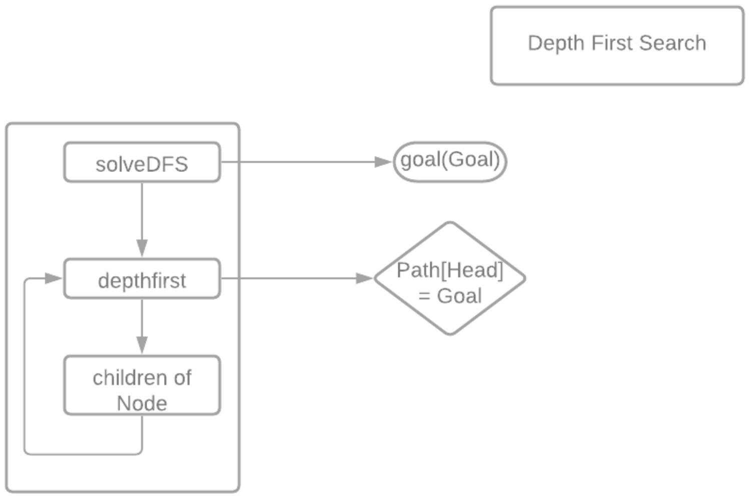
Assignment 2.
Search

By Manvi Goel, 2019472

Code Explanation

Code Structure

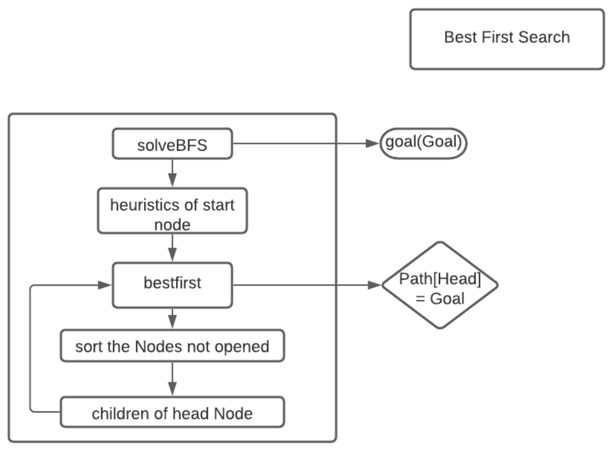
<u>Depth First Search</u>



It first checks if the goal and head of the path are the same.
If they are the same, then the path is returned.
If they are not the same, then all the children of the head are calculated. All the paths are calculated using backtracking and recursion.

The code is commented on for better understanding.

<u>Best First Search</u>



It first checks if the goal and head of the path are the same.
If they are the same the path is returned.
If they are not the same then all the unopened nodes are sorted in ascending order according to the heuristic values. The node with the least heuristic value is opened.

The code is commented on for better understanding.

Working and Assumptions

1. Input Required:
   Starting node: The starting city should belong to the list.
   Goal node: The goal city should belong to the list.
   Search Algorithm: The algorithm (1) or (2) should be selected for depth-first search or best first search respectively.
   Depth First Search. The maximum depth the algorithm should go. Path Length can be maximum depth + 1.

2. Run the program by typing: start.
   To run all the depth-first search paths, run solveDFS(Start, Goal, Variable for path, MaxDepth, Variable for Path Length).
   Use ";" for all the paths.

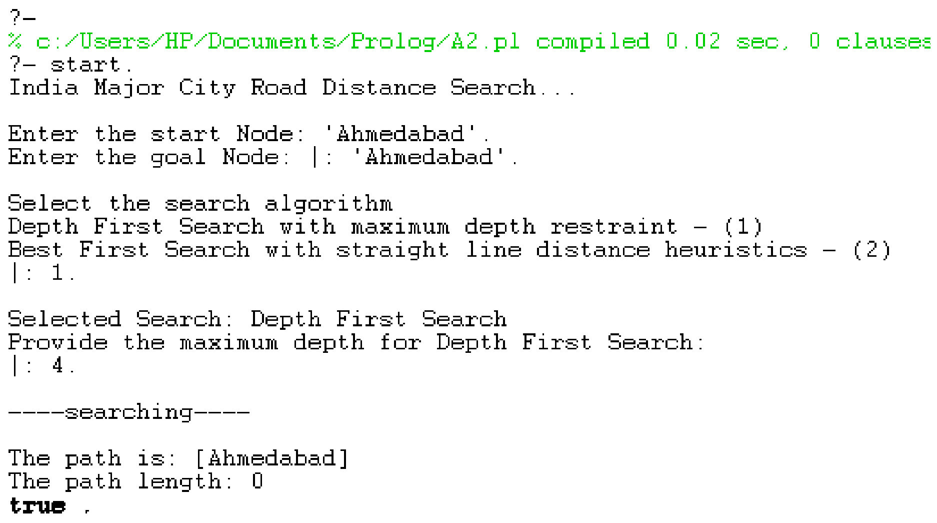
3. Heuristics for Best First Search.
   The heuristics used are the straight line distance of the specified node to the goal node. The heuristics are calculated using a Python program and a CSV file is provided. The python program and CSV file are also attached in the zip file.
   It can be seen that the heuristics function is both admissible and consistent. Since straight line distance is less than road distances, it is not overestimated.

4. Output.
   The path and path length.

Sample Outputs.

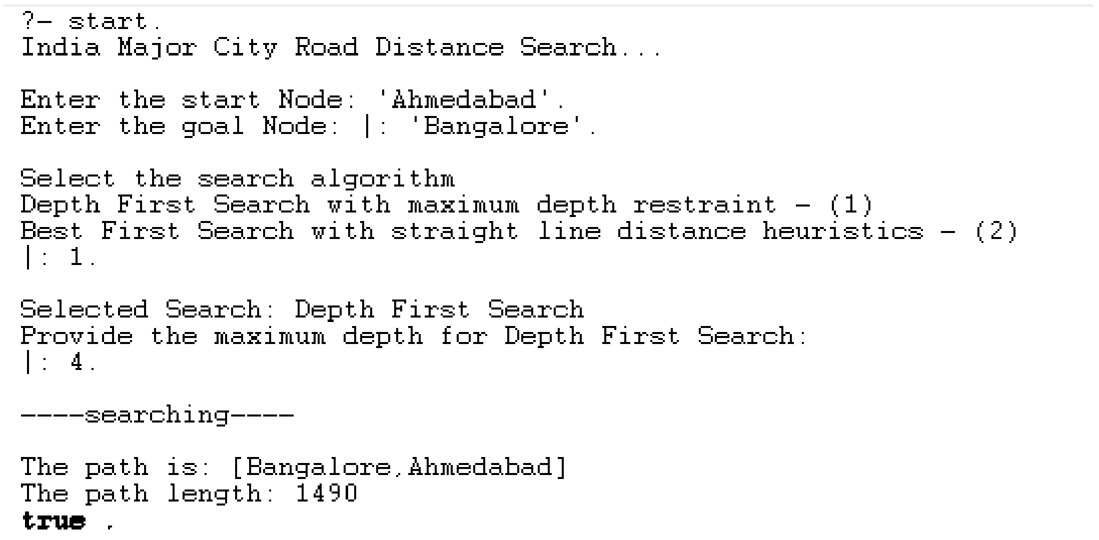
Depth First Search

1. Start: Ahmedabad, Goal: Ahmedabad



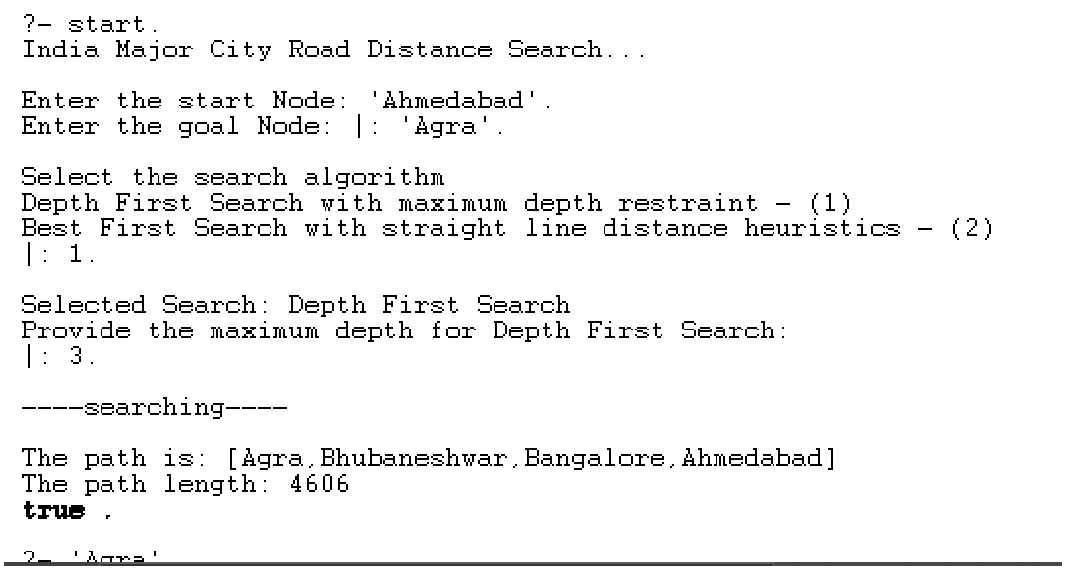
```
?-
% c:/Users/HP/Documents/Prolog/A2.pl compiled 0.02 sec, 0 clauses
?- start.
India Major City Road Distance Search...

Enter the start Node: 'Ahmedabad'.
Enter the goal Node: |: 'Ahmedabad'.

Select the search algorithm
Depth First Search with maximum depth restraint - (1)
Best First Search with straight line distance heuristics - (2)
|: 1.

Selected Search: Depth First Search
Provide the maximum depth for Depth First Search:
|: 4.

----searching----

The path is: [Ahmedabad]
The path length: 0
true .
```

2. Start: Ahmedabad, Goal: Bangalore



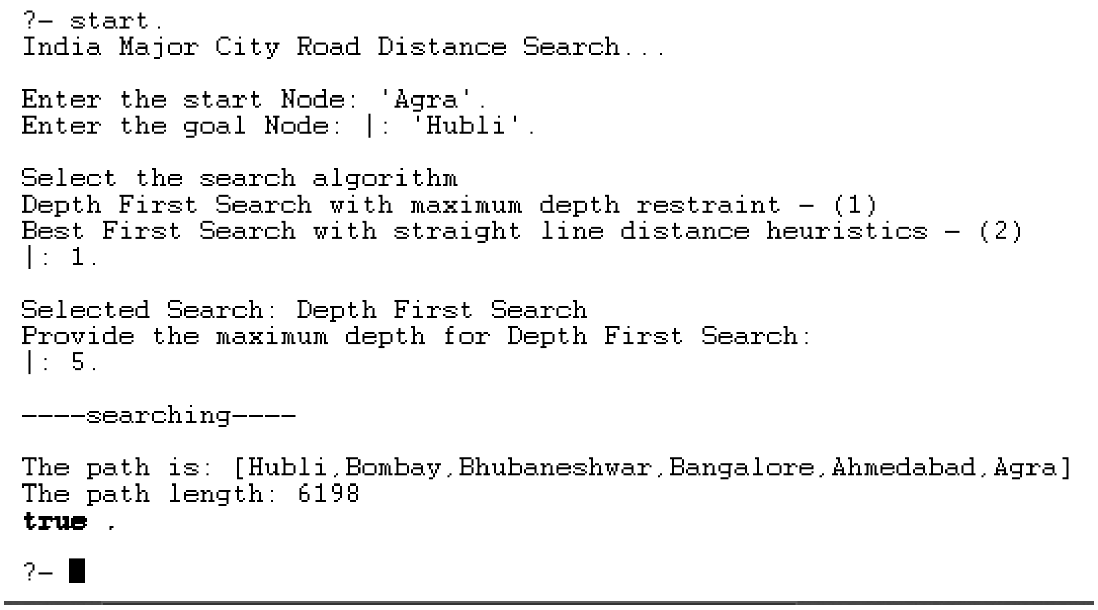
```
?- start.
India Major City Road Distance Search...

Enter the start Node: 'Ahmedabad'.
Enter the goal Node: |: 'Bangalore'.

Select the search algorithm
Depth First Search with maximum depth restraint - (1)
Best First Search with straight line distance heuristics - (2)
|: 1.

Selected Search: Depth First Search
Provide the maximum depth for Depth First Search:
|: 4.

----searching----

The path is: [Bangalore,Ahmedabad]
The path length: 1490
true .
```

3. Start: Ahmedabad, Goal: Agra



```
?- start.
India Major City Road Distance Search...

Enter the start Node: 'Ahmedabad'.
Enter the goal Node: |: 'Agra'.

Select the search algorithm
Depth First Search with maximum depth restraint - (1)
Best First Search with straight line distance heuristics - (2)
|: 1.

Selected Search: Depth First Search
Provide the maximum depth for Depth First Search:
|: 3.

----searching----

The path is: [Agra,Bhubaneshwar,Bangalore,Ahmedabad]
The path length: 4606
true .

?- 'Agra'
```

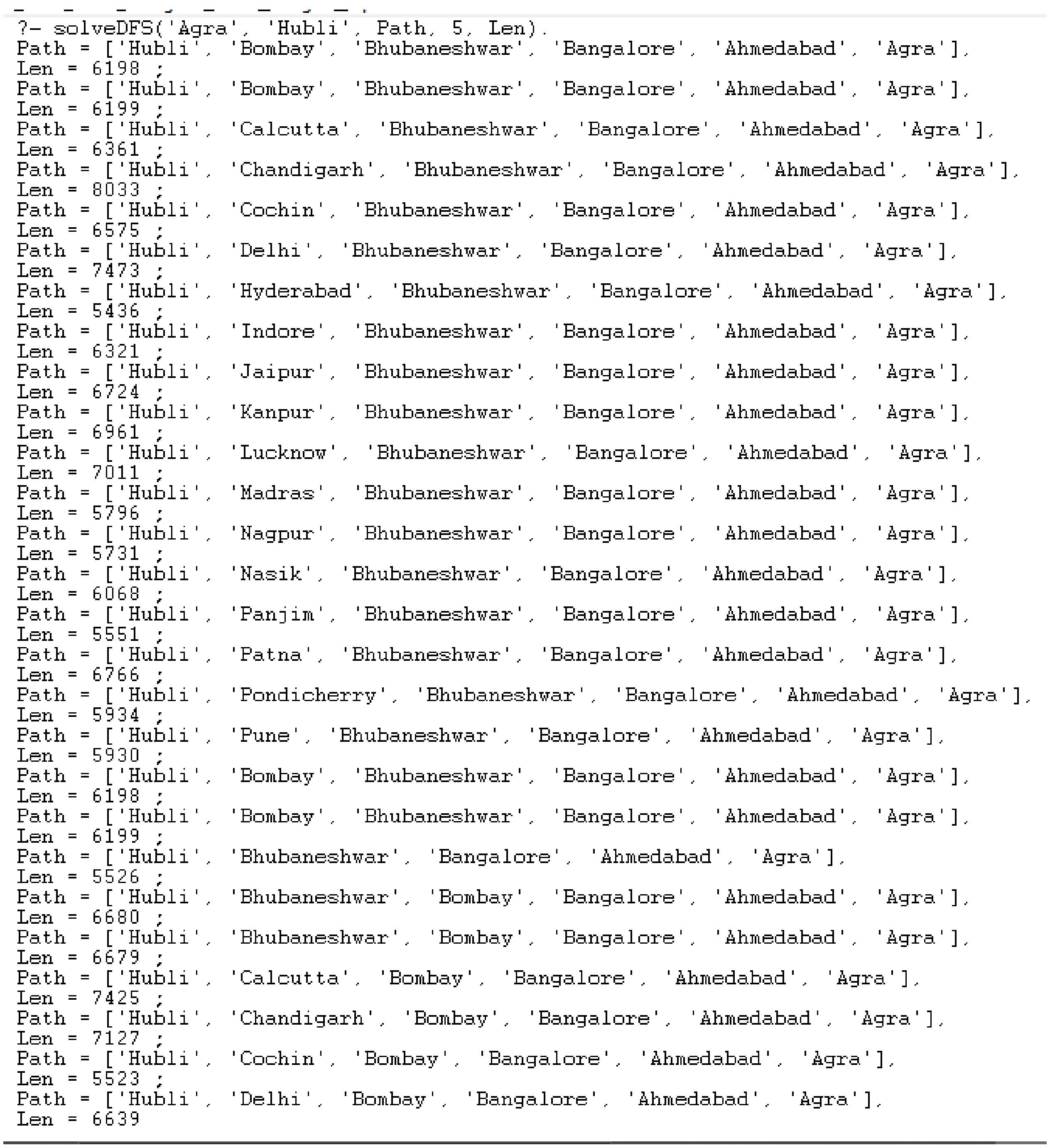
4. Start: Agra, Goal: Hubli



```
?- start.
India Major City Road Distance Search...

Enter the start Node: 'Agra'.
Enter the goal Node: |: 'Hubli'.

Select the search algorithm
Depth First Search with maximum depth restraint - (1)
Best First Search with straight line distance heuristics - (2)
|: 1.

Selected Search: Depth First Search
Provide the maximum depth for Depth First Search:
|: 5.

----searching----

The path is: [Hubli,Bombay,Bhubaneshwar,Bangalore,Ahmedabad,Agra]
The path length: 6198
true .

?- █
```

5. Start: Agra, Goal: Hubli

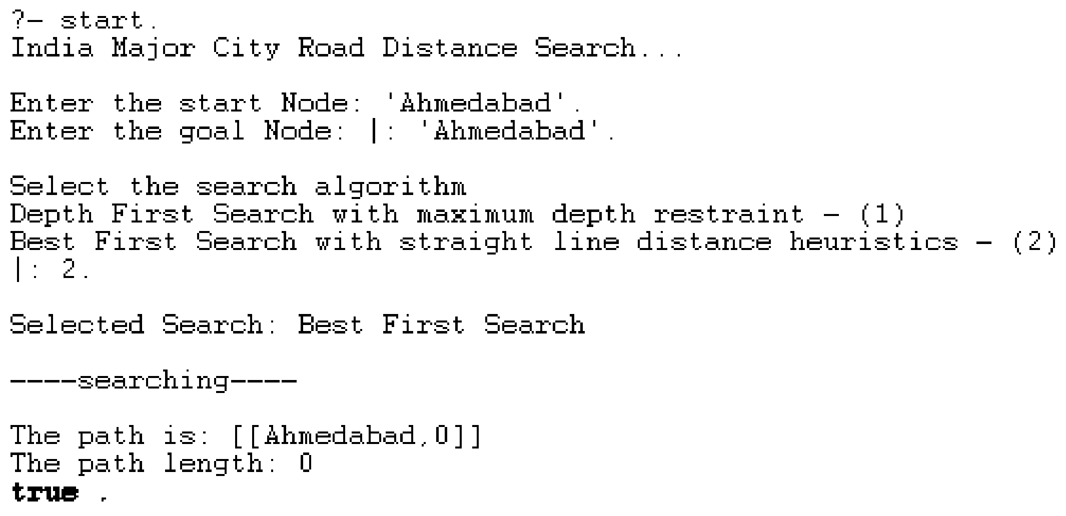
   Multiple Answers for Depth First Search using Backtracking.



```
?- solveDFS('Agra', 'Hubli', Path, 5, Len).
Path = ['Hubli', 'Bombay', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 6198 ;
Path = ['Hubli', 'Bombay', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 6199 ;
Path = ['Hubli', 'Calcutta', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 6361 ;
Path = ['Hubli', 'Chandigarh', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 8033 ;
Path = ['Hubli', 'Cochin', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 6575 ;
Path = ['Hubli', 'Delhi', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 7473 ;
Path = ['Hubli', 'Hyderabad', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 5436 ;
Path = ['Hubli', 'Indore', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 6321 ;
Path = ['Hubli', 'Jaipur', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 6724 ;
Path = ['Hubli', 'Kanpur', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 6961 ;
Path = ['Hubli', 'Lucknow', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 7011 ;
Path = ['Hubli', 'Madras', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 5796 ;
Path = ['Hubli', 'Nagpur', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 5731 ;
Path = ['Hubli', 'Nasik', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 6068 ;
Path = ['Hubli', 'Panjim', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 5551 ;
Path = ['Hubli', 'Patna', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 6766 ;
Path = ['Hubli', 'Pondicherry', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 5934 ;
Path = ['Hubli', 'Pune', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 5930 ;
Path = ['Hubli', 'Bombay', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 6198 ;
Path = ['Hubli', 'Bombay', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 6199 ;
Path = ['Hubli', 'Bhubaneshwar', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 5526 ;
Path = ['Hubli', 'Bhubaneshwar', 'Bombay', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 6680 ;
Path = ['Hubli', 'Bhubaneshwar', 'Bombay', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 6679 ;
Path = ['Hubli', 'Calcutta', 'Bombay', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 7425 ;
Path = ['Hubli', 'Chandigarh', 'Bombay', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 7127 ;
Path = ['Hubli', 'Cochin', 'Bombay', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 5523 ;
Path = ['Hubli', 'Delhi', 'Bombay', 'Bangalore', 'Ahmedabad', 'Agra'],
Len = 6639
```

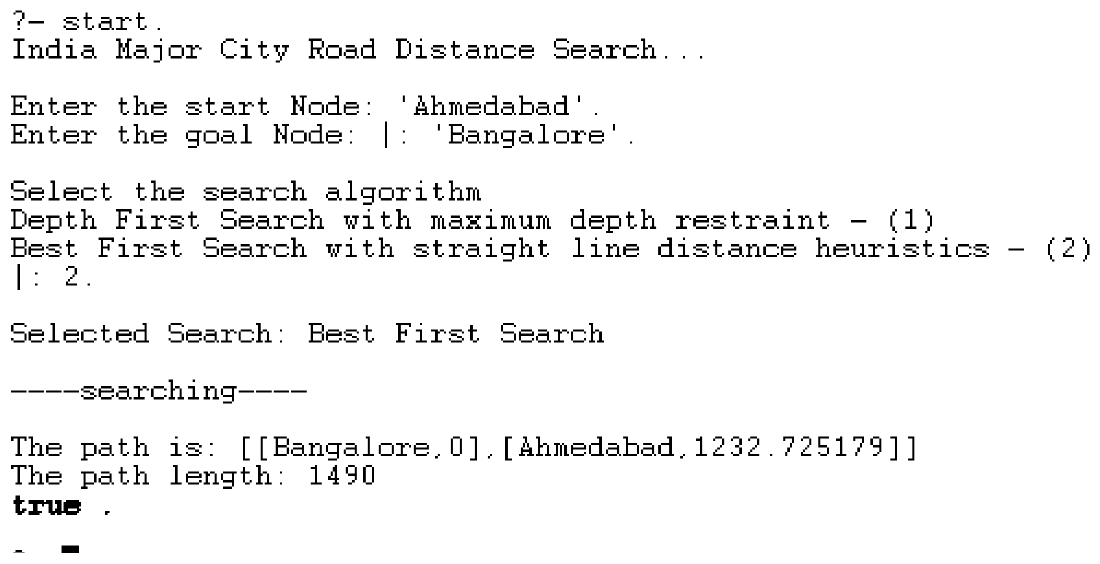
Best First Search

1. Start: Ahmedabad, Goal: Ahmedabad



```
?- start.
India Major City Road Distance Search...

Enter the start Node: 'Ahmedabad'.
Enter the goal Node: |: 'Ahmedabad'.

Select the search algorithm
Depth First Search with maximum depth restraint - (1)
Best First Search with straight line distance heuristics - (2)
|: 2.

Selected Search: Best First Search

----searching----

The path is: [[Ahmedabad,0]]
The path length: 0
true .
```

2. Start: Ahmedabad, Goal: Bangalore



```
?- start.
India Major City Road Distance Search...

Enter the start Node: 'Ahmedabad'.
Enter the goal Node: |: 'Bangalore'.

Select the search algorithm
Depth First Search with maximum depth restraint - (1)
Best First Search with straight line distance heuristics - (2)
|: 2.

Selected Search: Best First Search

----searching----

The path is: [[Bangalore,0],[Ahmedabad,1232.725179]]
The path length: 1490
true .
```

3. Start: Ahmedabad, Goal: Agra

```
?- start.
India Major City Road Distance Search...

Enter the start Node: 'Ahmedabad'.
Enter the goal Node: |: 'Agra'.

Select the search algorithm
Depth First Search with maximum depth restraint - (1)
Best First Search with straight line distance heuristics - (2)
|: 2.

Selected Search: Best First Search

----searching----

The path is: [[Agra,0],[Ahmedabad,715.7099726]]
The path length: 878
true .
```

4. Start: Agra, Goal: Hubli

```
?- start.
India Major City Road Distance Search...

Enter the start Node: 'Agra'.
Enter the goal Node: |: 'Hubli'.

Select the search algorithm
Depth First Search with maximum depth restraint - (1)
Best First Search with straight line distance heuristics - (2)
|: 2.

Selected Search: Best First Search

----searching----

The path is: [[Hubli,0],[Panjim,139.824738],[Agra,1341.506455]]
The path length: 1905
true .
```

Thank You.