**PROJECT REPORT**  
**ON  
 8-PUZZLE PROBLEM USING  
 A\* ALGORITHM**

**GUIDED BY:**  
 **Mr. Ankit Rajpal**

**PREPARED BY:**

Richa Sharma 15HCS4304  
 Aditi Parashar 15HCS4302   
 Jassa Singh 15HCS4353

****

Department of Computer Science

Deen Dayal Upadhyaya College

University of Delhi

Sector-3, Dwarka, New Delhi - 110078.

Acknowledgement

There are many people who have helped us directly or indirectly in the successful completion of our project. We would like to take this opportunity to thank one and all.

We are very thankful to our project guide **Mr Ankit Rajpal** who has been inspiring guide and committed caretaker for his unflinching devotion. The encouragement and support by him, especially in carrying out this project motivated us to complete this project.

 We would like to thank all our friends for their help and constructive criticism during our project period. Finally, we are very much indebted to our parents for their moral support and encouragement to achieve goals.

Aditi Parashar  
Richa Sharma  
Jassa Singh

CODE

left( [A,0,C,D,E,F,H,I,J] , [0,A,C,D,E,F,H,I,J] ).

left( [A,B,C,D,0,F,H,I,J] , [A,B,C,0,D,F,H,I,J] ).

left( [A,B,C,D,E,F,H,0,J] , [A,B,C,D,E,F,0,H,J] ).

left( [A,B,0,D,E,F,H,I,J] , [A,0,B,D,E,F,H,I,J] ).

left( [A,B,C,D,E,0,H,I,J] , [A,B,C,D,0,E,H,I,J] ).

left( [A,B,C,D,E,F,H,I,0] , [A,B,C,D,E,F,H,0,I] ).

up( [A,B,C,0,E,F,H,I,J] , [0,B,C,A,E,F,H,I,J] ).

up( [A,B,C,D,0,F,H,I,J] , [A,0,C,D,B,F,H,I,J] ).

up( [A,B,C,D,E,0,H,I,J] , [A,B,0,D,E,C,H,I,J] ).

up( [A,B,C,D,E,F,0,I,J] , [A,B,C,0,E,F,D,I,J] ).

up( [A,B,C,D,E,F,H,0,J] , [A,B,C,D,0,F,H,E,J] ).

up( [A,B,C,D,E,F,H,I,0] , [A,B,C,D,E,0,H,I,F] ).

right( [A,0,C,D,E,F,H,I,J] , [A,C,0,D,E,F,H,I,J] ).

right( [A,B,C,D,0,F,H,I,J] , [A,B,C,D,F,0,H,I,J] ).

right( [A,B,C,D,E,F,H,0,J] , [A,B,C,D,E,F,H,J,0] ).

right( [0,B,C,D,E,F,H,I,J] , [B,0,C,D,E,F,H,I,J] ).

right( [A,B,C,0,E,F,H,I,J] , [A,B,C,E,0,F,H,I,J] ).

right( [A,B,C,D,E,F,0,I,J] , [A,B,C,D,E,F,I,0,J] ).

down( [A,B,C,0,E,F,H,I,J] , [A,B,C,H,E,F,0,I,J] ).

down( [A,B,C,D,0,F,H,I,J] , [A,B,C,D,I,F,H,0,J] ).

down( [A,B,C,D,E,0,H,I,J] , [A,B,C,D,E,J,H,I,0] ).

down( [0,B,C,D,E,F,H,I,J] , [D,B,C,0,E,F,H,I,J] ).

down( [A,0,C,D,E,F,H,I,J] , [A,E,C,D,0,F,H,I,J] ).

down( [A,B,0,D,E,F,H,I,J] , [A,B,F,D,E,0,H,I,J] ).

**cmp\_list**([], [], R):- R is 0.

**cmp\_list**([H1|T1], [H1|T2],R) :- cmp\_list(T1, T2, R).

**cmp\_list**([\_|T1], [\_|T2],R) :- cmp\_list(T1, T2, R1), R is R1+1.

**lSuccessor**(I, G, Node, V) **:- left**(I, Node), **cmp\_list**(Node, G, V), !.

**lSuccessor**(\_, \_, [], 99).

**uSuccessor**(I, G, Node, V) **:- up**(I, Node), **cmp\_list**(Node, G, V), !.

**uSuccessor**(\_, \_, [], 99).

**rSuccessor**(I, G, Node, V) **:- right**(I, Node), **cmp\_list**(Node, G, V), !.

**rSuccessor**(\_, \_, [], 99).

**dSuccessor**(I, G, Node, V) **:- down**(I, Node), **cmp\_list**(Node, G, V), !.

**dSuccessor**(\_, \_, [], 99).

**addToOpen**([[\_,\_,\_,Node]|\_], Open, Close, Open)**:- member**([\_,\_,\_,Node], Close), !.

**addToOpen**(Node, Open, \_, [Node|Open]).

**getSuccessor**(I, Depth, G, Open, Close, NOpen) **:-**

**NDepth is Depth+1,**

**lSuccessor**(I, G, NodeL, L), F1 is L + NDepth, addToOpen([F1, L, NDepth, NodeL], Open, Close, T1),

**uSuccessor**(I, G, NodeU, U), F2 is U + NDepth, addToOpen([F2, U, NDepth, NodeU], T1, Close, T2),

**rSuccessor**(I, G, NodeR, R), F3 is R + NDepth, addToOpen([F3, R, NDepth, NodeR], T2, Close, T3),

**dSuccessor**(I, G, NodeD, D), F4 is D + NDepth, addToOpen([F4, D, NDepth, NodeD], T3, Close, NOpen).

**explore**(Current, \_, \_, \_, Current, \_, \_):- write('Done'), !.

**explore**(\_, \_, \_, \_, \_, [], \_):- write('No Solution'), !, fail.

**explore**(Current, CurrentF, CurrentH, CurrentD, Goal, Open, Close) **:-**

**delete**(Open, [CurrentF, CurrentH, CurrentD, Current], NOpen),

**append**(Close, [[CurrentF, CurrentH, CurrentD, Current]], NClose),

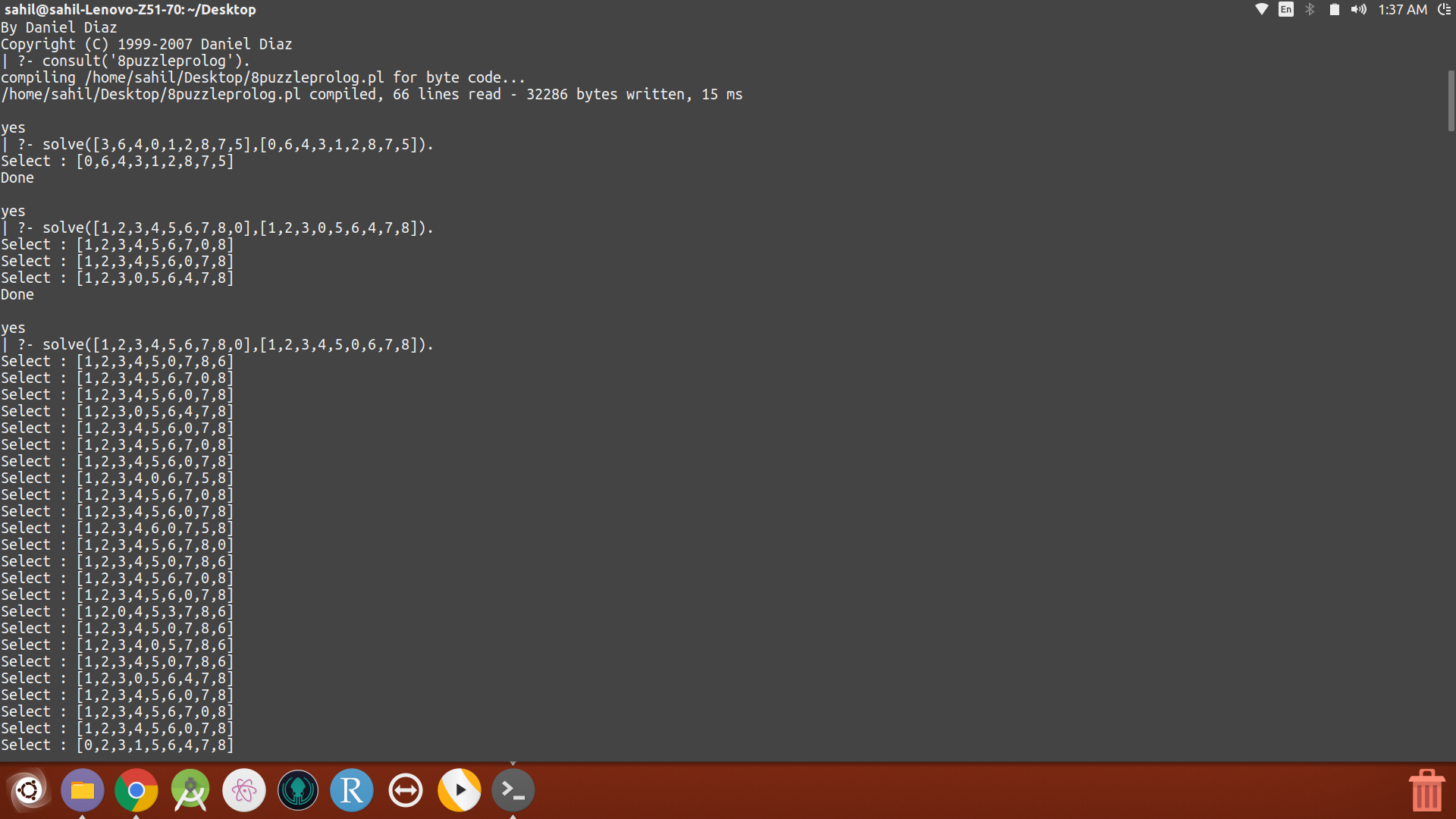
**getSuccessor**(Current, CurrentD, Goal, NOpen, NClose, UOpen),

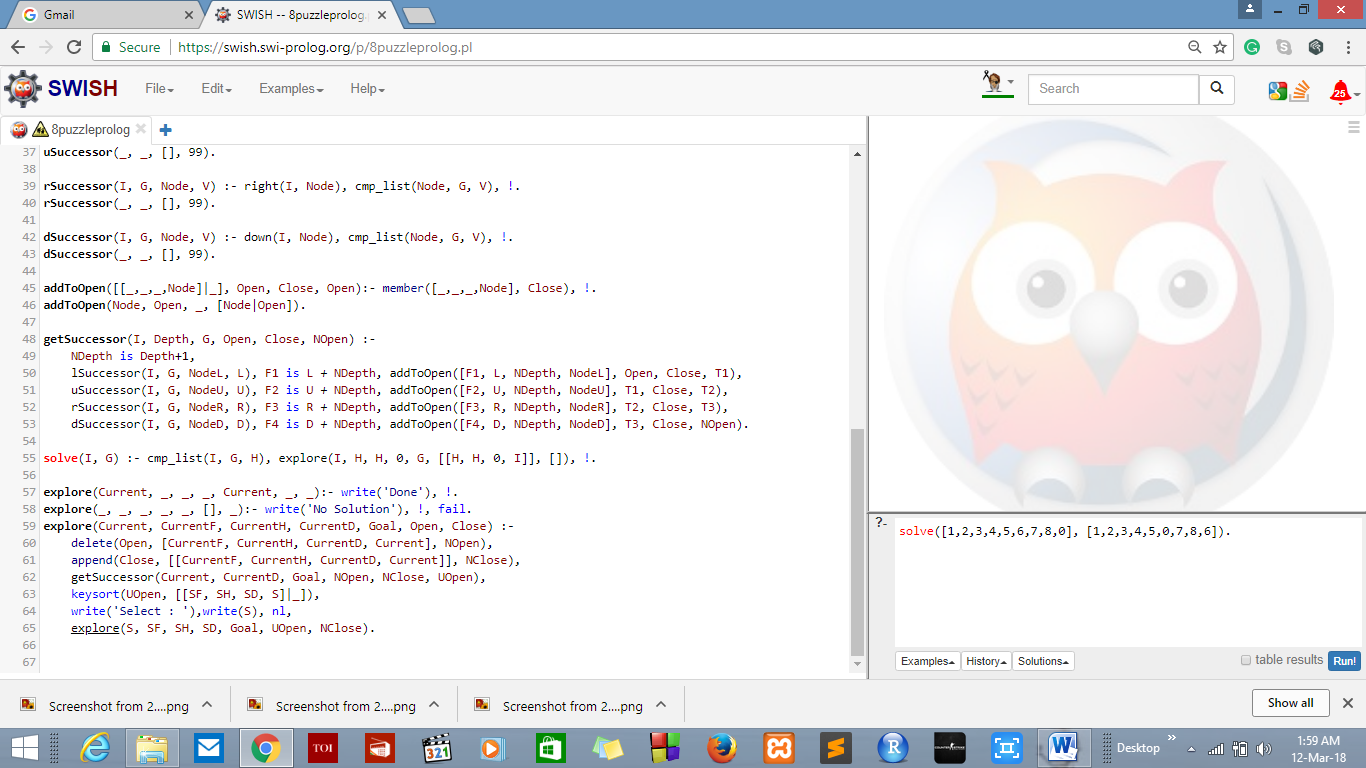
**keysort**(UOpen, [[SF, SH, SD, S]|\_]),

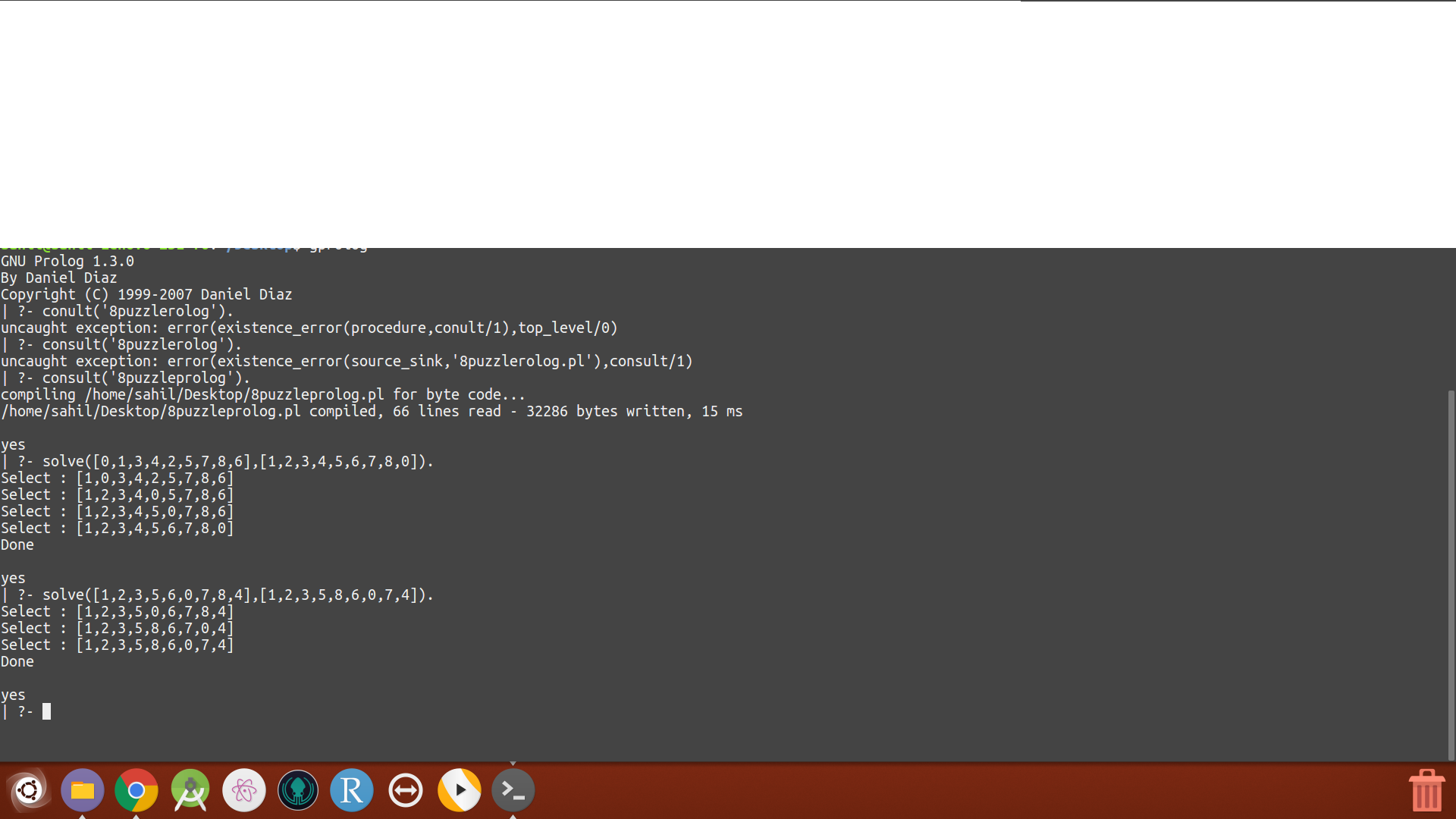
**write**('Select : '),write(S), nl,

**explore**(S, SF, SH, SD, Goal, UOpen, NClose).

**solve**(I, G) **:- cmp\_list**(I, G, H), **explore**(I, H, H, 0, G, [[H, H, 0, I]], []), !.







BIBLIOGRAPHY

1. [https://en.wikipedia.org/wiki/A\*\_search\_algorithm](https://en.wikipedia.org/wiki/A*_search_algorithm)
2. <https://www.geeksforgeeks.org/a-search-algorithm/>
3. <http://artificialintelligence-notes.blogspot.in/2010/07/8-puzzle-problem.html>
4. <https://www.cpp.edu/~jrfisher/www/prolog_tutorial/5_2.html>