

**Faculty of Engineering and Technology**

**Electrical and Computer Engineering Department**

**-** **Artificial Intelligence-**

**) ENCS3340 (**

**-Search Algorithms for Route Navigation-**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Student’s Name**: Qays Safa

**Student’s** **No**: 1190880

**Student’s Name**: Yazan AbdAlmutte

**Student’s** **No**: 1190145

**Instructor**: Dr. Adnan Yahya and Dr. Yazan Abu Farha

**Sec**: 1, 3

**Date**: 17-5-2022

**Program Implementation: -**

In our project we implement 3 search algorithms in group number 1 (DFS, BFC, A\*) and uniform search as a bonus algorithm as well as, we used python to implement our program using PyCharm.

We have 20 Palestinian cities as you can see in below and every city has unique id (Ascending from 0 to 19).

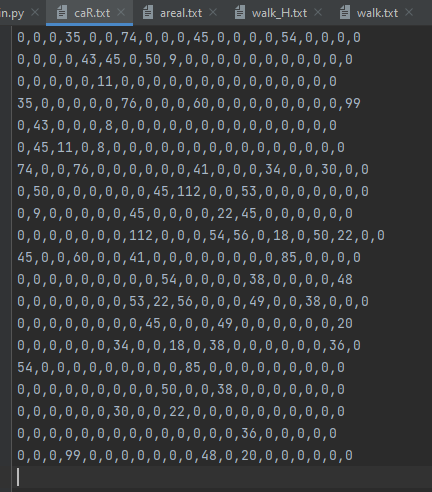
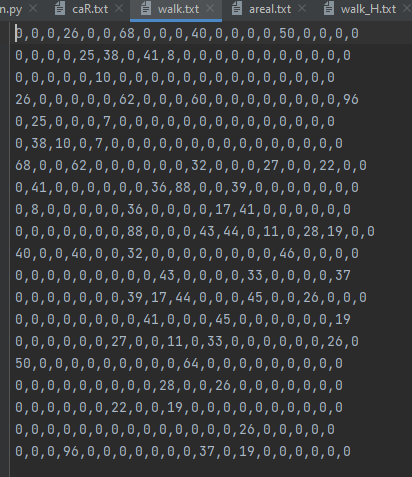
['0:AKa', '1:Bethlehm', '2:Dura', '3:Haifa', '4:Hlahoul', '5:Hebron', '6:Jenin', '7:Jericho', '8:Jerusalem', '9:Nablus', '10:Nazareth', '11:Qalqilya', '12:Ramallah', '13:Rmleh', '14:Sabastia', '15:Safad', '16:Salfit', '17:Tubas', '18:Tulkarm', '19:Yafa']

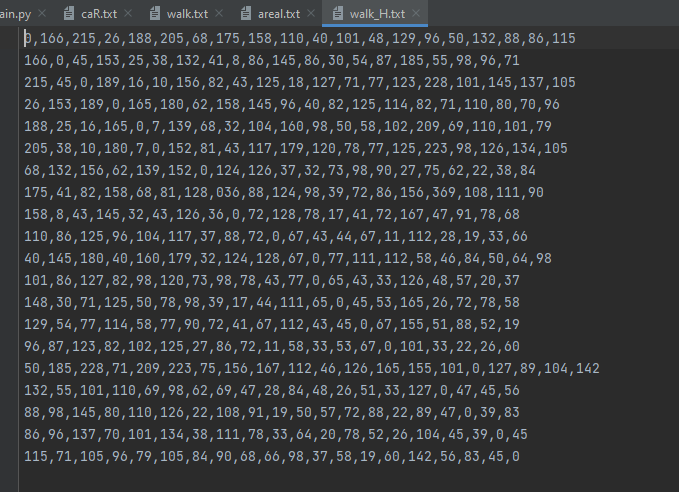
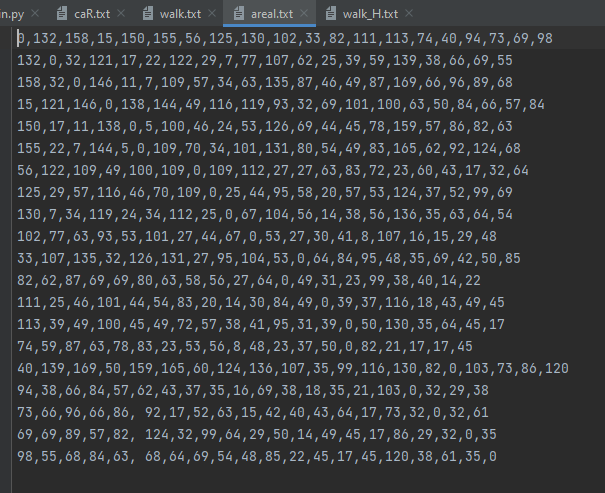
The distance between cites three kinds:

1. Car distance (Real cost)
2. Walk distance (Real cost and heuristic in some cases)
3. Areal distance (Heuristic)

So, to use these distances we insert it in four files and read it in our program

The file contains rows and columns with the number of cities, so that when we want the distance between a city and a city, we take the intersection between them, so we can get it.

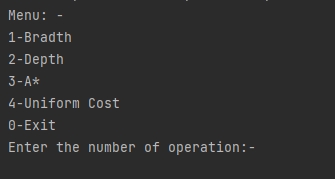




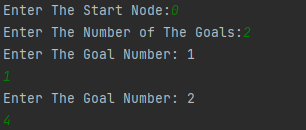
In real cost files like car and walk we add the distance between cities that just have directly connection so we can node if not directly connected we insert value 0 moreover this technique will help us to build graph and to specify connected city’s.

Note: city's file contains all city’s name and that help us to convert ids to names in future.

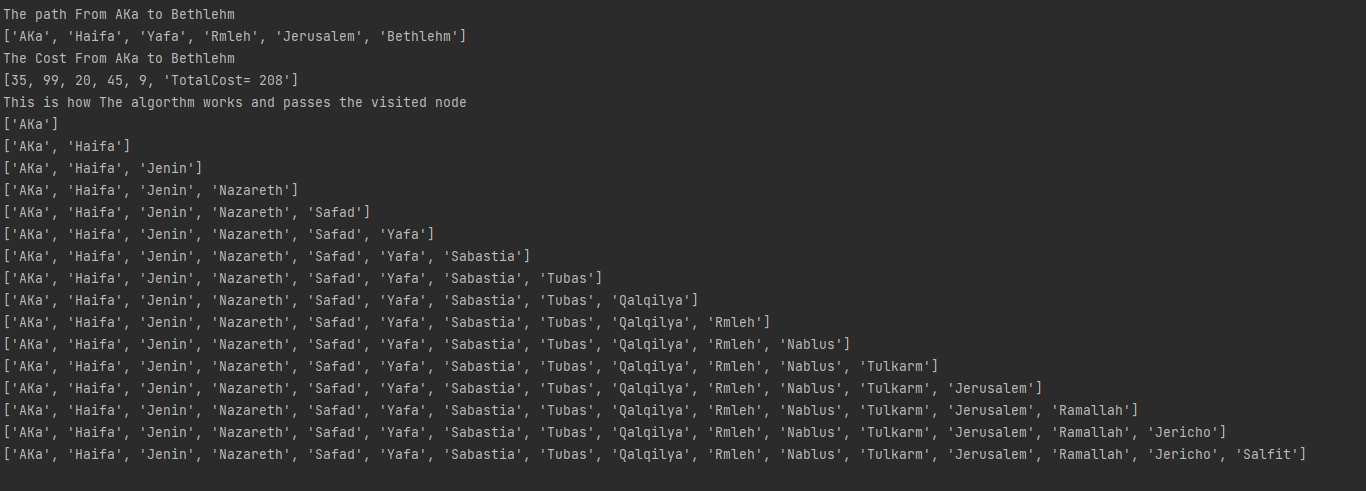
**How the Program Runs: -**

When we run the program, menu will appear to user to enter the number of algorithms he wants.

After select number operation you must enter the id of start node then enter the number of goals you want and final step to enter the id for all goals.



Note: the result will print after enter the last goal.

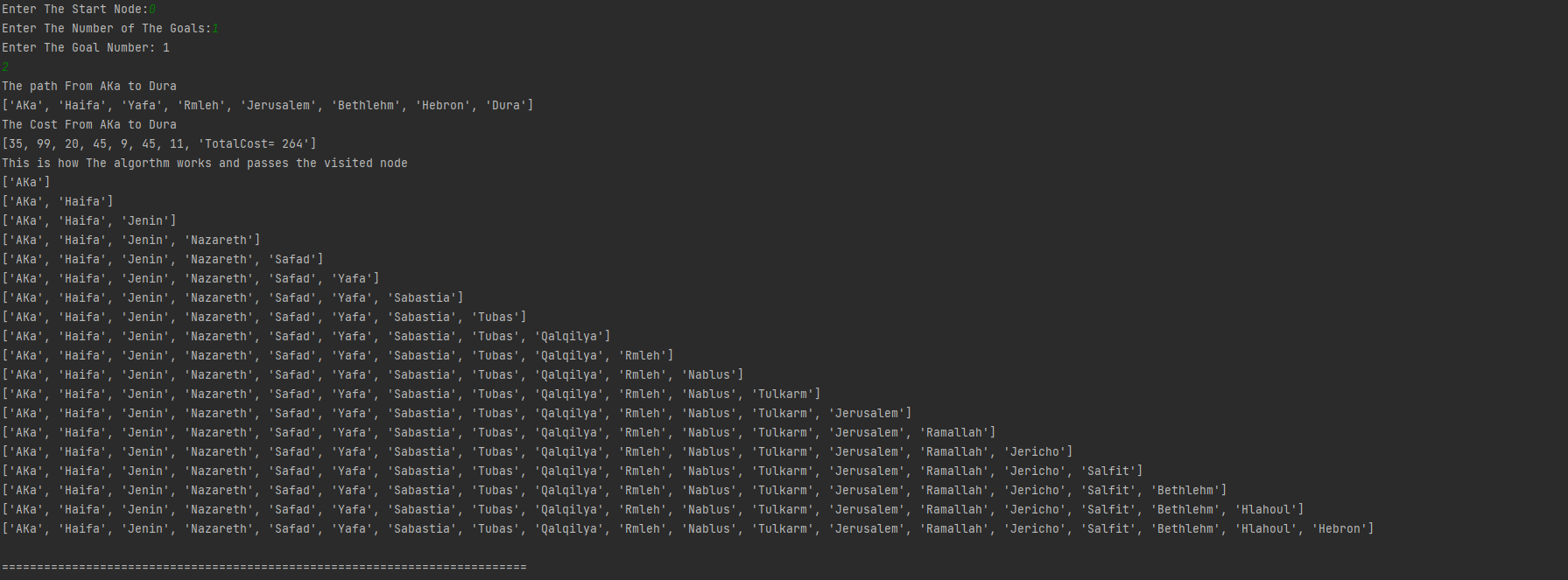
In result we will print final path and the cost of final cost moreover, we will print visited node step by step.

**Test and explain some cases:**

Let’s start with Breadth First Search: -

In BFS we don’t need know the cost but in result as an addition we calculate the total cost for the final path.

Example: -

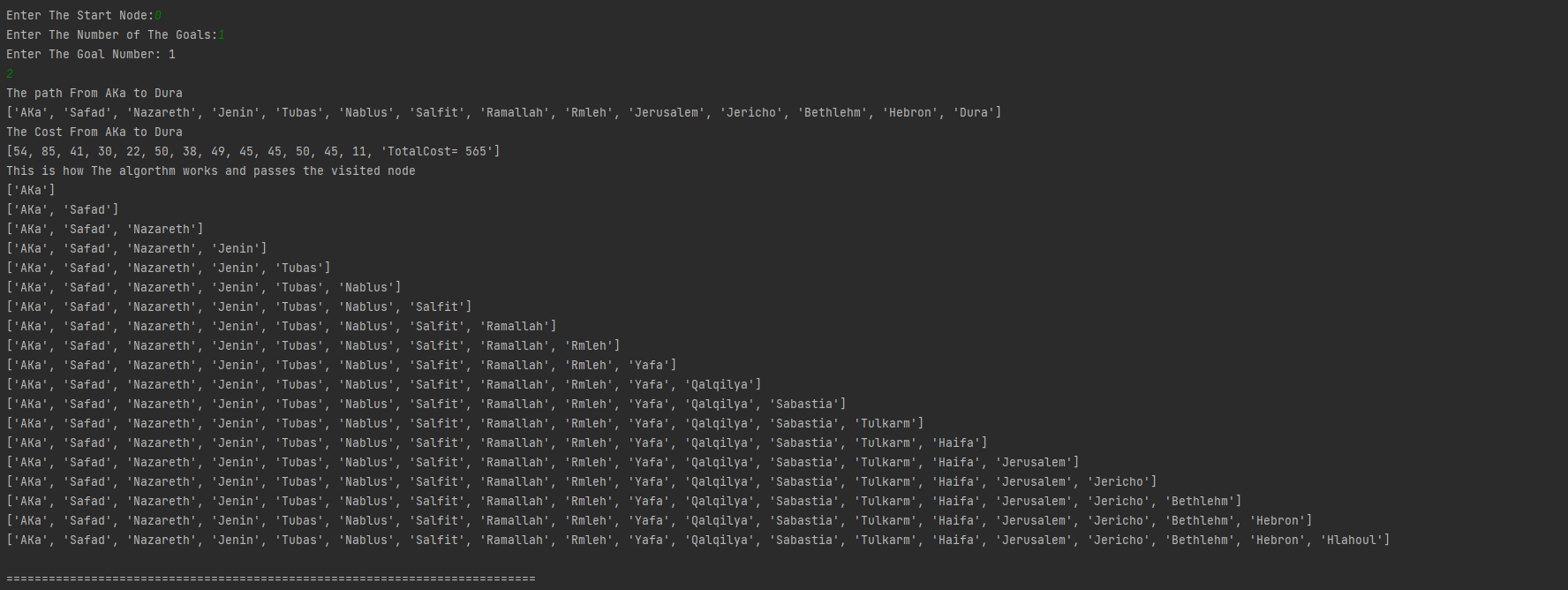
Start = Aka (0), Goal =Dura (2)

In this case we just need car.txt file to calculate the cost.

Thin In DFS (Depth First Search) we don’t need know the cost but in result as an addition we calculate the total cost for the final path.

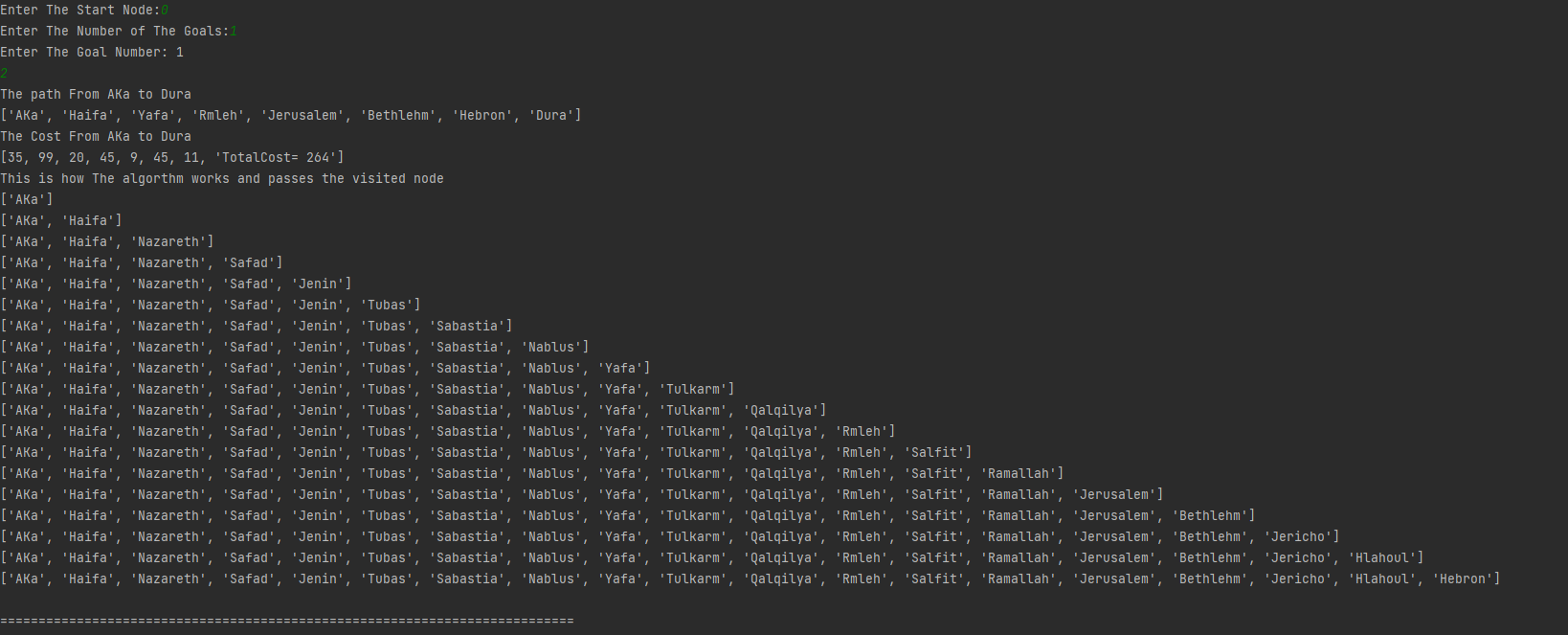
Example: -

Start = Aka (0), Goal =Dura (2)

In this case we just need car.txt file to calculate the cost.

In USC (Uniform Cost Search) we need the real cost only so we use the car.txt.

Example: -

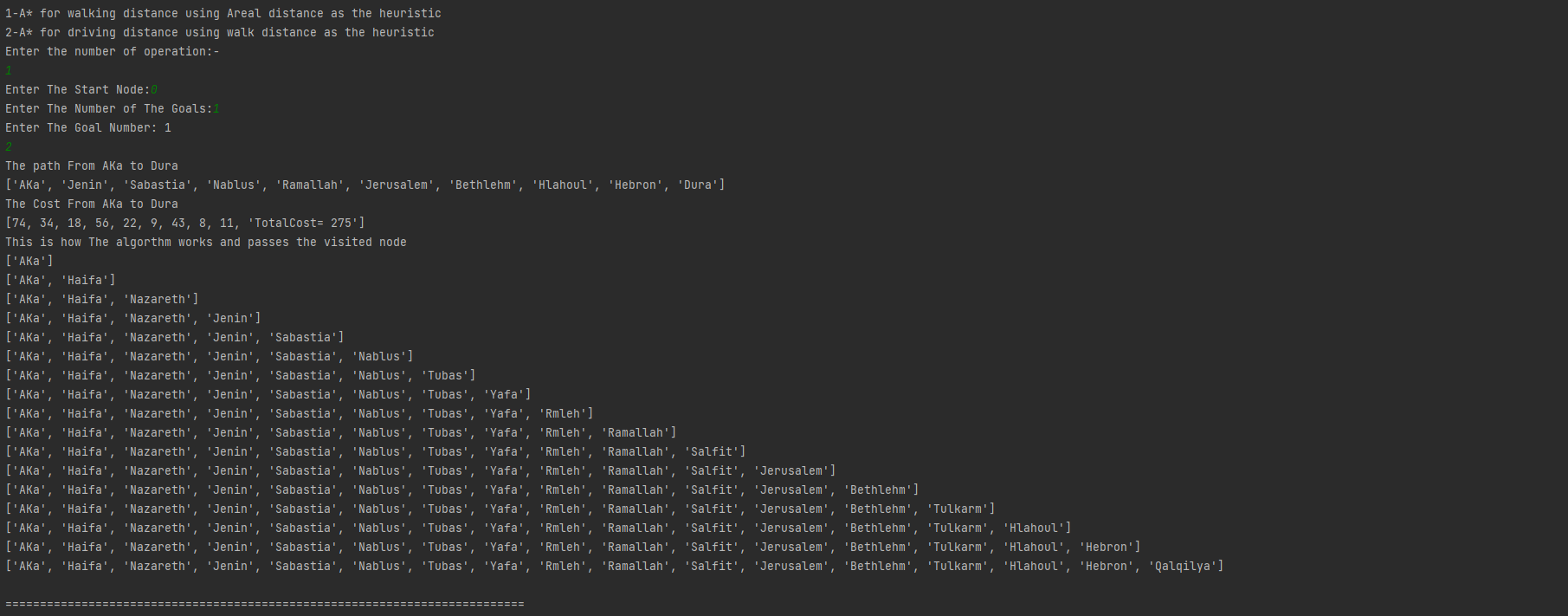
Start = Aka (0), Goal =Dura (2)

Finally, in A\* we have 2 cases:

1. A\* for walking distance using Areal distance as the heuristic

In this case we use walking distance as a real cost and areal cost as a heuristic.

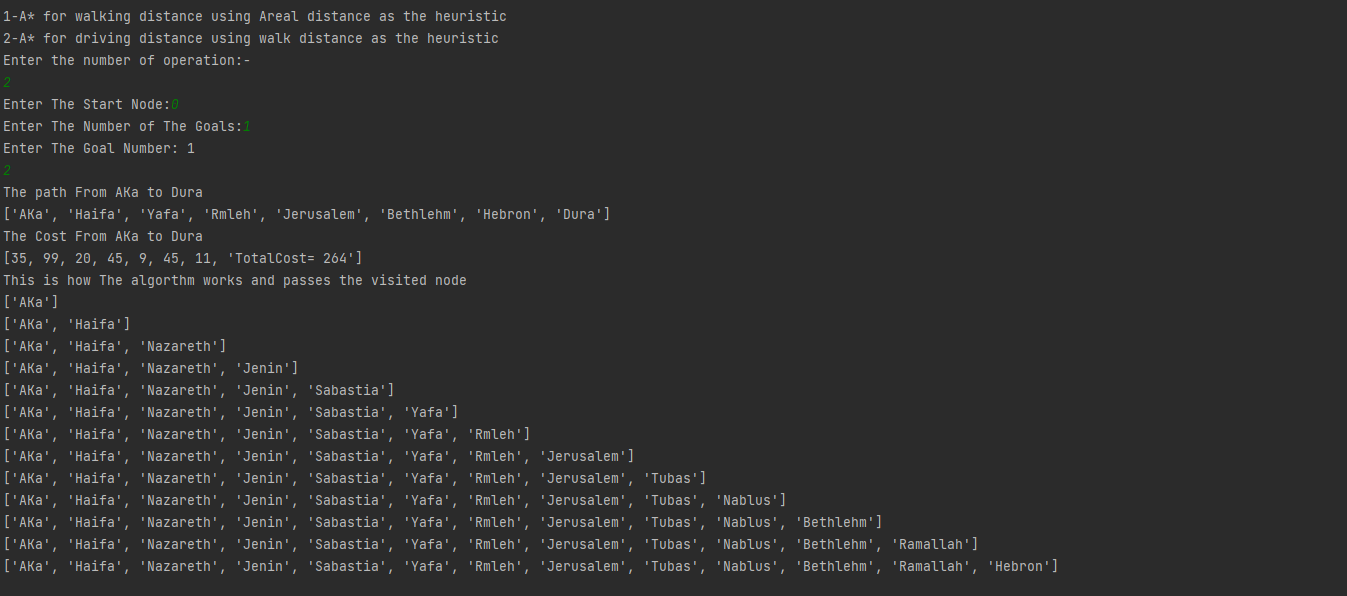
So, we need to use a walk.txt and areal.txt.

Start = Aka (0), Goal =Dura (2)

1. A\* for driving distance using walk distance as the heuristic

In this case we use driving distance as a real cost and walking distance cost as a heuristic.

So, we need to use a walk\_H.txt and car.txt.

Start = Aka (0), Goal =Dura (2)