```
%% First Program
%-% --------------------- Starts below this line ---------------------

%% The thief had long brown hair and wearing black shoes.

thief(X):- longbrownhair(X), wore(X,blackshoes).

%% A person has long black hair if he/she is staying in room 100.
longbrownhair(X):- stays(X,100).

%% A person has short brown hair if he/she is staying in room 102.
longbrownhair(X):- stays(X,205).

%% A person has long brown hair if he/she is staying in room 205.
longbrownhair(X):- stays(X,210).

%% A person has long brown hair if he/she is staying in room 210.
shortbrownhair(X):- stays(X,102).

%% A person is in room 205 if he/she wore black coat.
stays(X,205):- wore(X,blackcoat).

%% A person is in room 102 if he/she wore blue shirt.
stays(X,102):- wore(X,blueshirt).

%% A person is in room 210 if she wore red gown.
stays(X,210):- wore(X,redgown),female(X).

%% A person wore blue shirt if he was wearing a black tie.
wore(X,blueshirt):- wore(X,blacktie),male(X).

%% A person wore a red gown if she is bridesmaid.
wore(X,redgown):- bridesmaid(X),female(X).

%% A person wore black shoes if she was wearing a silver bracelet.
wore(X,blackshoes):- wore(X,silverbracelet),female(X).

%% A person wore black shoes if he was wearing a black tie.
wore(X,blackshoes):- wore(X,blacktie),male(X).

%% All the below mentioned clauses are fact

%% James was wearing black coat.
```

wore(james,blackcoat).

%% Joe was wearing black shoes.
wore(joe,blackshoes).

%% Jenny was wearing silver bracelet.
wore(jenny,silverbracelet).

%% Jenny is bridesmaid.
bridesmaid(jenny).

%% Joy is bridesmaid.
bridesmaid(joy).

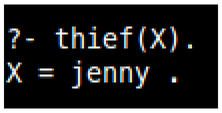
%% Jacy is bridesmaid
bridesmaid(jacy).

%% Although these facts were not mentioned but since rules differentiated between male and female, I inferred the facts and wrote them
female(jenny).
female(joy).
female(jacy).

male(james).
male(joe).

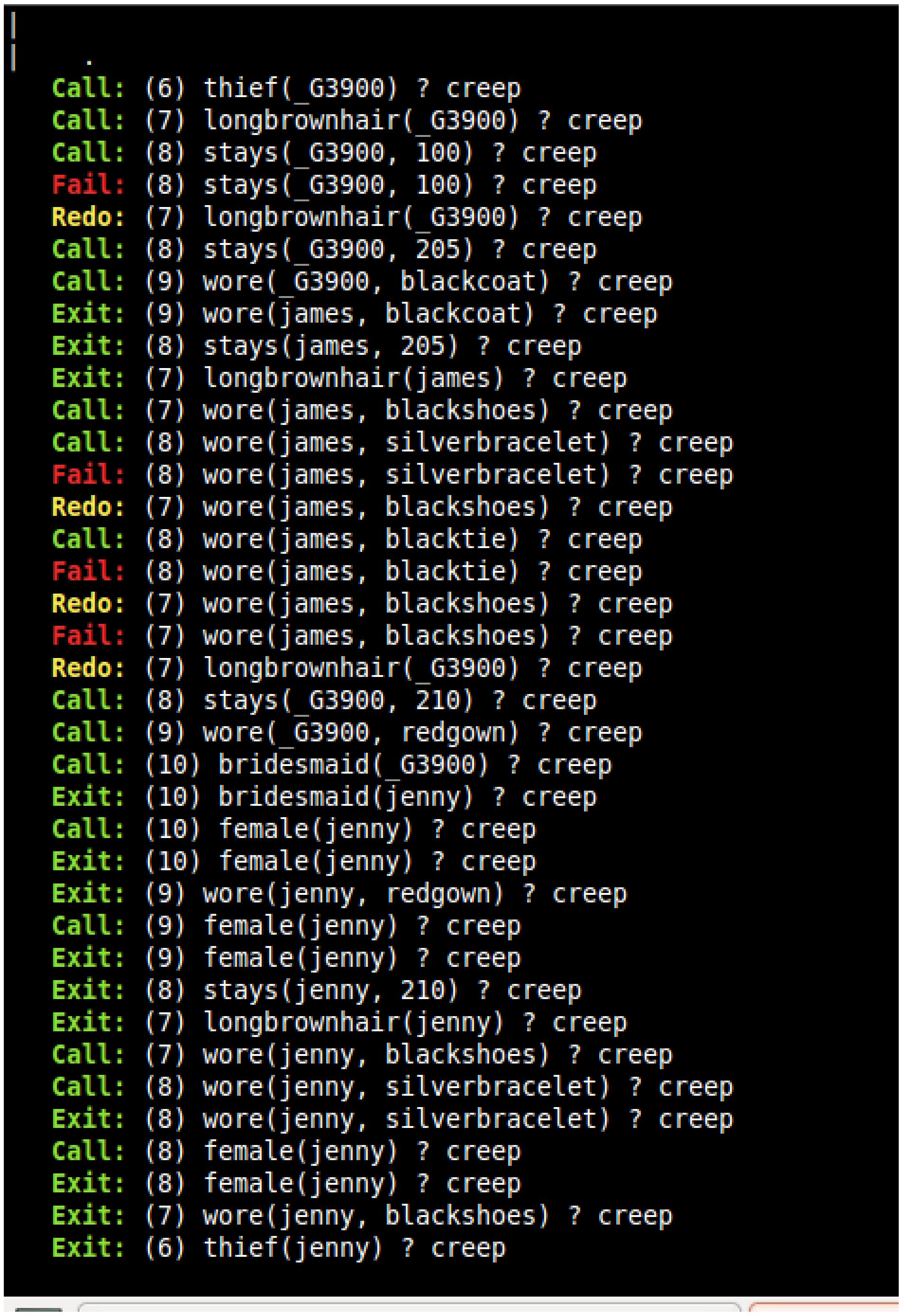
%% ----------------------- Ends ---------------------

Who is thief?

Ans: 

```
?- thief(X).
X = jenny .
```

Trace of first program:



```
|
|    .
   Call: (6) thief(_G3900) ? creep
   Call: (7) longbrownhair(_G3900) ? creep
   Call: (8) stays(_G3900, 100) ? creep
   Fail: (8) stays(_G3900, 100) ? creep
   Redo: (7) longbrownhair(_G3900) ? creep
   Call: (8) stays(_G3900, 205) ? creep
   Call: (9) wore(_G3900, blackcoat) ? creep
   Exit: (9) wore(james, blackcoat) ? creep
   Exit: (8) stays(james, 205) ? creep
   Exit: (7) longbrownhair(james) ? creep
   Call: (7) wore(james, blackshoes) ? creep
   Call: (8) wore(james, silverbracelet) ? creep
   Fail: (8) wore(james, silverbracelet) ? creep
   Redo: (7) wore(james, blackshoes) ? creep
   Call: (8) wore(james, blacktie) ? creep
   Fail: (8) wore(james, blacktie) ? creep
   Redo: (7) wore(james, blackshoes) ? creep
   Fail: (7) wore(james, blackshoes) ? creep
   Redo: (7) longbrownhair(_G3900) ? creep
   Call: (8) stays(_G3900, 210) ? creep
   Call: (9) wore(_G3900, redgown) ? creep
   Call: (10) bridesmaid(_G3900) ? creep
   Exit: (10) bridesmaid(jenny) ? creep
   Call: (10) female(jenny) ? creep
   Exit: (10) female(jenny) ? creep
   Exit: (9) wore(jenny, redgown) ? creep
   Call: (9) female(jenny) ? creep
   Exit: (9) female(jenny) ? creep
   Exit: (8) stays(jenny, 210) ? creep
   Exit: (7) longbrownhair(jenny) ? creep
   Call: (7) wore(jenny, blackshoes) ? creep
   Call: (8) wore(jenny, silverbracelet) ? creep
   Exit: (8) wore(jenny, silverbracelet) ? creep
   Call: (8) female(jenny) ? creep
   Exit: (8) female(jenny) ? creep
   Exit: (7) wore(jenny, blackshoes) ? creep
   Exit: (6) thief(jenny) ? creep
```

%% Second Program
%-% --------------------- Starts below this line ---------------------

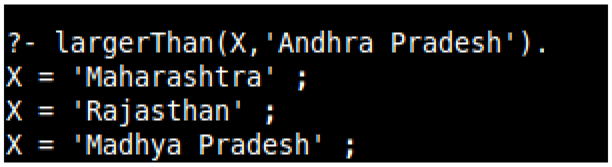
%% Following are the facts given in the problem
%% largerInSize is a functor denoting the first arguement is greater than second
largerInSize('Rajasthan','Madhya Pradesh').
largerInSize('Madhya Pradesh','Maharashtra').
largerInSize('Maharashtra','Andhra Pradesh').
largerInSize('Andhra Pradesh','Uttar Pradesh').

%% Base case of recursion - if it is one of the facts
%% Rule 1
largerThan(X,Y):- largerInSize(X,Y).

%% Rule 2
%% Recursion - Find a city Z which is smaller than X (X is larger then Z) and try to find cities which are smaller than Z
largerThan(X,Y):-largerInSize(X,Z),largerThan(Z,Y).

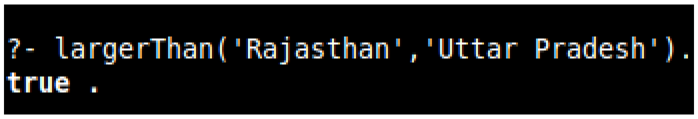
%-% ---------------------Ends Here ---------------------

(a) List all the states that are larger than Andhra Pradesh. [Hint: see the usage of ';' in Prolog]
Ans :



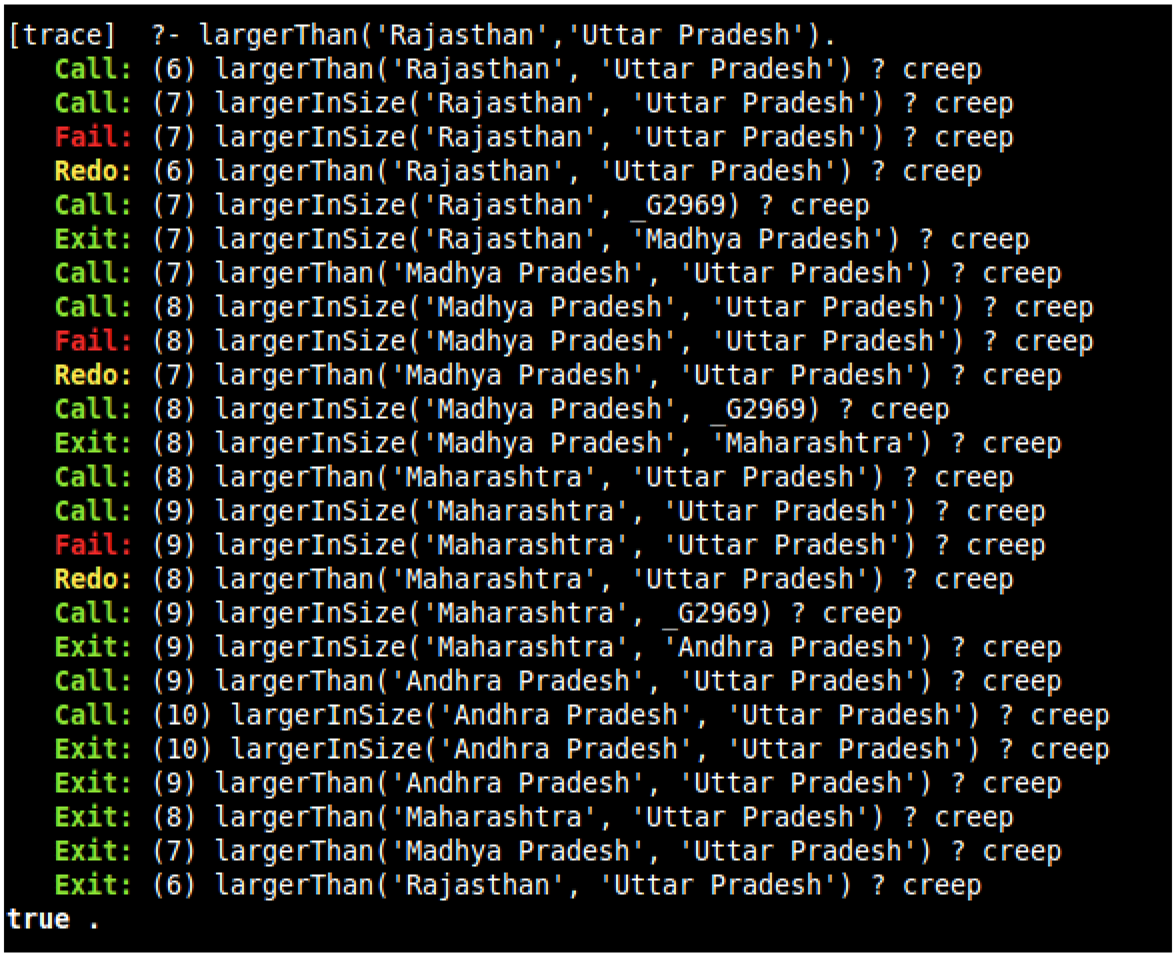
```
?- largerThan(X,'Andhra Pradesh').
X = 'Maharashtra' ;
X = 'Rajasthan' ;
X = 'Madhya Pradesh' ;
```

(b) Is Rajasthan larger than Uttar Pradesh?
Ans:



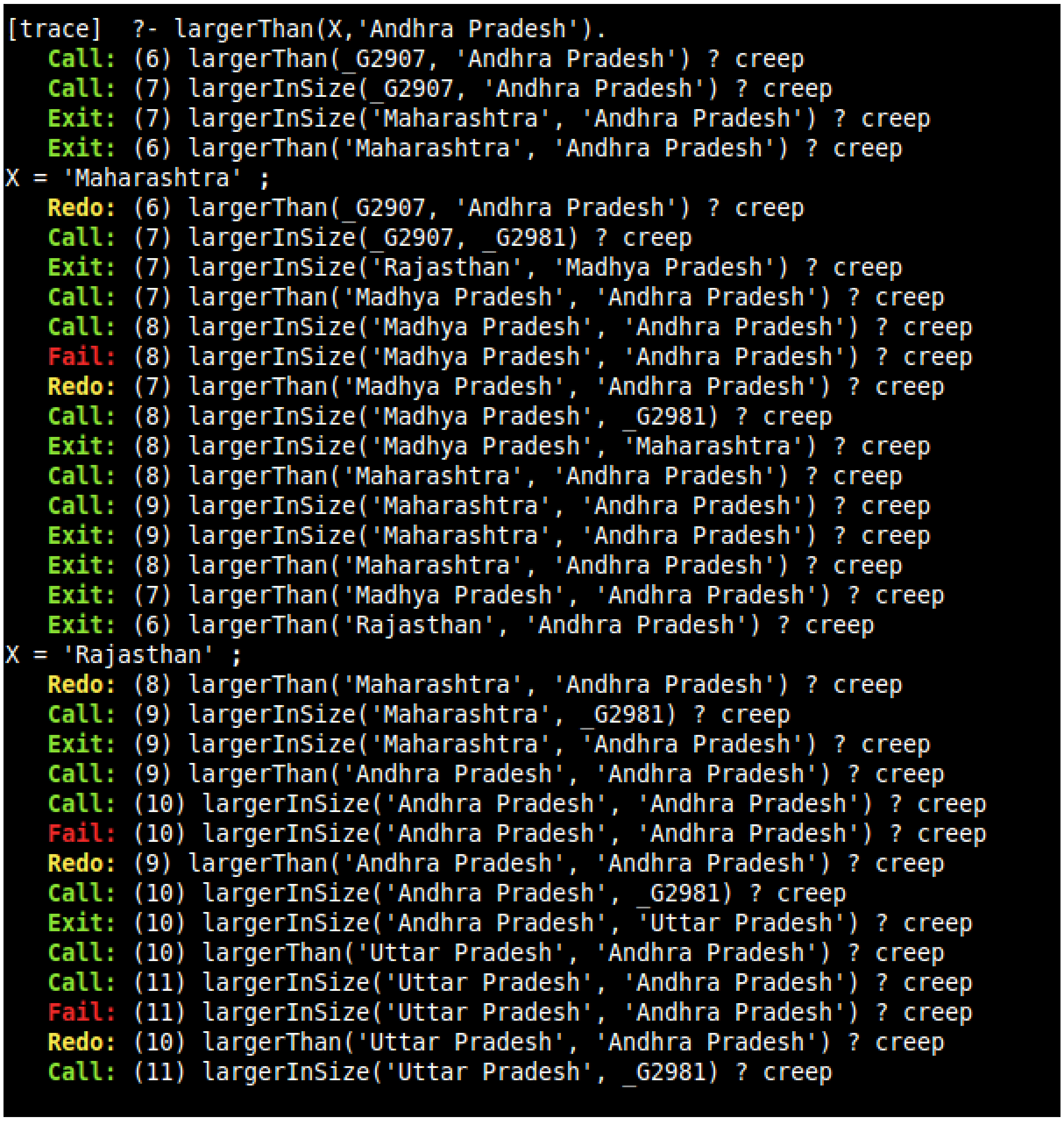
```
?- largerThan('Rajasthan','Uttar Pradesh').
true .
```

Trace of 2nd Program.
For Part A.

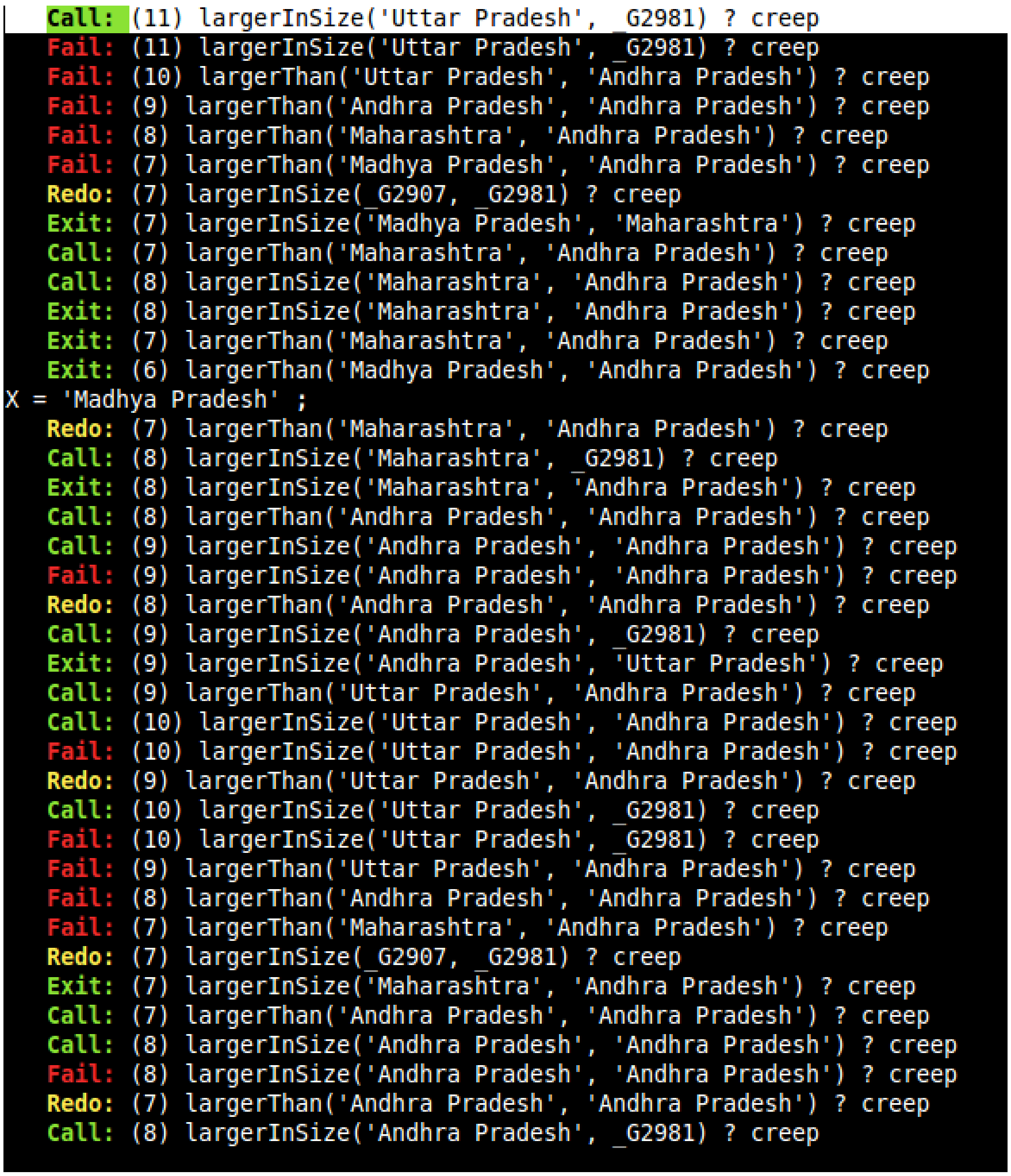


```
[trace]  ?- largerThan('Rajasthan','Uttar Pradesh').
   Call: (6) largerThan('Rajasthan', 'Uttar Pradesh') ? creep
   Call: (7) largerInSize('Rajasthan', 'Uttar Pradesh') ? creep
   Fail: (7) largerInSize('Rajasthan', 'Uttar Pradesh') ? creep
   Redo: (6) largerThan('Rajasthan', 'Uttar Pradesh') ? creep
   Call: (7) largerInSize('Rajasthan', _G2969) ? creep
   Exit: (7) largerInSize('Rajasthan', 'Madhya Pradesh') ? creep
   Call: (7) largerThan('Madhya Pradesh', 'Uttar Pradesh') ? creep
   Call: (8) largerInSize('Madhya Pradesh', 'Uttar Pradesh') ? creep
   Fail: (8) largerInSize('Madhya Pradesh', 'Uttar Pradesh') ? creep
   Redo: (7) largerThan('Madhya Pradesh', 'Uttar Pradesh') ? creep
   Call: (8) largerInSize('Madhya Pradesh', _G2969) ? creep
   Exit: (8) largerInSize('Madhya Pradesh', 'Maharashtra') ? creep
   Call: (8) largerThan('Maharashtra', 'Uttar Pradesh') ? creep
   Call: (9) largerInSize('Maharashtra', 'Uttar Pradesh') ? creep
   Fail: (9) largerInSize('Maharashtra', 'Uttar Pradesh') ? creep
   Redo: (8) largerThan('Maharashtra', 'Uttar Pradesh') ? creep
   Call: (9) largerInSize('Maharashtra', _G2969) ? creep
   Exit: (9) largerInSize('Maharashtra', 'Andhra Pradesh') ? creep
   Call: (9) largerThan('Andhra Pradesh', 'Uttar Pradesh') ? creep
   Call: (10) largerInSize('Andhra Pradesh', 'Uttar Pradesh') ? creep
   Exit: (10) largerInSize('Andhra Pradesh', 'Uttar Pradesh') ? creep
   Exit: (9) largerThan('Andhra Pradesh', 'Uttar Pradesh') ? creep
   Exit: (8) largerThan('Maharashtra', 'Uttar Pradesh') ? creep
   Exit: (7) largerThan('Madhya Pradesh', 'Uttar Pradesh') ? creep
   Exit: (6) largerThan('Rajasthan', 'Uttar Pradesh') ? creep
true .
```

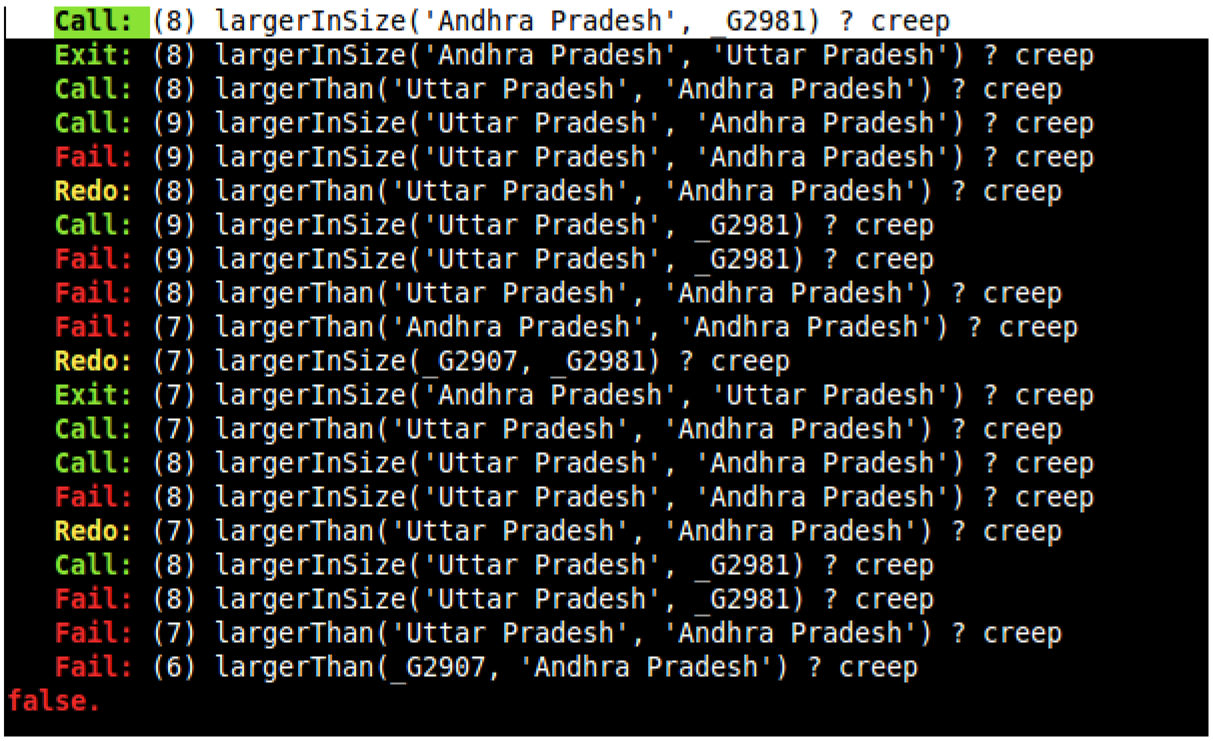
Part B



```
[trace]  ?- largerThan(X,'Andhra Pradesh').
   Call: (6) largerThan(_G2907, 'Andhra Pradesh') ? creep
   Call: (7) largerInSize(_G2907, 'Andhra Pradesh') ? creep
   Exit: (7) largerInSize('Maharashtra', 'Andhra Pradesh') ? creep
   Exit: (6) largerThan('Maharashtra', 'Andhra Pradesh') ? creep
X = 'Maharashtra' ;
   Redo: (6) largerThan(_G2907, 'Andhra Pradesh') ? creep
   Call: (7) largerInSize(_G2907, _G2981) ? creep
   Exit: (7) largerInSize('Rajasthan', 'Madhya Pradesh') ? creep
   Call: (7) largerThan('Madhya Pradesh', 'Andhra Pradesh') ? creep
   Call: (8) largerInSize('Madhya Pradesh', 'Andhra Pradesh') ? creep
   Fail: (8) largerInSize('Madhya Pradesh', 'Andhra Pradesh') ? creep
   Redo: (7) largerThan('Madhya Pradesh', 'Andhra Pradesh') ? creep
   Call: (8) largerInSize('Madhya Pradesh', _G2981) ? creep
   Exit: (8) largerInSize('Madhya Pradesh', 'Maharashtra') ? creep
   Call: (8) largerThan('Maharashtra', 'Andhra Pradesh') ? creep
   Call: (9) largerInSize('Maharashtra', 'Andhra Pradesh') ? creep
   Exit: (9) largerInSize('Maharashtra', 'Andhra Pradesh') ? creep
   Exit: (8) largerThan('Maharashtra', 'Andhra Pradesh') ? creep
   Exit: (7) largerThan('Madhya Pradesh', 'Andhra Pradesh') ? creep
   Exit: (6) largerThan('Rajasthan', 'Andhra Pradesh') ? creep
X = 'Rajasthan' ;
   Redo: (8) largerThan('Maharashtra', 'Andhra Pradesh') ? creep
   Call: (9) largerInSize('Maharashtra', _G2981) ? creep
   Exit: (9) largerInSize('Maharashtra', 'Andhra Pradesh') ? creep
   Call: (9) largerThan('Andhra Pradesh', 'Andhra Pradesh') ? creep
   Call: (10) largerInSize('Andhra Pradesh', 'Andhra Pradesh') ? creep
   Fail: (10) largerInSize('Andhra Pradesh', 'Andhra Pradesh') ? creep
   Redo: (9) largerThan('Andhra Pradesh', 'Andhra Pradesh') ? creep
   Call: (10) largerInSize('Andhra Pradesh', _G2981) ? creep
   Exit: (10) largerInSize('Andhra Pradesh', 'Uttar Pradesh') ? creep
   Call: (10) largerThan('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Call: (11) largerInSize('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Fail: (11) largerInSize('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Redo: (10) largerThan('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Call: (11) largerInSize('Uttar Pradesh', _G2981) ? creep
```



```
   Call: (11) largerInSize('Uttar Pradesh', _G2981) ? creep
   Fail: (11) largerInSize('Uttar Pradesh', _G2981) ? creep
   Fail: (10) largerThan('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Fail: (9) largerThan('Andhra Pradesh', 'Andhra Pradesh') ? creep
   Fail: (8) largerThan('Maharashtra', 'Andhra Pradesh') ? creep
   Fail: (7) largerThan('Madhya Pradesh', 'Andhra Pradesh') ? creep
   Redo: (7) largerInSize(_G2907, _G2981) ? creep
   Exit: (7) largerInSize('Madhya Pradesh', 'Maharashtra') ? creep
   Call: (7) largerThan('Maharashtra', 'Andhra Pradesh') ? creep
   Call: (8) largerInSize('Maharashtra', 'Andhra Pradesh') ? creep
   Exit: (8) largerInSize('Maharashtra', 'Andhra Pradesh') ? creep
   Exit: (7) largerThan('Maharashtra', 'Andhra Pradesh') ? creep
   Exit: (6) largerThan('Madhya Pradesh', 'Andhra Pradesh') ? creep
X = 'Madhya Pradesh' ;
   Redo: (7) largerThan('Maharashtra', 'Andhra Pradesh') ? creep
   Call: (8) largerInSize('Maharashtra', _G2981) ? creep
   Exit: (8) largerInSize('Maharashtra', 'Andhra Pradesh') ? creep
   Call: (8) largerThan('Andhra Pradesh', 'Andhra Pradesh') ? creep
   Call: (9) largerInSize('Andhra Pradesh', 'Andhra Pradesh') ? creep
   Fail: (9) largerInSize('Andhra Pradesh', 'Andhra Pradesh') ? creep
   Redo: (8) largerThan('Andhra Pradesh', 'Andhra Pradesh') ? creep
   Call: (9) largerInSize('Andhra Pradesh', _G2981) ? creep
   Exit: (9) largerInSize('Andhra Pradesh', 'Uttar Pradesh') ? creep
   Call: (9) largerThan('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Call: (10) largerInSize('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Fail: (10) largerInSize('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Redo: (9) largerThan('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Call: (10) largerInSize('Uttar Pradesh', _G2981) ? creep
   Fail: (10) largerInSize('Uttar Pradesh', _G2981) ? creep
   Fail: (9) largerThan('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Fail: (8) largerThan('Andhra Pradesh', 'Andhra Pradesh') ? creep
   Fail: (7) largerThan('Maharashtra', 'Andhra Pradesh') ? creep
   Redo: (7) largerInSize(_G2907, _G2981) ? creep
   Exit: (7) largerInSize('Maharashtra', 'Andhra Pradesh') ? creep
   Call: (7) largerThan('Andhra Pradesh', 'Andhra Pradesh') ? creep
   Call: (8) largerInSize('Andhra Pradesh', 'Andhra Pradesh') ? creep
   Fail: (8) largerInSize('Andhra Pradesh', 'Andhra Pradesh') ? creep
   Redo: (7) largerThan('Andhra Pradesh', 'Andhra Pradesh') ? creep
   Call: (8) largerInSize('Andhra Pradesh', _G2981) ? creep
```

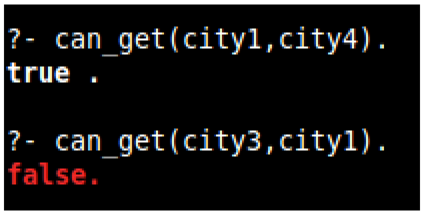


```
   Call: (8) largerInSize('Andhra Pradesh', _G2981) ? creep
   Exit: (8) largerInSize('Andhra Pradesh', 'Uttar Pradesh') ? creep
   Call: (8) largerThan('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Call: (9) largerInSize('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Fail: (9) largerInSize('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Redo: (8) largerThan('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Call: (9) largerInSize('Uttar Pradesh', _G2981) ? creep
   Fail: (9) largerInSize('Uttar Pradesh', _G2981) ? creep
   Fail: (8) largerThan('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Fail: (7) largerThan('Andhra Pradesh', 'Andhra Pradesh') ? creep
   Redo: (7) largerInSize(_G2907, _G2981) ? creep
   Exit: (7) largerInSize('Andhra Pradesh', 'Uttar Pradesh') ? creep
   Call: (7) largerThan('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Call: (8) largerInSize('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Fail: (8) largerInSize('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Redo: (7) largerThan('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Call: (8) largerInSize('Uttar Pradesh', _G2981) ? creep
   Fail: (8) largerInSize('Uttar Pradesh', _G2981) ? creep
   Fail: (7) largerThan('Uttar Pradesh', 'Andhra Pradesh') ? creep
   Fail: (6) largerThan(_G2907, 'Andhra Pradesh') ? creep
false.
```

%% Third Program
%-% ---------------------- Starts below this line ---------------------

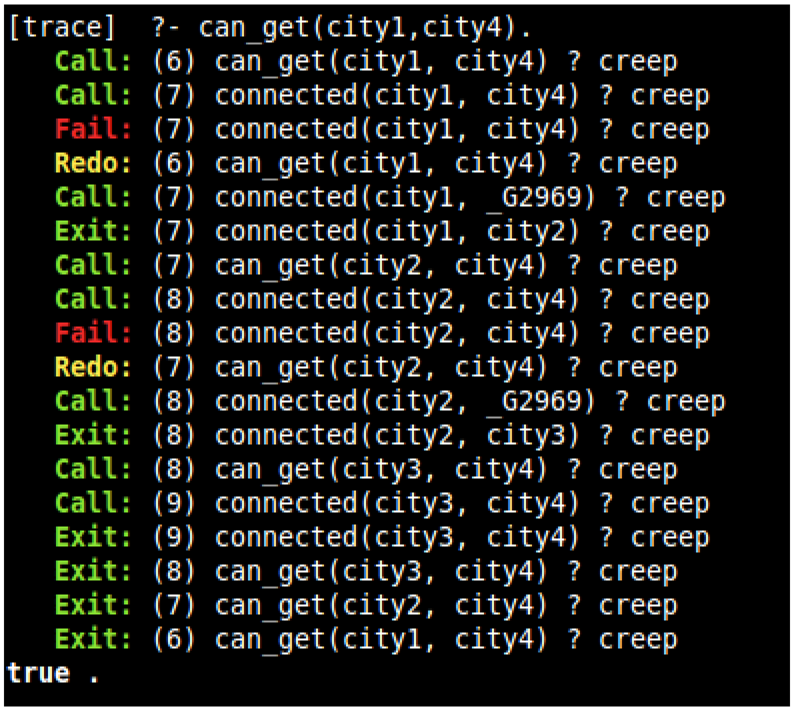
```
%  city1 -> city2 -> city3 -> city4 -> city5 -> city6
% connected is a functor and following are the facts how the cities are connected

connected(city1,city2).
connected(city2,city3).
connected(city3,city4).
connected(city4,city5).
connected(city5,city6).

% One can always reach the city where he/she is in :)
%Rule 1
can_get(X,X):- true.

%Base case - if X and Y are connected
%Rule 2
can_get(X,Y):- connected(X,Y).

%Recursion - Goto the next possible city and see whether he can go to the destination from the next city
%Rule 3
can_get(X,Y):- connected(X,Z),can_get(Z,Y).
```

%-% ---------------------- Ends here ---------------------

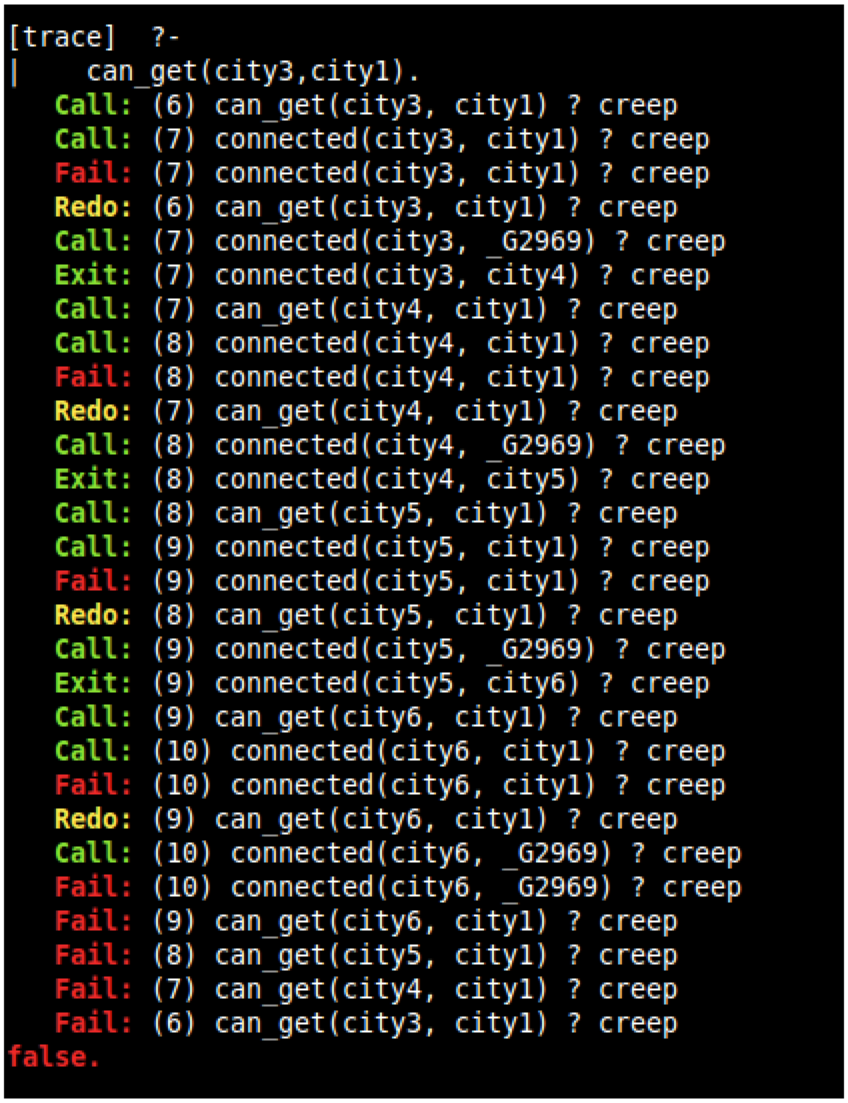


```
?- can_get(city1,city4).
true .

?- can_get(city3,city1).
false.
```

Trace for 3rd Program:



```
[trace]  ?- can_get(city1,city4).
   Call: (6) can_get(city1, city4) ? creep
   Call: (7) connected(city1, city4) ? creep
   Fail: (7) connected(city1, city4) ? creep
   Redo: (6) can_get(city1, city4) ? creep
   Call: (7) connected(city1, _G2969) ? creep
   Exit: (7) connected(city1, city2) ? creep
   Call: (7) can_get(city2, city4) ? creep
   Call: (8) connected(city2, city4) ? creep
   Fail: (8) connected(city2, city4) ? creep
   Redo: (7) can_get(city2, city4) ? creep
   Call: (8) connected(city2, _G2969) ? creep
   Exit: (8) connected(city2, city3) ? creep
   Call: (8) can_get(city3, city4) ? creep
   Call: (9) connected(city3, city4) ? creep
   Exit: (9) connected(city3, city4) ? creep
   Exit: (8) can_get(city3, city4) ? creep
   Exit: (7) can_get(city2, city4) ? creep
   Exit: (6) can_get(city1, city4) ? creep
true .
```



```
[trace]  ?-
|    can_get(city3,city1).
   Call: (6) can_get(city3, city1) ? creep
   Call: (7) connected(city3, city1) ? creep
   Fail: (7) connected(city3, city1) ? creep
   Redo: (6) can_get(city3, city1) ? creep
   Call: (7) connected(city3, _G2969) ? creep
   Exit: (7) connected(city3, city4) ? creep
   Call: (7) can_get(city4, city1) ? creep
   Call: (8) connected(city4, city1) ? creep
   Fail: (8) connected(city4, city1) ? creep
   Redo: (7) can_get(city4, city1) ? creep
   Call: (8) connected(city4, _G2969) ? creep
   Exit: (8) connected(city4, city5) ? creep
   Call: (8) can_get(city5, city1) ? creep
   Call: (9) connected(city5, city1) ? creep
   Fail: (9) connected(city5, city1) ? creep
   Redo: (8) can_get(city5, city1) ? creep
   Call: (9) connected(city5, _G2969) ? creep
   Exit: (9) connected(city5, city6) ? creep
   Call: (9) can_get(city6, city1) ? creep
   Call: (10) connected(city6, city1) ? creep
   Fail: (10) connected(city6, city1) ? creep
   Redo: (9) can_get(city6, city1) ? creep
   Call: (10) connected(city6, _G2969) ? creep
   Fail: (10) connected(city6, _G2969) ? creep
   Fail: (9) can_get(city6, city1) ? creep
   Fail: (8) can_get(city5, city1) ? creep
   Fail: (7) can_get(city4, city1) ? creep
   Fail: (6) can_get(city3, city1) ? creep
false.
```