Experiment No.: 6

Implement Monkey Banana problem using Prolog. Also, demonstrate the use of trace and notrace command of the Prolog.

Output:

```
SWI-Prolog (AMD64, Multi-threaded, version 9.2.6)
File  Edit  Settings  Run  Debug  Help
Welcome to SWI-Prolog (threaded, 64 bits, version 9.2.6)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?-
% c:/Users/nayan/Documents/Btech/3-4 year/7 sem/AI/Lab/Practicals/practical-6/practical-6.pl compiled 0.00 sec, 10 clauses
?-
|    can_reach(X,Y).
X = monkey,
Y = bananas ,

?- is_close(X,Z).
X = monkey,
Z = bananas ,

?- under(Y,Z).
Y = chair,
Z = bananas
```

```
?-
% c:/Users/nayan/Documents/Btech/3-4 year/7 sem/AI/Lab/Practicals/practical-6/practical-6.pl compiled 0.00 sec, 10 clauses
?-
|    trace.
true.

[trace]  ?- can_reach(X,Y).
   Call: (12) can_reach(_22386, _22388) ? creep
   Call: (13) dexterous(_22386) ? creep
   Exit: (13) dexterous(monkey) ? creep
   Call: (13) is_close(monkey, _22388) ? creep
   Call: (14) can_climb(monkey, _26138) ? creep
   Exit: (14) can_climb(monkey, chair) ? creep
   Call: (14) under(chair, _22388) ? creep
   Call: (15) in_room(_28570) ? creep
   Exit: (15) in_room(bananas) ? creep
   Call: (15) in_room(chair) ? creep
   Exit: (15) in_room(chair) ? creep
   Call: (15) in_room(_22388) ? creep
   Exit: (15) in_room(bananas) ? creep
   Call: (15) can_move(bananas, chair, bananas) ? creep
   Fail: (15) can_move(bananas, chair, bananas) ? creep
   Redo: (15) in_room(_22388) ? creep
   Exit: (15) in_room(chair) ? creep
   Call: (15) can_move(bananas, chair, chair) ? creep
   Fail: (15) can_move(bananas, chair, chair) ? creep
   Redo: (15) in_room(_22388) ? creep
   Exit: (15) in_room(monkey) ? creep
   Call: (15) can_move(bananas, chair, monkey) ? creep
   Fail: (15) can_move(bananas, chair, monkey) ? creep
   Redo: (15) in_room(_28570) ? creep
```

```
         Call: (15) in_room(_22388) ? creep
         Exit: (15) in_room(bananas) ? creep
         Call: (15) can_move(chair, chair, bananas) ? creep
         Fail: (15) can_move(chair, chair, bananas) ? creep
         Redo: (15) in_room(_22388) ? creep
         Exit: (15) in_room(chair) ? creep
         Call: (15) can_move(chair, chair, chair) ? creep
         Fail: (15) can_move(chair, chair, chair) ? creep
         Redo: (15) in_room(_22388) ? creep
         Exit: (15) in_room(monkey) ? creep
         Call: (15) can_move(chair, chair, monkey) ? creep
         Fail: (15) can_move(chair, chair, monkey) ? creep
         Redo: (15) in_room(_28570) ? creep
         Exit: (15) in_room(monkey) ? creep
         Call: (15) in_room(chair) ? creep
         Exit: (15) in_room(chair) ? creep
         Call: (15) in_room(_22388) ? creep
         Exit: (15) in_room(bananas) ? creep
         Call: (15) can_move(monkey, chair, bananas) ? creep
         Exit: (15) can_move(monkey, chair, bananas) ? creep
         Exit: (14) under(chair, bananas) ? creep
         Call: (14) tall(chair) ? creep
         Exit: (14) tall(chair) ? creep
         Exit: (13) is_close(monkey, bananas) ? creep
         Exit: (12) can_reach(monkey, bananas) ? creep
      X = monkey,
      Y = bananas ;
         Redo: (15) in_room(_60) ? creep
         Exit: (15) in_room(chair) ? creep
         Call: (15) can_move(monkey, chair, chair) ?
```