Algorithm Theory & Design  
COMP333

# Assignment 1: Part 2 Report

# Bradley Kenny | 45209723 John Kim | 45163782 Mark Smith | 45176655

# Stage 3

> A discussion on the general approach of the algorithm that you are using (exhaustive search,

greedy, dynamic programming).

> An explanation on how the algorithm works, including its name if it is an existing algorithm

(use an example if this helps explain your code).

> An explanation for parts of your code that may help in understanding your implementation

(do not write the entirety of your code without any explanation!).

## Part 1: computeRatio()

A simple division operation of d1/d2

double d1 = findTotalDistance(routeMinDistance(origin, destination));

double d2 = computeDistance(origin, destination);

**Lookup table**Realised that operating routeMinDistance for every operation is too expensive and have implemented a lookup table, which functions like if when looking for Chats – Mq Uni, then look up if Chatswood exists in the table, if so, f

        String name;

        if (origin.charAt(0) > destination.charAt(0)) {

            name = destination + "-" + origin;

        } else {

            name = origin + "-" + destination;

        }

## Part 2: computeAllRatio()

**Complete brute force**  
A double FOR loop, iterate through all stations in stationList, adding other stations to a HashMap  
<station I, <station J, computeRatio>>

**Add(I, j) && Add(j, i)**

When adding stations to a new Hashmap, double down on adding stations, that is:  
while operating <station I, <station J, computeRatio>>, add its counter part station (j) to the HashMap <station J, <station I, computeRatio>>

**Lookup table integration**

**Recursive findMinDist**

**Floyd Warshall: ~2.3s**

<https://www.geeksforgeeks.org/floyd-warshall-algorithm-dp-16/>

# Stage 4

## routeMinStopWithRoutes()

> An explanation of any classes or data structures that you added to the project

and how you use them in the algorithm.

Read lines data and integrate it to Station

If

# Individual Group Work Responsibilities

## Brad

## John

## Mark