CSE 579
Programming Assignment 2

Problem 1: Consider the clingo program blocks.lp below that is introduced in the Blocks World in ASP lecture. (completed version of the code in the table).

Modify the file blocks.lp to reflect the assumption that the table is small, so that the number of blocks that can be placed on the table simultaneously is limited by a given constant. How many steps are required to solve the example problem above if only 4 blocks can be on the table at the same time? What if only 3? You may test your codes with a scenario, which is also represented by a clingo program such as blocks-scenario.lp below.

Input Program	Hint: you only need one program with a new term, whose value will be assigned to 3 or 4 in the command line. %%%%%%%%%%%%%%%%%% % File: pa2_1.txt: Blocks World %%%%%%%%%%%%%%%%%% %%%%%%%%%%%%%%%%%%%%%%%%%%%% % sort and object declaration %%%%%%%%%%%%%%%%%%%%%%%%%%%% % every block is a location location(B) :- block(B). % the table is a location location(table). %%%%%%%%%%%%%%%%%%%%%%%%% % state description %%%%%%%%%%%%%%%%%%%%%%%%% % two blocks can't be on the same block at the same time :- 2{on(BB,B,T)}, block(B), T = 0..m. %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% % effect and preconditions of action %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% % effect of moving a block on(B,L,T+1) :- move(B,L,T). % concurrent actions are limited by num of grippers :- not {move(BB,LL,T)} grippers, T = 0..m-1.

	% a block can be moved only when it is clear :- move(B,L,T), on(B1,B,T). % a block can't be moved onto a block that is being moved also :- move(B,B1,T), move(B1,L,T). %%%%%%%%%%%%%%%%%%%%%%%%%%%% % domain independent axioms %%%%%%%%%%%%%%%%%%%%%%%%%%%% % fluents are initially exogenous 1{on(B,LL,0):location(LL)}1 :- block(B). % uniqueness and existence of value constraints :- not 1{on(B,LL,T)}1, block(B), T=1..m. % actions are exogenous {move(B,L,T)} :- block(B), location(L), T = 0..m-1. % commonsense law of inertia {on(B,L,T+1)} :- on(B,L,T), T < m. % Ensure the number of blocks on the table does not exceed the available space (n) at any time T :- not {on(B,table,T)}n, T=0..m. #show move/3. %%%%%%%%%%%%%%%%%%% % File: scenario.txt %%%%%%%%%%%%%%%%%%% block(1..6). % initial state :- not on(1,2,0; 2,table,0; 3,4,0; 4,table,0; 5,6,0; 6,table,0). % goal :- not on(3,2,m; 2,1,m; 1,table,m; 6,5,m; 5,4,m; 4,table,m).
Command Lines	You should write multiple command lines below. clingo pa2_1.txt scenario.txt -c m=4 -c n=3 0 clingo pa2_1.txt scenario.txt -c m=5 -c n=3 0 clingo pa2_1.txt scenario.txt -c m=2 -c n=4 0 clingo pa2_1.txt scenario.txt -c m=3 -c n=4 0
Outputs of clingo	You should write multiple outputs, one for each command. These outputs serve as the evidences of your answer to the following question.

Hint 1: Let n be the maximal number of blocks that can be placed directly on the table. There should be 2 command lines and outputs for n=3, where

- the 1st command line and output show k steps are not enough and
- the 2nd command line and output show k+1 steps are enough.

Similarly, there should be another 2 command lines and outputs for n=4.

Hint 2: We do not give any limitation on the number of grippers.

For m=4 and n=3

```
E:\ASU\CLASSES\Spring '24\KRR\Project\clingo-5.4.0-win64>clingo pa2_1.txt
 scenario.txt -c m=4 -c n=3 0
clingo version 5.4.0
Reading from pa2_1.txt ...
Solving...
UNSATISFIABLE

Models       : 0
Calls        : 1
Time         : 0.038s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.031s
```

For m= 5 and n = 3

```
E:\ASU\CLASSES\Spring '24\KRR\Project\clingo-5.4.0-win64>clingo pa2_1.txt
 scenario.txt -c m=5 -c n=3 0
clingo version 5.4.0
Reading from pa2_1.txt ...
Solving...
Answer: 1
move(3,1,0) move(5,4,0) move(3,table,1) move(6,5,1) move(1,table,2) move(
3,6,2) move(2,1,3) move(3,table,3) move(3,2,4)
Answer: 2
move(3,1,0) move(5,4,0) move(3,table,1) move(6,5,1) move(1,table,2) move(
3,6,2) move(2,1,3) move(3,2,4)
```

```
Answer: 135
move(3,6,0) move(5,4,0) move(1,2,1) move(3,5,1) move(1,table,2) move(6,3,
2) move(2,1,3) move(6,table,3) move(3,2,4) move(6,5,4)
Answer: 136
move(1,2,0) move(3,6,0) move(5,4,0) move(1,2,1) move(3,5,1) move(1,table,
2) move(6,3,2) move(2,1,3) move(6,table,3) move(3,2,4) move(6,5,4)
SATISFIABLE

Models       : 136
Calls        : 1
Time         : 0.173s (Solving: 0.16s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.016s
```

	For m = 2 and n = 4 ``` E:\ASU\CLASSES\Spring '24\KRR\Project\clingo-5.4.0-win64>clingo pa2_1.txt scenario.txt -c m=2 -c n=4 0 clingo version 5.4.0 Reading from pa2_1.txt ... Solving... UNSATISFIABLE Models : 0 Calls : 1 Time : 0.003s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s) CPU Time : 0.000s ``` **For m = 3 and n = 4** ``` E:\ASU\CLASSES\Spring '24\KRR\Project\clingo-5.4.0-win64>clingo pa2_1.txt scenario.txt -c m=3 -c n=4 0 clingo version 5.4.0 Reading from pa2_1.txt ... Solving... Answer: 1 move(1,table,0) move(3,6,0) move(5,4,0) move(2,1,1) move(3,table,1) move(3,2,2) move(6,5,2) Answer: 2 move(1,table,0) move(3,6,0) move(5,4,0) move(2,1,1) move(3,table,1) move(5,4,1) move(3,2,2) move(6,5,2) ``` ``` Answer: 6 move(1,table,0) move(3,4,0) move(2,1,1) move(3,table,1) move(5,4,1) move(3,2,2) move(6,5,2) Answer: 7 move(1,table,0) move(3,4,0) move(5,6,0) move(2,1,1) move(3,table,1) move(5,4,1) move(3,2,2) move(6,5,2) SATISFIABLE Models : 7 Calls : 1 Time : 0.047s (Solving: 0.02s 1st Model: 0.00s Unsat: 0.00s) CPU Time : 0.031s ```
Answer to Questions	Fill in the following table that lists the minimum number of steps to solve the modified block world problem for different values of n, where n is the maximal number of blocks that can be placed directly on the table. <table><tr><td>n</td><td>Number of steps</td></tr><tr><td>3</td><td>5</td></tr><tr><td>4</td><td>3</td></tr></table>

Problem 2: The file blocks.lp above specifies that the initial state correctly. Without the specification, there will be stable models that do not correspond to valid states, like the following.

on(1,2,0) on(2,1,0) on(3,3,0) on(4,table,0) on(5,6,0) on(6,table,0) Modify the file blocks.lp so that the stable models are in a 1-1 correspondence with valid states.

How many valid states are there?

Input Program	Hint 1: You don't need to represent any scenario since you want to find out all possible valid states with 6 blocks. Think about the value of m. Hint 2: You don't need to consider the limitation of the maximum number of blocks on the table. That's only required in Problem 1. %%%%%%%%%%%%%%%%%%% % File: pa2_2.txt: Blocks World %%%%%%%%%%%%%%%%%%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%% % sort and object declaration %%%%%%%%%%%%%%%%%%%%%%%%%%%%%% % every block is a location location(B) :- block(B). % the table is a location location(table). %%%%%%%%%%%%%%%%%%%%%%%%%% % state description %%%%%%%%%%%%%%%%%%%%%%%%%% % two blocks can't be on the same block at the same time :- 2{on(BB,B,T)}, block(B), T = 0..m. %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% % effect and preconditions of action %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% % effect of moving a block on(B,L,T+1) :- move(B,L,T). % concurrent actions are limited by num of grippers :- not {move(BB,LL,T)} grippers, T = 0..m-1. % a block can be moved only when it is clear

<table>
<tr><td></td><td>

```
:- move(B,L,T), on(B1,B,T).

% a block can't be moved onto a block that is being moved also
:- move(B,B1,T), move(B1,L,T).

%%%%%%%%%%%%%%%%%%%%%%%%%%%
% domain independent axioms
%%%%%%%%%%%%%%%%%%%%%%%%%%%

% fluents are initially exogenous
1{on(B,LL,0):location(LL)}1 :- block(B).

% uniqueness and existence of value constraints
:- not 1{on(B,LL,T)}1, block(B), T=1..m.

% actions are exogenous
{move(B,L,T)} :- block(B), location(L), T = 0..m-1.

% commonsense law of inertia
{on(B,L,T+1)} :- on(B,L,T), T < m.

% two same blocks cannot be top of one another
below(P,Q,T) :- on(Q,P,T).
below(P,R,T) :- below(P,Q,T), below(Q,R,T).
:- on(P,P,T).
:- on(P,Q,T), below(P,Q,T).

block(1..6).
#show move/3.
```

</td></tr>
<tr><td>Command Line</td><td>clingo p2_2.txt -c l=3 -c grippers=2 -c m=0 0</td></tr>
<tr><td>Output of clingo</td><td>

```
E:\ASU\CLASSES\Spring '24\KRR\Project\clingo-5.4.0-win64>clingo p2_2.txt -c l=3 -c grippers=2 -c m=0 0
clingo version 5.4.0
Reading from p2_2.txt
Solving...
Answer: 1

Answer: 2

Answer: 3

Answer: 4
```

</td></tr>
</table>

<table>
<tr><td></td><td>

```
Answer: 4049

Answer: 4050

Answer: 4051

SATISFIABLE

Models       : 4051
Calls        : 1
Time         : 0.803s (Solving: 0.79s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.047s
```

</td></tr>
<tr><td>Answer
to Questions</td><td>How many valid states are there when there are 6 blocks? (Note that the limitation of blocks introduced in question 1 is not considered here.)
4051 valid states</td></tr>
</table>

Problem 3: A serializable plan is such that the actions that are scheduled for the same time period can be instead executed consecutively, in any order without affecting the result.
Modify blocks.lp to generate only serializable plans. Find a minimal length plan for the following scenario. (Hint: you need to modify blocks-scenario.lp to reflect this new scenario.)

Initially:
loc(m)=table, loc(l)=m, loc(a)=l, loc(b)=a, loc(c)=b,
loc(o)=table, loc(n)=o, loc(d)=n, loc(e)=d, loc(j)=e,
loc(k)=j, loc(f)=table, loc(g)=f, loc(h)=g, loc(i)=h
In maxstep:
loc(e)=j, loc(a)= e, loc(n)=a, loc(i)=d, loc(h)=i,
loc(m)=h, loc(o)= m, loc(k)=g, loc(c)=k, loc(b)=c,
loc(l)=b.

Reading: A plan may allow multiple actions happening at the same time, e.g., when we have multiple robots working together to increase efficiency. However, if there is a little bit delay on one action, then we may get unexpected results. For example, when 2 robots are moving 2 adjacent blocks to the left at the same time, if there is a delay for the robot on the left-hand side, then these 2 robots may hit with each other. To make sure that our plan will get the expected result, we introduce the restriction "serializable" on the actions happening at the same time. This restriction simply says that, even if some actions in the same time stamp happen in serial with arbitrary ordering, the final result would be the same.

Input Program	Hint: the number of grippers is unlimited, meaning that you can have as many movements as you want as far as the movements are serializable. %%%%%%%%%%%%%%%%%% % File: pa2_3.txt %%%%%%%%%%%%%%%%%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%% % sort and object declaration %%%%%%%%%%%%%%%%%%%%%%%%%%%%% % every block is a location location(B) :- block(B). % the table is a location location(table). %%%%%%%%%%%%%%%%%%%%%%%%%% % state description %%%%%%%%%%%%%%%%%%%%%%%%%% % two blocks can't be on the same block at the same time :- 2{on(BB,B,T)}, block(B), T = 0..m.

<table>
<tr><td></td><td>%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% effect and preconditions of action
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% effect of moving a block
on(B,L,T+1) :- move(B,L,T).
% concurrent actions are limited by num of grippers
:- not {move(BB,LL,T)} grippers, T = 0..m-1.
% a block can be moved only when it is clear
:- move(B,L,T), on(B1,B,T).
% a block can't be moved onto a block that is being moved also
:- move(B,B1,T), move(B1,L,T).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% domain independent axioms
%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% fluents are initially exogenous
1{on(B,LL,0):location(LL)}1 :- block(B).
% uniqueness and existence of value constraints
:- not 1{on(B,LL,T)}1, block(B), T=1..m.
% actions are exogenous
{move(B,L,T)} :- block(B), location(L), T = 0..m-1.
% commonsense law of inertia
{on(B,L,T+1)} :- on(B,L,T), T < m.
%constraint for serialization
:- move(B,L,T), move(BB,LL,T), on(BB,L,T), block(B), T=1..m-1.
#show move/3.

%%%%%%%%%%%%%%%%%%%
% File: instances.txt
%%%%%%%%%%%%%%%%%%%

block(a;b;c;d;e;f;g;h;i;j;k;l;m1;n;o).
% initial state
:- not on(m1,table,0; l,m1,0; a,l,0; b,a,0; c,b,0; o,table,0; n,o,0; d,n,0; e,d,0; j,e,0; k,j,0; f,table,0; g,f,0; h,g,0; i,h,0).
% goal
:- not on(e,j,m; a,e,m; n,a,m; i,d,m; h,i,m; m1,h,m; o,m1,m; k,g,m; c,k,m; b,c,m; l,b,m).</td></tr>
<tr><td>Command Line</td><td>Please only show the command line that outputs the minimal length plan.
clingo pa2_3.txt instances.txt -c grippers=10 -c m=8</td></tr>
<tr><td>Output of clingo</td><td></td></tr>
</table>

```
E:\ASU\CLASSES\Spring '24\KRR\Project\clingo-5.4.0-win64>clingo pa2_3.txt instances.txt -c
grippers=10 -c m=8
clingo version 5.4.0
Reading from pa2_3.txt ...
Solving...
Answer: 1
move(c,table,0) move(i,table,0) move(k,table,0) move(b,table,1) move(h,c,1) move(j,table,1)
 move(e,h,2) move(i,b,2) move(k,g,2) move(a,k,3) move(d,table,3) move(e,j,3) move(a,e,4) mo
ve(h,table,4) move(i,d,4) move(l,table,4) move(n,table,4) move(b,a,5) move(c,k,5) move(h,i,
5) move(l,n,5) move(m1,o,5) move(b,c,6) move(l,table,6) move(m1,h,6) move(l,b,7) move(n,a,7
) move(o,m1,7)
SATISFIABLE

Models       : 1+
Calls        : 1
Time         : 0.688s (Solving: 0.03s 1st Model: 0.03s Unsat: 0.00s)
CPU Time     : 0.688s
```

Problem 4: A minimal length plan is not necessarily optimal. Modify the program done for Problem 3 to find a plan that has the least number of actions. What is that number when maxstep m is 8, 9, and 10?

Input Program

```
%%%%%%%%%%%%%%%%%%%
% File: pa2_4.txt
%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% sort and object declaration
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% every block is a location
location(B) :- block(B).
% the table is a location
location(table).

%%%%%%%%%%%%%%%%%%%%%%%%%%
% state description
%%%%%%%%%%%%%%%%%%%%%%%%%%

% two blocks can't be on the same block at the same time
:- 2{on(BB,B,T)}, block(B), T = 0..m.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% effect and preconditions of action
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% effect of moving a block
on(B,L,T+1) :- move(B,L,T).
% concurrent actions are limited by num of grippers
:- not {move(BB,LL,T)} grippers, T = 0..m-1.
```

	% a block can be moved only when it is clear :- move(B,L,T), on(B1,B,T). % a block can't be moved onto a block that is being moved also :- move(B,B1,T), move(B1,L,T). %%%%%%%%%%%%%%%%%%%%%%%%%%% % domain independent axioms %%%%%%%%%%%%%%%%%%%%%%%%%%% % fluents are initially exogenous 1{on(B,LL,0):location(LL)}1 :- block(B). % uniqueness and existence of value constraints :- not 1{on(B,LL,T)}1, block(B), T=1..m. % actions are exogenous {move(B,L,T)} :- block(B), location(L), T = 0..m-1. % commonsense law of inertia {on(B,L,T+1)} :- on(B,L,T), T < m. %constraint for serialization :- move(B,L,T), move(BB,LL,T), on(BB,L,T), block(B), T=1..m-1. #minimize{1,B,L,T:move(B,L,T)}. #show move/3. %%%%%%%%%%%%%%%%%%%% % File: instances.txt %%%%%%%%%%%%%%%%%%%% block(a;b;c;d;e;f;g;h;i;j;k;l;m1;n;o). % initial state :- not on(m1,table,0; l,m1,0; a,l,0; b,a,0; c,b,0; o,table,0; n,o,0; d,n,0; e,d,0; j,e,0; k,j,0; f,table,0; g,f,0; h,g,0; i,h,0). % goal :- not on(e,j,m; a,e,m; n,a,m; i,d,m; h,i,m; m1,h,m; o,m1,m; k,g,m; c,k,m; b,c,m; l,b,m).
Comman d Line	You should write multiple command lines below. clingo pa2_4.txt instances.txt -c grippers=10 -c s=8 clingo pa2_4.txt instances.txt -c grippers=10 -c s=9 clingo pa2_4.txt instances.txt -c grippers=10 -c s=10 -t4
Output of clingo	You should write multiple outputs, one for each command. These outputs serve as the evidences of your answer to the question below. **For m=8**

```
E:\ASU\CLASSES\Spring '24\KRR\Project\clingo-5.4.0-win64>clingo pa2_4.txt instances.txt -c
grippers=10 -c m=8
clingo version 5.4.0
Reading from pa2_4.txt ...
Solving...
Answer: 1
move(c,table,0) move(i,table,0) move(k,table,0) move(b,c,1) move(h,table,1) move(j,i,1) mov
e(a,table,2) move(e,b,2) move(g,table,2) move(j,h,2) move(d,table,3) move(e,table,3) move(j
,table,3) move(l,table,3) move(b,n,4) move(e,j,4) move(i,d,4) move(k,g,4) move(a,e,5) move(
b,f,5) move(c,k,5) move(h,i,5) move(m1,l,5) move(b,c,6) move(m1,h,6) move(n,table,6) move(l
,b,7) move(n,a,7) move(o,m1,7)
Optimization: 29
Answer: 2
move(c,table,0) move(i,table,0) move(k,table,0) move(b,c,1) move(h,table,1) move(j,i,1) mov
e(a,table,2) move(e,b,2) move(g,table,2) move(j,h,2) move(d,table,3) move(e,table,3) move(j
,table,3) move(l,table,3) move(b,f,4) move(e,j,4) move(i,d,4) move(k,g,4) move(a,e,5) move(
c,k,5) move(h,i,5) move(m1,l,5) move(b,c,6) move(m1,h,6) move(n,table,6) move(l,b,7) move(n
,a,7) move(o,m1,7)
Optimization: 28
```

```
Answer: 11
move(i,table,0) move(k,c,0) move(h,table,1) move(j,table,1) move(k,table,1) move(c,h,2) mov
e(e,j,2) move(k,g,2) move(b,table,3) move(c,k,3) move(d,table,3) move(a,e,4) move(b,c,4) mo
ve(i,d,4) move(h,i,5) move(l,b,5) move(n,a,5) move(m1,h,6) move(o,m1,7)
Optimization: 19
Answer: 12
move(i,table,0) move(k,table,0) move(h,table,1) move(j,table,1) move(c,h,2) move(e,j,2) mov
e(k,g,2) move(b,table,3) move(c,k,3) move(d,table,3) move(a,e,4) move(b,c,4) move(i,d,4) mo
ve(h,i,5) move(l,b,5) move(n,a,5) move(m1,h,6) move(o,m1,7)
Optimization: 18
OPTIMUM FOUND

Models       : 12
  Optimum    : yes
Optimization : 18
Calls        : 1
Time         : 1.236s (Solving: 0.63s 1st Model: 0.03s Unsat: 0.35s)
CPU Time     : 1.078s
```

For m = 9

```
E:\ASU\CLASSES\Spring '24\KRR\Project\clingo-5.4.0-win64>clingo pa2_4.txt instances.txt -c
grippers=10 -c m=9
clingo version 5.4.0
Reading from pa2_4.txt ...
Solving...
Answer: 1
move(c,table,0) move(i,table,0) move(k,table,0) move(b,table,1) move(j,table,1) move(a,tabl
e,2) move(e,table,2) move(h,table,2) move(d,table,3) move(g,table,3) move(l,table,3) move(i
,d,4) move(k,g,4) move(c,k,5) move(e,j,5) move(h,i,5) move(n,l,5) move(a,e,6) move(b,c,6) m
ove(m1,h,6) move(n,f,6) move(l,b,7) move(n,a,7) move(o,m1,7)
Optimization: 24
Answer: 2
move(c,table,0) move(i,table,0) move(k,table,0) move(b,table,1) move(h,table,1) move(j,tabl
e,1) move(a,table,2) move(e,table,2) move(d,table,3) move(g,table,3) move(l,table,3) move(i
,d,4) move(k,g,4) move(c,k,5) move(e,j,5) move(h,i,5) move(a,e,6) move(b,c,6) move(m1,h,6)
move(n,f,6) move(l,b,7) move(n,a,7) move(o,m1,7)
Optimization: 23
```

```
Answer: 8
move(i,table,0) move(k,c,0) move(h,table,1) move(j,table,1) move(e,table,2) move(k,g,2) mov
e(c,k,3) move(d,table,3) move(b,c,4) move(e,j,4) move(i,d,4) move(a,e,5) move(h,i,5) move(l
,b,6) move(m1,h,7) move(n,a,7) move(o,m1,8)
Optimization: 17
Answer: 9
move(i,table,0) move(k,c,0) move(h,table,1) move(j,table,1) move(e,j,2) move(k,g,2) move(c,
k,3) move(d,table,3) move(b,c,4) move(i,d,4) move(a,e,5) move(h,i,5) move(l,b,6) move(m1,h,
7) move(n,a,7) move(o,m1,8)
Optimization: 16
OPTIMUM FOUND

Models       : 9
  Optimum    : yes
Optimization : 16
Calls        : 1
Time         : 30.949s (Solving: 30.22s 1st Model: 0.00s Unsat: 6.88s)
CPU Time     : 30.328s
```

For m = 10

```
E:\ASU\CLASSES\Spring '24\KRR\Project\clingo-5.4.0-win64>clingo pa2_4.txt instances.txt -c grippers=10 -c m=10 -t4
clingo version 5.4.0
Reading from pa2_4.txt ...
Solving...
Progression : [1;inf]
Progression : [2;inf]
Answer: 1
move(c,table,0) move(i,table,0) move(k,table,0) move(b,table,1) move(j,table,1) move(a,table,2) move(e,table,2) move(h,ta
ble,2) move(d,table,3) move(g,table,3) move(l,table,4) move(n,table,4) move(i,d,5) move(k,g,5) move(c,k,6) move(e,j,6) mo
ve(h,i,6) move(a,e,7) move(b,c,7) move(m1,h,7) move(l,b,8) move(o,m1,8) move(n,a,9)
Optimization: 23
Progression : [ 3;23] (Error: 6.66667)
Progression : [ 4;23] (Error: 4.75)
Progression : [ 5;23] (Error: 3.6)
Answer: 2
move(c,table,0) move(i,table,0) move(k,table,0) move(b,table,1) move(h,table,1) move(j,table,1) move(a,table,2) move(e,ta
ble,2) move(d,table,3) move(l,table,4) move(n,table,4) move(i,d,5) move(k,g,5) move(c,k,6) move(e,j,6) move(h,i,6) move(a
,e,7) move(b,c,7) move(m1,h,7) move(l,b,8) move(o,m1,8) move(n,a,9)
Optimization: 22
Progression : [ 6;22] (Error: 2.66667)
Progression : [ 7;22] (Error: 2.14286)
Progression : [ 8;22] (Error: 1.75)
Progression : [ 9;22] (Error: 1.44444)
Progression : [10;22] (Error: 1.2)
Answer: 3
move(c,table,0) move(i,table,0) move(h,table,1) move(k,i,1) move(j,table,2) move(k,g,2) move(c,k,3) move(e,j,3) move(b,c,
4) move(d,table,4) move(a,e,5) move(i,d,5) move(h,i,6) move(l,b,7) move(n,table,7) move(m1,h,8) move(n,a,8) move(o,m1,9)
Optimization: 18
Progression : [11;18] (Error: 0.636364)
Progression : [12;18] (Error: 0.5)
Progression : [13;18] (Error: 0.384615)
Progression : [14;18] (Error: 0.285714)
Progression : [15;18] (Error: 0.2)
Answer: 4
move(i,table,0) move(h,table,1) move(k,g,2) move(c,k,3) move(j,table,3) move(b,c,4) move(e,j,4) move(a,e,5) move(d,table,
5) move(i,d,6) move(h,i,7) move(l,b,7) move(n,a,7) move(m1,h,8) move(o,m1,9)
Optimization: 15
OPTIMUM FOUND

Models       : 4
  Optimum    : yes
Optimization : 15
Calls        : 1
Time         : 0.907s (Solving: 0.09s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 1.047s
Threads      : 4        (Winner: 1)
```

Answer to Questions

What is the least number of actions when maxstep m is 8, 9, and 10?

m	least number of actions
8	18
9	16
10	15