## A REPORT ON

**Maze Solver (shortest path) using Dijkstra’s algorithm**

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE IN THE PARTIAL FULFILLMENT OF THE REQUIREMENT

FOR

# DSA MINI-PROJECT (SECOND YEAR ENGINEERING)

## SUBMITTED BY

**ANKUSH MAITY Exam No. S190224224**

**ROSHAN PIOUS Exam No. S190224293**



DEPARTMENT OF COMPUTER ENGINEERING

## ARMY INSTITUTE OF TECHNOLOGY

DIGHI HILLS, ALANDI ROAD, PUNE 411015 **SAVITRIBAI PHULE PUNE UNIVERSITY 2021-2022**



# CERTIFICATE

This is to certify that the project report entitles

**Maze Solver (shortest path) using Dijkstra’s algorithm**

SUBMITTED BY

**ANKUSH MAITY Exam No. S190224224**

## ROSHAN PIOUS Exam No. S190224293

are bonafide students of this institute and the work have been carried out by them under the supervision of **Prof. Vaishali Ganganwar** and it has been approved for the partial fulfilment of the requirement of, Second-Year course on Project Based Learning of Savitribai Phule Pune University.

**(Prof. Vaishali Ganganwar)**  **(Prof. Dr. S. Dhore)**

**Guide**  **HoD**

Department of Computer Engineering Department of Computer Engineering

ARMY INSTITUTE OF TECHNOLOGY, DIGHI, PUNE-411015

Place: Pune Date:

# ACKNOWLEDGEMENT

We are overwhelmed in all humbleness and gratefulness to acknowledge our depth to all those who have helped us to put these ideas, well above the level of simplicity and helped us develop this into something concrete. The outcome of this project required a lot of guidance and assistance from many people, and we are extremely grateful and privileged to be provided with it through all the helping entities.

We owe our deep gratefulness to our project guide **Prof. Vaishali Ganganwar**, who gave her attention and took an interest on our project work and steered us in the right direction all along, throughout our project work by providing all the necessary information for developing a good system. We are extremely thankful to her for providing her fascinating support and guidance, despite her busy schedule.

We are thankful and fortunate enough to get constant encouragement, support and guidance from all the teaching staff of the Computer Engineering Department which helped us in successfully completing our project work. Also, we would like to extend our sincere esteems to all our seniors for their timely and knowledgeable support without which completion of this project would’ve been much harder to achieve.

**ANKUSH MAITY**

**ROSHAN PIOUS**

**ABSTRACTION**

**Overview**

This report discusses the design and implementation of the game called maze path finder using Dijkstra’s algorithm. This game is created using tkinter , where initial point is give and we have to reach to the final point using shortest path and avoiding the obstructions. For the Dijkstra's algorithm allows us to find the shortest path between any two vertices of a graph. By normal method, the path will be traced wherever it finds no obstruction but by Dijkstra’s algorithm, the path with the minimum distance is traced. It differs from the minimum spanning tree because the shortest distance between two vertices might not include all the vertices of the graph.

The objective of this mini-project is to understand the concept of Dijkstra’s algorithm in Data Structures and its application and implementation in solving a maze.

**Working**

This project makes use of tkinter for Graphical user interface-based application for building the maze. We have imported the module named “PYAMAZE”, which is a maze creator with objects and walls. We apply Dijkstra’s algorithm to find the smallest path possible and trace that path.

Dijkstra's Algorithm works on the basis that any subpath B -> D of the shortest path A -> D between vertices A and D is also the shortest path between vertices B and D. Dijkstra used this property in the opposite direction i.e we overestimate the distance of each vertex from the starting vertex. Then we visit each node and its neighbours to find the shortest sub path to those neighbours.

The algorithm uses a greedy approach in the sense that we find the next best solution hoping that the end result is the best solution for the whole problem.

After finding the smallest path and tracing the path we print the Path distance using text label on the window. The obstructions(agents) are coloured, and the path avoids them and treats as wall, restricting the path from going over it.

**SOFTWARE REQUIREMENTS**

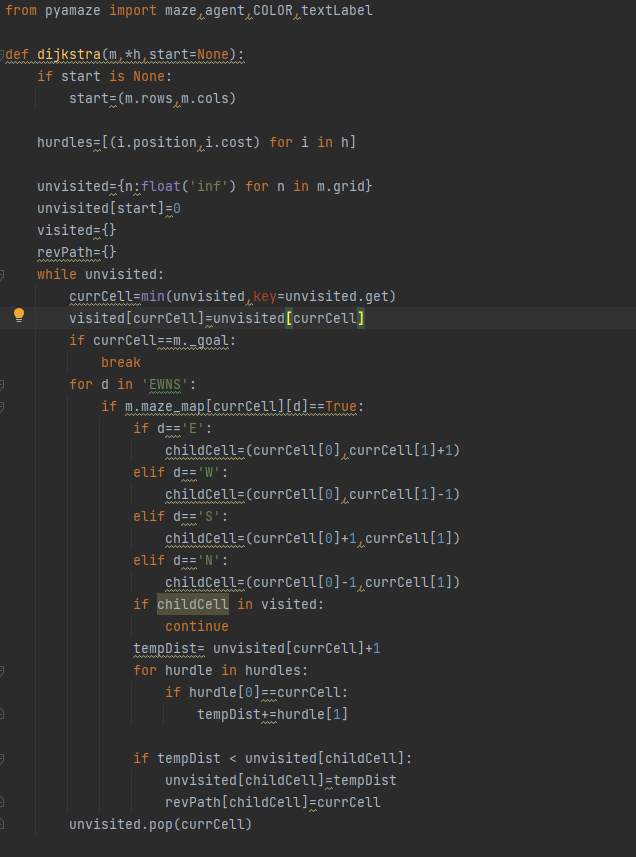
* PYTHON 3.7 and above
* IDE like Pycharm or Visual Studio Code
* Python GUI library TKINTER
* Python Module Pyamaze (For Building Maze)

**HARDWARE REQUIREMENTS**

* Quad-core processor and above
* Clock Speed of 2.4GHz and Above
* 3GB of RAM
* 1 GB of free space

**Implementation**

**Main Algorithm**

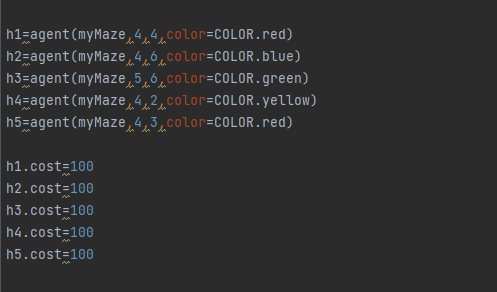


**Creating the Maze and Destination Point**

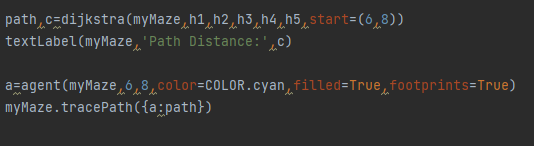
**Text

Description automatically generated**

**Creating Agents(Obstruction)**

****

**Initializing the Maze and Trace the Path**

****

**OUTPUT**

**Diagram

Description automatically generated**

**APPLICATIONS**

* This program is applied in Maps in Navigation to trace the fastest path possible and not the smallest as there may be obstructions like traffic which adds to the cost.
* It is used in IP routing to find the open shortest path first. The algorithm provide the shortest cost path from source router to other routers in the network.
* Think about how a social network app like Facebook or delivery apps. They suggest friends/potential matches to you by calculating the “distance” of those friends/shops to you. Here the users are considered to be “nodes” of the graph (social network/delivery app) and the “distance” could be defined using several factors (mutual friends, location, availability, etc).