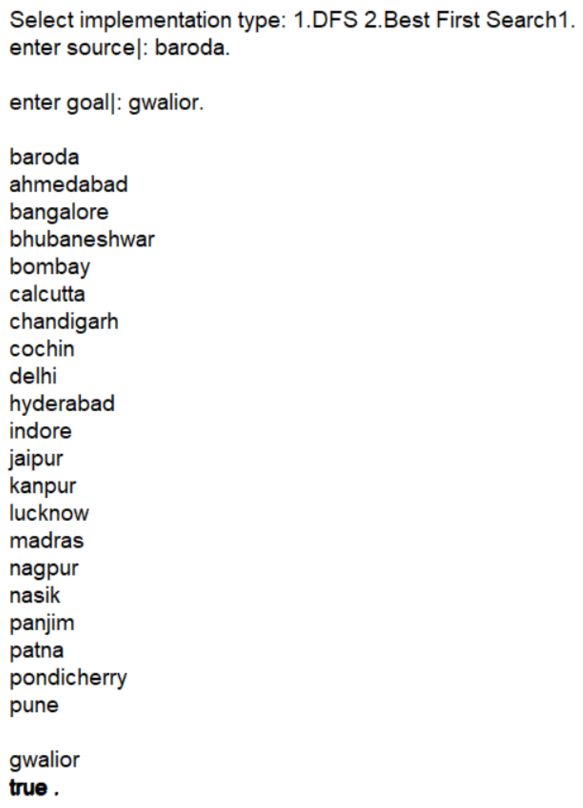
Assignment-2

Name: Shradha Agarawal

Roll No. MT20123

I have implemented Depth first search and Best first search. Several Prolog features are used- lists, csv read, input/output, backtracking, recursion, sorting, findall, forall, etc. Heuristic used- Aerial distance between given city to goal city.

Sample output- 1 (Depth first search)



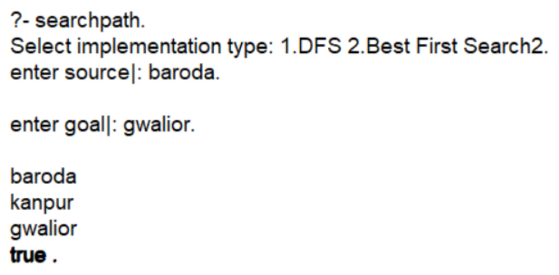
```
Select implementation type: 1.DFS 2.Best First Search1.
enter source|: baroda.

enter goal|: gwalior.

baroda
ahmedabad
bangalore
bhubaneshwar
bombay
calcutta
chandigarh
cochin
delhi
hyderabad
indore
jaipur
kanpur
lucknow
madras
nagpur
nasik
panjim
patna
pondicherry
pune

gwalior
true .
```

Sample output – 2 (Best first search)



```
?- searchpath.
Select implementation type: 1.DFS 2.Best First Search2.
enter source|: baroda.

enter goal|: gwalior.

baroda
kanpur
gwalior
true .
```

Code:

```
:- [library(csv)].

:- [library(heaps)].

:- dynamic samples/3.
:- dynamic column_keys/1.

:- dynamic heuristic/3.

:- dynamic column_keys1/1.


searchpath :-

            retractall(goal(_)),

    retractall(column_keys(_)),

    retractall(column_keys1(_)),

    retractall(heuristic(_,_,_)),

    retractall(samples(_,_,_)),

    method(Method).


method(Method):-

    write('Select implementation type: 1.DFS 2.Best First Search'), read(Method), read_csvfile(File),
implement(Method).


implement(Method):-

    Method = 1, !,takeinput(Source), dfs(Source, Path), printpath(Path).

implement(Method):-

    Method = 2, read_csvfile1(File), !, takeinput(Source), bestfs(Source, Path), printpath(Path).


% depth first search

read_csvfile(File) :-

  ( nonvar(File) ; File = 'C:/Users/Admin/Desktop/Sem1/AI/Assignment2_graded/roaddistance.csv'),

  forall(read_row(File, Row), store_row(Row)).


read_row(File, Row) :-

  csv_read_file_row(File, Row,[]).


store_row(Row) :-

  column_keys(Colkeys), Row =.. [_|Cols], Cols = [RowKey|Samples], findall(X, column_keys(X),Z),
store_sample(RowKey, Z, Samples).
```

```
store_row(Row):-
   Row =.. [_|Cols], without_last(Cols,Col), create_fact(Col).


create_fact([]).
create_fact([H|T]):- assert(column_keys(H)),create_fact(T).


store_sample(_,[],[]).
store_sample(RowKey, [ColKey|Tail], [Sample|T]) :- assert(samples(RowKey, ColKey,
Sample)),store_sample(RowKey,Tail, T).


without_last([_], []).
without_last([X|Xs], [X|WithoutLast]) :-
   without_last(Xs, WithoutLast).


takeinput(Source):-
      write('enter source'), read(Source), nl, write('enter goal'), read(Goal), nl, assert(goal(Goal)).
dfs(Source, Path):-
      dfs1(Source,[Source],Path1), reverse(Path1,Path).
dfs1(S,Path,Path) :- goal(S).
dfs1(S,Closed,Path) :- (samples(S,S2,_); samples(S2,S,_)),not(member(S2,Closed)),
dfs1(S2,[S2|Closed], Path).


printpath([]).
printpath([H|T]):- writeln(H), printpath(T).


% best first search


read_csvfile1(File) :-
   ( nonvar(File) ; File = 'C:/Users/Admin/Desktop/Sem1/AI/Assignment2_graded/aerialdist.csv'),
forall(read_row(File, Row), store_row1(Row)).


store_row1(Row) :-
```

```
    column_keys1(Colkeys), Row =.. [_|Cols], Cols = [RowKey|Samples], findall(X, column_keys1(X),Z),
store_sample1(RowKey, Z, Samples).

store_row1(Row):-

    Row =.. [_|Cols], without_last(Cols,Col), create_fact1(Col).


create_fact1([]).

create_fact1([H|T]):-

    assert(column_keys1(H)),create_fact1(T).


store_sample1(_,[],[]).

store_sample1(RowKey, [ColKey|Tail], [Sample|T]) :-

    assert(heuristic(RowKey, ColKey, Sample)),store_sample1(RowKey,Tail, T).


bestfs(Source, Path):-

     bestfs1(Source,[[9999,dummy]],[Source],Path1), reverse(Path1,Path).

bestfs1(S,T1,Path,Path) :- goal(S).

bestfs1(S,T1,Closed,Path) :-

     goal(G), findall([Y,X],((samples(S,X,_); samples(X,S,_)),heuristic(X,G,Y),not(member(X, Closed))),Z),
append(Z,T1,Result),

     sort(1,@<,Result,[H|T]), nth1(2,H, H1), bestfs1(H1,T,[H1|Closed],Path).
```