Programming Homework 1 REPORT

**CMPE 343/224-04**

**Information:**

**ID: -**

**Name: Begüm Şara Surname: Ünal**

**Section: 04**

**Assignment Number:1**

Table of Contents

[Problem Statement and Code Design 2](#_Toc132575251)

[Implementation and Functionality 2](#_Toc132575252)

[Question 1 2](#_Toc132575253)

[Q1 Class 2](#_Toc132575254)

[Question 2 3](#_Toc132575255)

[Main Class 3](#_Toc132575256)

[Bag class 3](#_Toc132575257)

[BagNode Class 3](#_Toc132575258)

[Graph Class 3](#_Toc132575259)

[BFS Class 3](#_Toc132575260)

[Testing 4](#_Toc132575261)

[Final Assessments 4](#_Toc132575262)

# Problem Statement and Code Design

The task of this assignment is to implement a Bag class and a DFS (depth-first search) class. The Bag class is a similar class with a linked list but does not have a remove operation. On the other hand, the DFS class takes a Graph object, and two indices as its input, and its purpose is to find if there is a path between two indices in the graph. This section summarizes the problem and sub-tasks to be completed in this assignment.

# Implementation and Functionality

In this section, I describe each sub-module of our solution.

## Question 1

In the Q1 class, rather than dividing it into classes like Graph, BFS, I did it all in one class.

### Q1 Class

First, I created a graph. Then I added this graph to my graph according to the information I received from the user. Then I got the start and end values from the user.

According to these values, I created my paths to find the shortest path using the breath first search. In Breath-first search, I've arranged to add all the neighbors of the nodes it visits a queue and select the nodes it will see in the following order.

Then, I created a path and calculated the total travel time by calculating the first value according to this path, the time required for the airports to change status from the external user, and the travel time on any flight route.

Finally, I printed the path size; paths traveled, and total travel time.

## Question 2

### Main Class

In the Main class, these are Graph and BFS (making operations on the graph for the first question) objects.

### Bag class

The Bag class is implemented using a linked list. I have implemented the following methods for the Bag class:

* Bag (): Constructor
* isEmpty: returns true if the first node is null.
* nodeSize: returns the size of the linked list.
* add (): adds a new item to the beginning of the linked list.
  + BagIterator(): Returns an iterator object which allows us to iterate over adjacent.

### BagNode Class

In that class, I defined a public Node class to keep the items. The first node of the linked list is saved in the first instance variable.

### Graph Class

This class is forcreating the graph and adding edges to this graph. Also, the Graph class uses an Adjacency List structure to implement. I have implemented the following methods for the Graph class:

* Graph (): Constructor.
* addEdge(): Adding edges to the adjList array.
* adjList(): Returns adjacent of the given vertex by converting string to integer vertex name.
* V (): Returns the number of vertices.

### BFS Class

The BFS class is implemented to find the minor path in a graph. I define a marked Boolean array to keep track of the visited vertices.

* BFS (): Constructor
* BreadthFirstSearch(): To do a breadth-first search in the graph.
* hasPathTo(): Checks if there is a path between two given vertices as parameters.
* degree ():
* pathTo(): Returns a path between given two cities. By pushing vertices to the stack.
* totalTime(): Calculate the total traveling time.

# Testing

In this section, I provide a tester class to identify the critical test points of our program. The tester class generates additional test data apart from the given sample input/output to clarify what aspects of the solution are being tested with each set. I have also described any program bugs. I have written these to describe our tests, summarize our results, and argue that they cover all types of program behavior.

# Final Assessments

In this final section, I briefly answer the following questions:

* What were the trouble points in completing this assignment?
* When doing the homework, I only had trouble calculating the arrival time of the first question.
* Which parts were the most challenging for you?
* The time calculation part of the first question was the most challenging.
* What did you like about the assignment? What did you learn from it?
* I liked that the assignment allowed us to implement a linked list and a graph search algorithm. I learned how to use the breath-first search method, depth-first search method, and HashMap classes to implement the graph search algorithm efficiently.