Shri Vile Parle Kelavani Mandal's

INSTITUTE OF TECHNOLOGY

DHULE (M.S.)

DEPARMENT OF COMPUTER ENGINEERING

Subject : Artificial Intelligence Lab

Name : Mohammad Anas Aarif Baig Mirza **Roll No. :** 40

Class : B.tech Final Year **Batch : B2** **Division: -**

Expt. No. :04 **Date : 04/09/2025**

Title : Solve any problem using best first search.

Remark

Signature

Code:

```
% ------------------------------------
% Undirected graph (subset of Romania map)
% ------------------------------------
edge(arad, sibiu, 140).
edge(sibiu, fagaras, 99).
edge(sibiu, rimnicu, 80).
edge(fagaras, bucharest, 211).
edge(rimnicu, pitesti, 97).
edge(pitesti, bucharest, 101).

% Make edges undirected
adj(X, Y) :- edge(X, Y, _).
adj(X, Y) :- edge(Y, X, _).

% ------------------------------------
% Heuristic: straight-line distance to Bucharest (example values)
% (Only relative ranking matters for greedy best-first)
% ------------------------------------
heuristic(arad, 366).
heuristic(sibiu, 253).
heuristic(fagaras, 176).
heuristic(rimnicu, 193).
heuristic(pitesti, 100).
heuristic(bucharest, 0).

% ------------------------------------
```

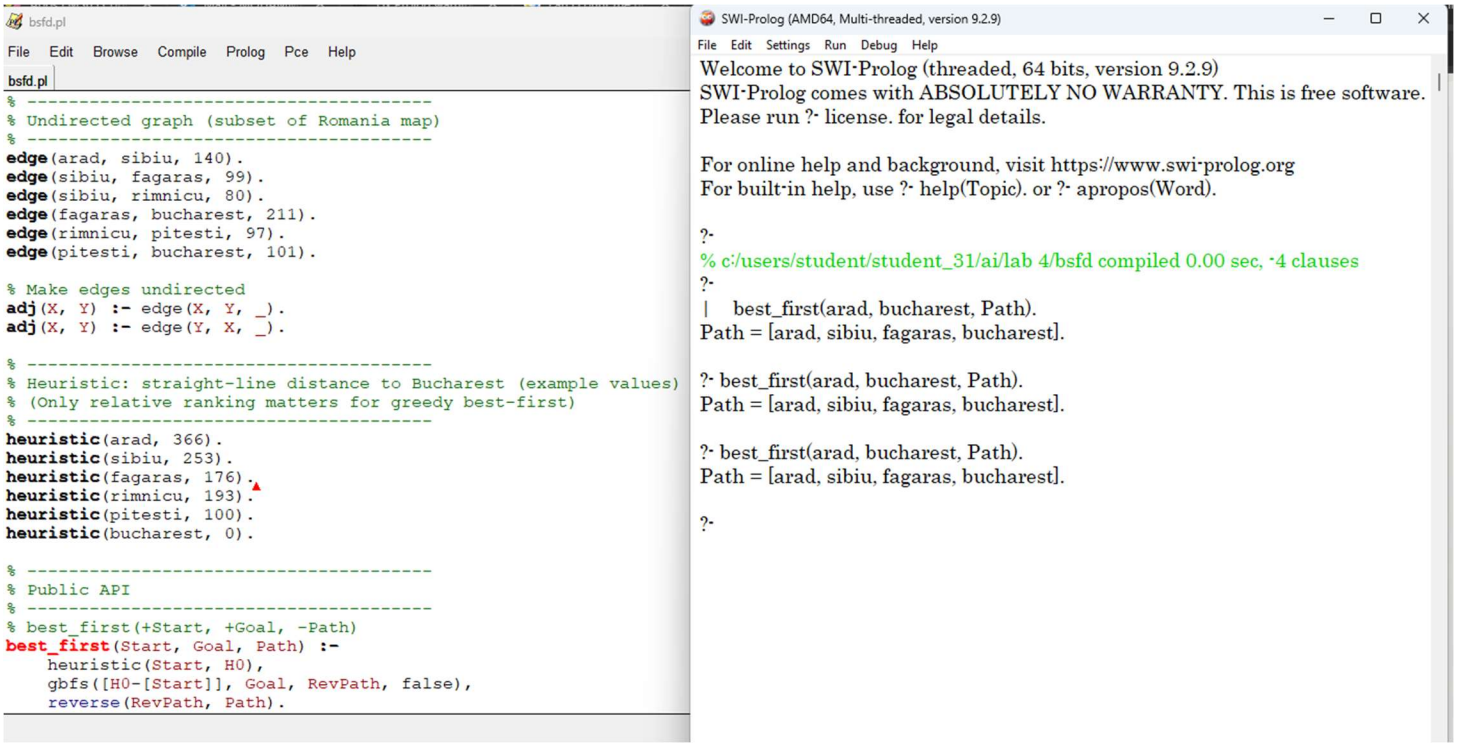
```
% Public API
% ------------------------------------
% best_first(+Start, +Goal, -Path)
best_first(Start, Goal, Path) :-
   heuristic(Start, H0),
   gbfs([H0-[Start]], Goal, RevPath, false),
   reverse(RevPath, Path).

% best_first_debug(+Start, +Goal, -Path)
%   Same as best_first/3, but prints expansions and frontier at each step.
best_first_debug(Start, Goal, Path) :-
   heuristic(Start, H0),
   gbfs([H0-[Start]], Goal, RevPath, true),
   reverse(RevPath, Path).

% ------------------------------------
% Greedy Best-First core
% Open list is a list of H-Path pairs, kept sorted by H (ascending)
% ------------------------------------
gbfs([_H-[Goal|Rest] | _], Goal, [Goal|Rest], _Debug) :- !.
gbfs([H-[Current|RestPath] | Open], Goal, Path, Debug) :-
   ( Debug == true ->
      format('Expanding: ~w~n', [Current]),
      print_frontier([H-[Current|RestPath] | Open])
   ;  true
   ),
   findall(H1-[Next,Current|RestPath],
         ( adj(Current, Next),
           \+ member(Next, [Current|RestPath]),   % avoid cycles
           heuristic(Next, H1)
         ),
         Children),
   append(Open, Children, Open1),
   keysort(Open1, OpenSorted),  % sort by heuristic key
   gbfs(OpenSorted, Goal, Path, Debug).

print_frontier(Frontier) :-
   findall((H,Head),
         ( member(H-[Head|_], Frontier) ),
         Pairs),
   format('Frontier (H,Node): ~w~n~n', [Pairs]).
```

Output:



bsfd.pl

File Edit Browse Compile Prolog Pce Help

bsfd.pl

```
% ----------------------------------------
% Undirected graph (subset of Romania map)
% ----------------------------------------
edge(arad, sibiu, 140).
edge(sibiu, fagaras, 99).
edge(sibiu, rimnicu, 80).
edge(fagaras, bucharest, 211).
edge(rimnicu, pitesti, 97).
edge(pitesti, bucharest, 101).

% Make edges undirected
adj(X, Y) :- edge(X, Y, _).
adj(X, Y) :- edge(Y, X, _).

% ----------------------------------------
% Heuristic: straight-line distance to Bucharest (example values)
% (Only relative ranking matters for greedy best-first)
% ----------------------------------------
heuristic(arad, 366).
heuristic(sibiu, 253).
heuristic(fagaras, 176).
heuristic(rimnicu, 193).
heuristic(pitesti, 100).
heuristic(bucharest, 0).

% ----------------------------------------
% Public API
% ----------------------------------------
% best_first(+Start, +Goal, -Path)
best_first(Start, Goal, Path) :-
    heuristic(Start, H0),
    gbfs([H0-[Start]], Goal, RevPath, false),
    reverse(RevPath, Path).
```

SWI-Prolog (AMD64, Multi-threaded, version 9.2.9)

File Edit Settings Run Debug Help

```
Welcome to SWI-Prolog (threaded, 64 bits, version 9.2.9)
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/student/student_31/ai/lab 4/bsfd compiled 0.00 sec, 4 clauses
?-
|    best_first(arad, bucharest, Path).
Path = [arad, sibiu, fagaras, bucharest].

?- best_first(arad, bucharest, Path).
Path = [arad, sibiu, fagaras, bucharest].

?- best_first(arad, bucharest, Path).
Path = [arad, sibiu, fagaras, bucharest].

?-
```