

Data Structures & Algorithms

Software Year 2

CA3

Project Report

[Student Name]

[Student Number]

[Date of Submission]

**Contents**

[Introduction 2](#_Toc152329292)

[Section 1: Description of the Application 3](#_Toc152329293)

[Section 2: Data Structures Used 4](#_Toc152329294)

[Section 3: Pseudocode of operations: 5](#_Toc152329295)

[Section 4: Minimum Spanning Tree 6](#_Toc152329296)

[Section 5: Description of methods used 7](#_Toc152329297)

[Section 6: Test data Used 8](#_Toc152329298)

[Section 7: Sample execution 9](#_Toc152329299)

[Section 8: Copy of Code 10](#_Toc152329300)

[References 11](#_Toc152329301)

# Introduction

Replace this text with an appropriate Project Introduction.

Introduce the project application, and a summary of the contents of the report.

What is covered in the report? What Data structure did you use .. what language did you program in ?

Keep this section short!

This report was commissioned by Áine Byrne, lecturer Data Structures & Algorithms as submission for assessment for this module. This report is about a walking tour application for Paris. Adjacency matrix implemented using 2d arrays is the chosen data structure for the graph application. The application is written using Java.

# Section 1: Description of the Application

Drawing of graph used

Description of application-where, sites,edges?

This walking tour was designed for Paris, France. The walking tour includes 6 sites which are The Eiffel tower, Arc de Triomphe, Orsay Museum, Louvre museum, Notre Dame and Moulin rouge. From the Eiffel tower you can only go to the Arc de triomphe. From the Arc de triomphe you can go to the Eiffel tower, Orsay museum or Moulin rouge. From Orsay museum you can go to Arc de triomphe, Louvre museum or Notre dame. From Louvre museum you can go to the Orsay museum, Moulin rouge or Notre Dame.From Notre Dame you can go to the Orsay museum or the Louvre museum.From Moulin rouge you can go to the Arc de triomphe or the Louvre museum. The user will be able to search sites, search edges, add edges, find all edges and find the closest site.

# Section 2: Data Structures Used

Short description of Data Structures used, including helper variables. Include drawings of the data structures and names used for these structures.

The data structure which will be used for this application to store the graph is an adjacency matrix. The name of the adjacency matrix is edges.The weightings for this Adjacency Matrix represents the distance between the sites in kms. A 2D array will be used to implement this adjacency matrix.

A paper with numbers and symbols

Description automatically generated

# Section 3: Pseudocode of operations:

Pseudocode of each of the methods used

1. **Open and input a graph from an external file**

Input fileName

If (filename ==“graph.txt”)

{

Open file

Process file

}

Else

{

Output “incorrect file name”

}

1. **Search for a site**

Insert siteName

siteIndex = findIndex(siteName)

if (siteIndex != -1)

{

Output “Site Found”

}

Else

{

Output “Site Not Found”

}

1. **Insert an edge**

Input siteName1

Input siteName2

Input distance

edges[siteName1, siteName2] = distance

edges[siteName1,siteName2]=distance

1. **Search for an edge**

Input siteName1

Input siteName2

If (edges[siteName1, siteName2]> 0)

{

Output “There is an edge between the two sites entered”

}

Else

{

Output “There is no edge between the two sites entered”

}

1. **Given a site, display all sites connected to it**
2. **Given a site, display the closest site to it**

# Section 4: Minimum Spanning Tree

Algorithm used: Kruskal’s

**A paper with text and circles
Showing minimum spanning tree using kruskals algorithm. 
Description automatically generated**Total Cost: 11.9

# Section 5: Description of methods used

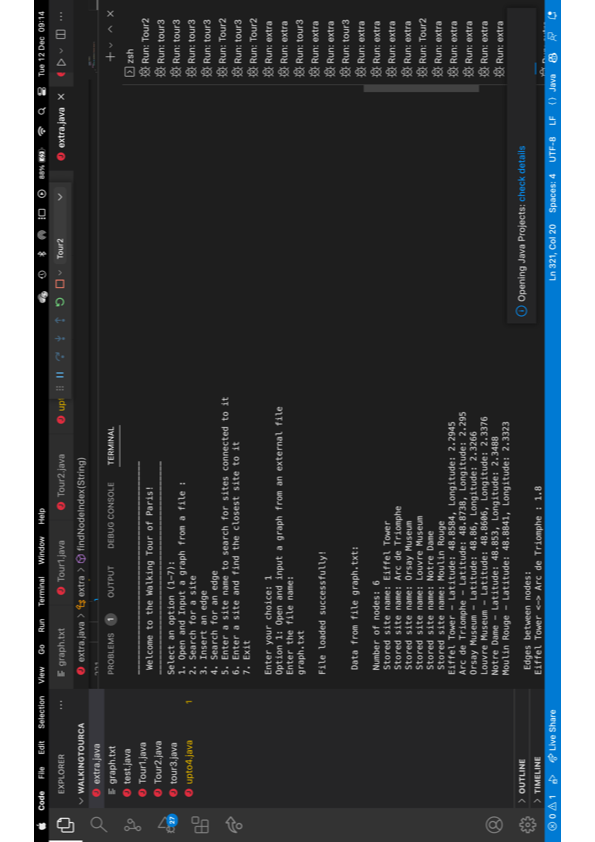
Replace this text with description of methods used.

One line description for each method used in code.

# Section 6: Test data Used

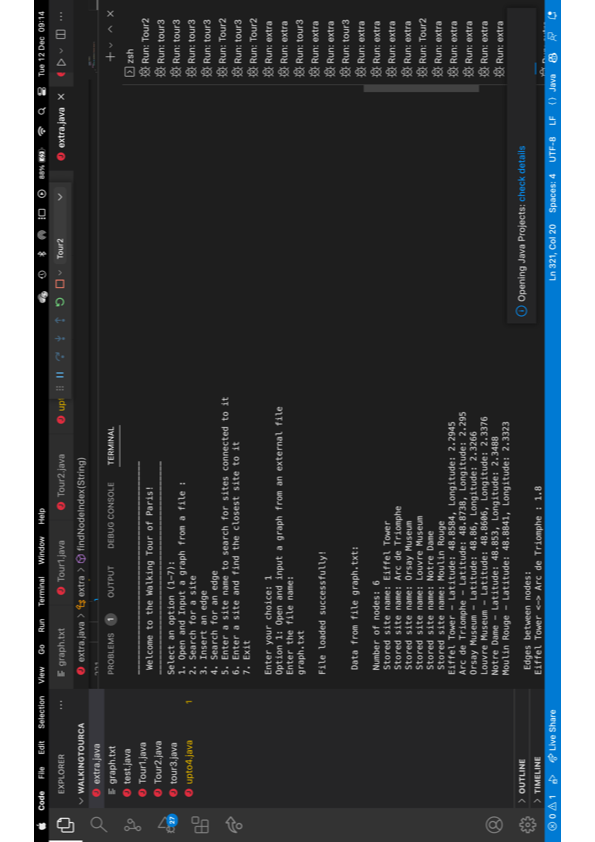
Replace this text with Test Data.

Test data is the input given to the program during test execution. Include in this section a diagram of the graph used, a description of the test file used, and a copy (or screenshot) of the test file.



# Section 7: Sample execution

The selection menu



Option 1: Open and input a graph from an external file.

User must input “graph.txt” as the file name. Terminal then prints the processed file to the screen.

If user doesn’t enter graph.txt

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedWhen user enters graph.txt terminal prints processed data to screen

Option 2: Search for a site.

A screenshot of a computer

Description automatically generatedIf user enters a correct site name (Eiffel Tower, Arc de Triomphe, Moulin Rouge,Orsay Museum, Louvre Museum, Notre Dame)

A screenshot of a computer

Description automatically generatedIf the user enters a site name that is not in the file

Option 3: Insert an edge

A screenshot of a computer

Description automatically generatedIf user enters all correct details(two valid site names which have no edge between them,valid distance)

A screenshot of a computer

Description automatically generatedIf the user enters two sites, which already have existing edge between them

A screenshot of a computer

Description automatically generatedIf the user enters a site that does not exist

Option 4: Search for an edge

A screenshot of a computer

Description automatically generatedIf user enters two valid site names and there is an edge

A screenshot of a computer

Description automatically generatedIf there is no edge

A screenshot of a computer

Description automatically generatedIf a site is not found

Option 5: Enter site and display all sites connected to it

A screen shot of a computer

Description automatically generated

Section 8: Copy of Code

Replace this text with a copy of Code.

Include a copy of Code either by including the code file or screenshots of the code.

# References

Include Acknowledge Describe Evidence Form (if appropriate)