Table of Contents

1. Graph Edge list Algorithm - Airport Problem — 1
2. Runtime analysis of methods — 1
3. Graph infrastructure — 2
4. Graph edge list Algorithm - Airport Problem

*T*he data structure used is graph edge list. So input vertices are connected using the edges. Each edge we used is connected with a start vertex and end vertex.

We tried to find out the missing edges by comparing the adjacent neighbors, whenever the adjacent neighbors are missing we are adding that as a missing edge. All these values are read from the inputPS12.txt file.

Problem Statement 12

Problem Statement 12

2. Runtime analysis of methods

|  |  |
| --- | --- |
| Method Name | Asymptotic Notation |
| read\_input | O(n) - Running time increases with size of input |
| find\_path | o(log n) + O(n^2)  - perform a linear time operation for each value in the input data |
| generate\_Edges | O(n^2) - Performing liner time operation for each value of input |
| add\_edges | O(log n) + O(1) - Running time increases with size of input and input decreases after each iteration |
| missing\_path | O(n) - Time increasing based on the input data |

3. Graph Infrastructure

A. Helper one

"""

This is the Vertex class that hold the vertices and adjacent neighbors as edges.

The list of edges contain a start vertex and end vertex, through which the code indices match for airport algo.

"""

class Vertex:

def \_\_init\_\_(self, n):

self.name = n

self.edges = list()

B. Helper Two

"""

Edges contains the start vetext and end vertex joining the indices. Start and end are the neighbours that will be linked via Graph edge list.

"""

class Edges:

def \_\_init\_\_(self, vertex):

self.destinationVertex = vertex

C. Helper Three

"""

To read the input from the input.txt file and render the values that are next to '=' symbol.

Problem Statement 12

returns a String of airport value that is added next to 'StatingAirport'.

"""

def read\_input(fromPath):

with open(fromPath) as f:

lines = f.readlines()[1]

return lines.rsplit('=', 1)[1]

"""

To read the list of airports from the inputPS12.txt file

"""

def read\_list\_of\_airports(fromPath):

with open(fromPath) as f:

lines = f.readlines()[0].splitlines()

return lines[0].rsplit('=', 1)[1]

D. Helper Four

"""

Graph edge list will have the edges and vertices stored in different arrays.

They are mapped together with edge list.

Problem Statement 12

"""

class Graph:

"""Initilazing a empty list of vertices on initializing graph"""

def \_\_init\_\_(self):

self.vertices = list()

"""Adding vertices to the list of initialzed graph. """

def add\_vertex(self, vertex):

self.vertices.append(vertex)

"""

Takes vertex one and vertext two as parameters and adds the vertices to the actual pointer location.

The graph is only one directional. So only vertext one is added to vertex two but not vice versa.

"""

def add\_edges(self, vertex\_one, vertex\_two):

return vertex\_one.edges.append(Edges(vertex\_two)) # Since graph is one directional