1 != C2.  % No two cell in the same subgrid can have the same value.  :- cell(R1, C1, V), cell(R2, C2, V), (R1-1)/3 == (R2-1)/3, (C1-1)/3 == (C2-1)/3, R1 != R2, C1 != C2.  #show cell/3. |
| Command  Line | clingo ./program\_3.txt 0 |
| Output  of clingo |  |

Problem 4

|  |  |
| --- | --- |
| Input  Program |  |
| Command  Line |  |
| Output  of clingo |  |

Problem 5

|  |  |
| --- | --- |
| Input  Program |  |
| Command  Line |  |
| Output  of clingo |  |

Problem 6

|  |  |
| --- | --- |
| Input  Program |  |
| Command  Line |  |
| Output  of clingo |  |

Problem 7

|  |  |
| --- | --- |
| Input  Program |  |
| Command  Line |  |
| Output  of clingo |  |

Problem 8

|  |  |
| --- | --- |
| Input  Program |  |
| Command  Line | You should write multiple command lines below. |
| Output  of clingo |  |
| 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) that returns the maximum value of bishops.   |  |  | | --- | --- | | Value n | f(n) | | 3 |  | | 4 |  | | 5 |  | | 6 |  | | 7 |  | | 8 |  |   f(n) = |

Problem 9

|  |  |
| --- | --- |
| Input  Program |  |
| Command  Line | You should write multiple command lines below. |
| Output  of clingo |  |
| Answer  to Questions | Fill in the values accordingly.   |  |  | | --- | --- | | Exact value of A(1) |  | | Exact value of A(2) |  | | Exact value of A(3) |  | | Largest lower bound for A(4)  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. |  | |