#### A Project Report On



**COMPARATIVE ANALYSIS**

**OF**

**PATH FINDING ALGORITHMS**

A Dissertation submitted to JNTU Hyderabad in partial fulfilment of the academic requirements for the award of the degree.

## Bachelor of Technology in Computer Science & Engineering (AI&ML)

**Submitted by**

M. ABHISHEK **(20H51A6656)**

P.SAI KIRAN **(20H51A6662)**

VINAY (**20H51A6619)** Under the esteemed guidance of

**MR.P.SAI KUMAR**

(Asst. professor, Dept CSE (AI&ML))

**Department of AI & ML**

**CMR COLLEGE OF ENGINEERING & TECHNOLOGY**

(An Autonomous Institution, Approved by AICTE, Affiliated to JNTUH, NAAC ’A+’)

Kandlakoya, Medchal Road, Hyderabad

2020- 2024



**CMR COLLEGE OF ENGINEERING & TECHNOLOGY**

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD – 501401

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (AI & ML)**

**CERTIFICATE**

This is to certify that the Mini Project Phase-1 report entitled **" COMPARATIVE ANALYSIS OF PATH FINDING ALGORITHMS "** being submitted by **M.ABHISHEK (20H5A6656), P.SAI KIRAN (20H51A6662), VINAY (20H51A6619)** in partial fulfilment for the award of **Bachelor of Technology in Computer Science and Engineering (AI&ML)** is a record of bonafide work carried out his/her under my guidance and supervision .The results embodied in this project report have not been submitted to any other University or Institute for the award of any Degree.

**MR.P.SAI KUMAR Dr. P. SRUTHI**

**(Assistant Professor) (Associate Professor and HOD) Dept. of CSE (AI&ML) Dept. of CSE (AI&ML)**

**ACKNOWLEDGMENT**

With great pleasure I want to take this opportunity to express my heartfelt gratitude to

all the people who helped in making this project work a grand success.

We are grateful to **Mr. P. SAI KUMAR,** Assistant Professor, Dept. of Computer Science and Engineering (AI & ML) for her valuable suggestions and guidance during the execution of this project work.

We would like to thank, **Dr. P. SRUTHI,** Associate Professor & Head CSE (AI&ML), CMR College of Engineering and Technology, who is the major driving forces to complete my project work successfully.

We are very grateful to **Dr. Vijay Kumar Koppula**, Dean-Academics, CMR College of Engineering and Technology, for his constant support and motivation in carrying out the project work successfully.

We are highly indebted to **Major Dr. V.A. Narayana**, Principal CMRCET for giving permission to carry out this project in a successful and fruitful way.

We express our sincere thanks to **Mr. Ch. Gopal Reddy**, Secretary, CMR Group of Institutions, for his continuous care.

Finally, we extend thanks to our parents who stood behind us at different stages of this Project. We sincerely acknowledge and thank all those who gave support directly and indirectly in completion of this project work.

M.ABHISHEK **(20H51A6656)**

P. SAI KIRAN **(20H51A6662)**

VINAY (**20H51A6619)**

## TABLE OF CONTENTS

|  |  |  |  |
| --- | --- | --- | --- |
| **CHAPTER NO**  **1** | **TITLE**  LIST OF FIGURES  ABSTRACT  **INTRODUCTION** | **PAGE NO**  5  6 |  |
|  | * 1. Literature review | 9 |  |
|  | * 1. Problem Statement | 10 |  |
|  | 1.3 Research Objective  1.4 Project Scope and Limitations | 10  11 |  |
| **2** | **BACKGROUND WORK**  2.1 Path finding algorithms efficiency  analysis in 2D grid | 13 |  |
|  | 2.1.1. Introduction | 13 |  |
|  | 2.1.2. Merits , Demerits and Challenges  2.1.3. Implementation of path finding algorithms efficiency analysis in 2D grid | 14  15-17 |  |
| **3** | **SYSTEM DESIGN**  3.1 Proposed system Introduction | 19 |  |
|  | 3.2 Proposed system Architecture | 20 |  |
|  | 3.3 Advantages of proposed methods  3.4. Implementation of Comparative analysis of  Path finding algorithms | 20  21 |  |
| **4** | **RESULTS AND DISCUSSION**  4.1 Data Collection and Performance metrics | 32 |  |
|  | 4.2 Comparison of Existing Solution  4.3. Results and screen shots | 33  34 |  |
| **5** | **CONCLUSIONS** |  |  |
|  | 5.1 Conclusion | 38 |  |
| **6** | **REFERENCES** | 40 |  |

**LIST OF FIGURES**

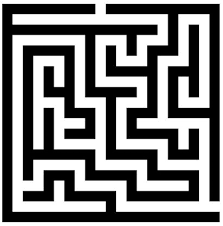
**FIGURE**

**NO. TITLE PAGE NO.**

|  |  |  |
| --- | --- | --- |
| 1 | Maze image | 7 |
| 2.1.5(a) | Heap node and PQ diagram | 16 |
| 2.1.5(b) | *Dequeue* | 17 |
| 3.1 | Grid image | 19 |
| 3.2 | Architecture | 20 |
| 3.4(a) -3.4.1(d) | Code | 21-30 |
| 4.1 | Python | 32 |
| 4.3 | Maze image | 34 |
| 4.3(a) | Output | 35 |
| 4.3(b) | Output | 35 |
| 4.3(c) | Output | 36 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## ABSTRACT

The proposal is to build a 2D game and comparision between Breadth First Search (BFS) and A\* and Depth First Search(DFS) algorithm used in this maze game . Here character need to go from start point to goal point having obstacles in between , painted in black in the path using the above algorithms , an optimal path is found out for the character to reach the goal point . Since we know using A\* algorithm we can find optimal shortest path to reach the goal than BFS and DFS algorithms , so we will optimize the BFS and DFS and compare the performance between the three approaches . The main goal of the project is to implement the maze game for finding the shortest path and use it for further purpose by using A\* and Breadth First Search and Depth First Search algorithm and analyze and compare the performance of above approaches.



*FIG 1. MAZE*



# CHAPTER 1

# 

# INTRODUCTION



### LITERATURE REVIEW

1. Research conducted by Singhal and Harish with the title, “A Review of the Paper of Navigation and Pathfinding using Mobile Cellular Automata” describes pathfinding, cellular automata and software agents. In this analysis pathfinding algorithms were studied and that they conjointly studied finding the space between two nodes. This analysis focuses on cellular automata and for advanced modeling systems and computing tools. This analysis uses BFS algorithmic rule, Dijkstra's algorithmic rule, and Astar search algorithms to seek out the gap between the cellular automata .The study titled, “Pathfinding car racing game using dynamic pathfinding algorithm and algorithm A\*” conducted by Sazaki, etal used a pathfinding algorithmic program for NPC to race against players in a very racing game .The pathfinding technique utilized by NPCs during this game is A star algorithmic rule and Dynamic Pathfinding formula, here Astar algorithmic rule was accustomed realize the shortest path on the track while the Dynamic Pathfinding formula was accustomed avoid static or dynamic obstacles in its path. The experimental leads to this study show that the mix of each strategies is enforced well in racing games with the track conditions being blocked with static obstacles .While moving on the track with dynamic obstacles, the mix of each strategies passes through the course solely below bound conditions
2. Pathfinding Algorithm Efficiency Analysis in 2D Grid” by Zarembo and Sergejs in this

The path finding algorithm efficiency in 2D grid is being done which says us about the number of nodes and the time complexity of the algorithm.



* 1. **PROBLEM STATEMENT**

The main goal is to develop a maze for finding the shortest path for the given maze with obstacles and comparing the path finding algorithms within them and produce the result

### RESOURCE OBJECTIVE

The comparative analysis of the algorithms is used for the analyses of the algorithm and comparision of the algorithm which makes a brief understanding about the algorithms involved and the implementation process and find the shortest path of the labyrinth from the start node to goal node by implementing the path finding algorithms.



### PROJECT SCOPE AND LIMITATIONS

**SCOPE**:

* We can improvise the labyrinth of different type and completely difficult path for

Further.

* We will be using the different input and output mazes for game purpose and make it into

Dynamic path which make the game more interesting and improvise the GUI .

* We will improve this as a game by adding a playable character and also by adding

Better version of obstacles in the path to reach the goal state, present some of the

Gifts for player to make up the mood for further playing and make each level harder for the player to grab the interest of the player.

**LIMITATION**:

* The path some time will be almost same for the different algorithms and the number of nodes taking for the different algorithms are high at maximum time.
* The time taking for the algorithms is high for the dfs
* The algorithm has its own limitations like the time complexity .



**CHAPTER 2**

# BACKGROUND WORK

# 

* 1. **PATH FINDING ALGORITHMEFFICIENCY ANALYSIS IN 2D GRID**

### INTRODUCTION

These days, AI has spread through the game world. Some games have started implementing AI in its game. Varying from action games, adventure, action adventure, RPG, simulation, strategy, sports, to idle game. AI spreads in all kinds of growing game genres. The use of AI can bring revenue from the game. Players can feel challenged and buy some of the facilities provided by the game. It is not rare for games that use AI to be better than games that do not use AI . Maze Runner game is a labyrinth created by players. The player has to block the path of the NPC that has been given a pathfinding algorithm with Tetris-like block. Players are provided with various blocks with these letters to block the course of the NPC. The more blocks traversed by the NPC, the more score the player will get. Therefore, the NPC needs the fastest path to the destination node. A Pathfinding algorithm is required to determine the fastest path possible. There are several algorithms to find the fastest path. The algorithms are A\*, Breadth First Search (BFS). These algorithms are the best pathfinding algorithms . Each of these algorithms has their own weaknesses and strengths in the process of determining the fastest path. This research will find the best algorithm for the Maze Runner game. The A\* algorithm is one of the search algorithms that analyze inputs, evaluates a number of possible paths and generates solutions. The A\* algorithm is a computer algorithm that is used extensively in graph traversal and the search for paths along with the efficient path planning process around the points called nodes . Breadth First Search is a method that performs a wider search that extends a node pre orderly, extending a node and then extending all neighbors of the last node. After that, extend the un extended nodes and is neighboring the extended nodes, and so on. The comparison process in this algorithm was conducted by measuring 3 variables of pathfinding computation. These three variables are the measurement of the process time, the length of the path, and the numbers of blocks played in the existing computing process. In the process of measuring time, time will be measured in milliseconds (ms). The calculated time is the time needed from start node to destination node. The time will start when NPC starts running. The length of the path is measured by counting the number of blocks traversed by the NPC. Blocks played in the computation process are calculated from the number of blocks (nodes) counted before the pathfinder determines its path. The purpose of this research will determine the best algorithm to be applied in this Maze Runner game. The algorithm which will be recommended to be applied in the game is considered to be taken from various aspects of the comparison test results. The results of this research are expected 8to help game developers to implement the best algorithm for making maze-based games.



### MERITS, DEMERITS AND CHALLENGES

**MERITS:**

* This model is very much useful for 2d grids which make it take the input of any kind of labyrinth drawn handy.
* It will calculate the heuristic value for the path and analyze the path.

### DEMERITS AND CHALLENGES:

* The path will be found for Only 2d grids.
* Cant handle larger complex grids.
* Once the process starts it cant be stopped until it reaches the goal state.



### IMPLEMENTATION OF PATH FINDING ALGORITHM

### EFFICIENCY ANALYSIS IN 2D

In our project we will compare A\* and BFS and DFS algorithms and compare which algorithm provides the optimal path for the given maze image.

**Step1**: Here we will consider a maze that will be given as input. We have considered the maze of different sizes. To implement these algorithms we have used some data structures like priority queue and dequeue.

**Step2**: After the execution of our code DFS and A\* and BFS will create the path for the goal state that which is considered as the output.

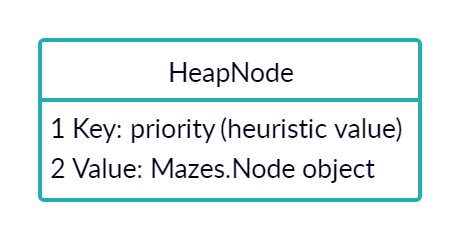
**Step3**: for each maze we will get a different output with the 3 different algorithm , based on the results obtained we can compare the nodes explored by each algorithm and finally conclude the best algorithm that provides the optimal path

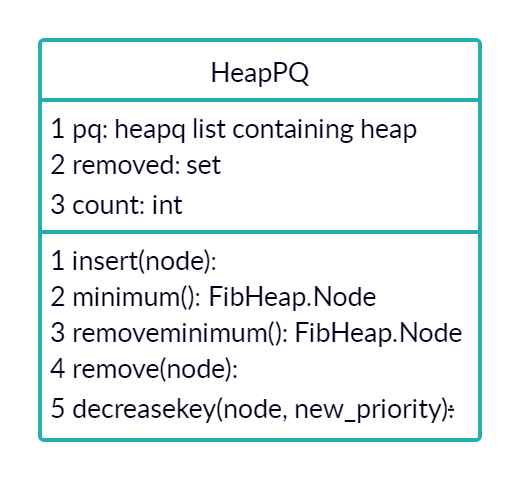
**2.1.5.Data structures:**.

1. **Priority queue(heap):**

For the implementation of A\* search algorithm we have used priority queue. A priority queue is a special type of queue in which each element is associated with a priority and is served according to its priority. If elements with the same priority occur, they are served according to their order in the queue. Generally, the value of the element itself is considered for assigning the priority. For example, The element with the highest value is considered as the highest priority element. However, in other cases, we can assume the element with the lowest value as the highest priority element. In other cases, we can set priorities according to our needs.

The min heap data structure is used in priority queue with priority used as key and value as Maze. Node object. The priority in the case is the distance from the start node and current node and the distance from the current node and end node. (Note that manhattan distance is used here). To hold the key and value together, Heap Node class is used. It has two properties i.e; key and value which are the same as that of Priority Queue and makes the code easy and more readable.

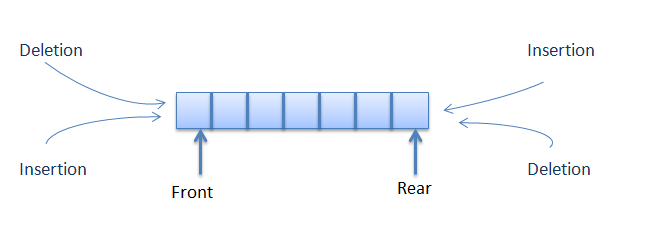




*FIG2.1.5(a):* .Heap node and pq diagram

1. **Dequeue:**

For implementing the Breadth First Search algorithm we have used a queue called dequeue. A **deque**, also known as a double-ended queue, is an ordered collection of items similar to the queue. It has two ends, a front and a rear, and the items remain positioned in the collection. What makes a deque different is the unrestrictive nature of adding and removing items. New items can be added at either the front or the rear. Likewise, existing items can be removed from either end. In a sense, this hybrid linear structure provides all the capabilities of stacks and queues in a single data structure. In our project deque data structure is used to implement the BFS Algorithm. Nodes are inserted from the left and are popped from the right. This makes the implementation simple.



*FIG2.1.5(b). Dequeue*



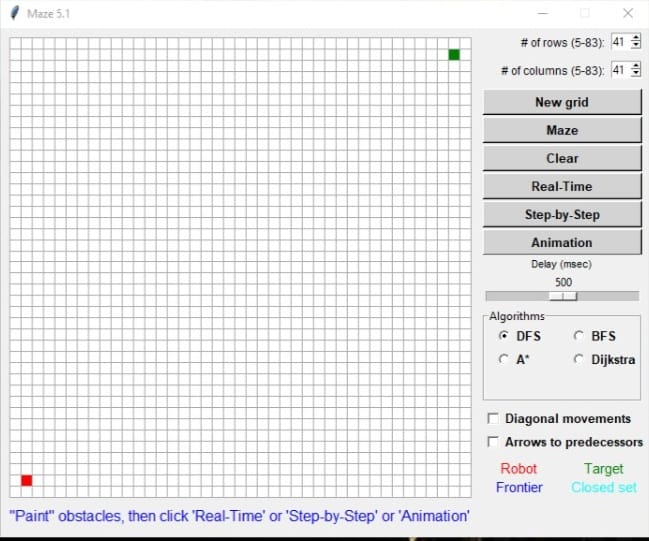
# CHAPTER 3

# SYSTEM DESIGN



### PROPOSED SYSTEM INTRODUCTION:

Finding the shortest path in a maze in less time and by iterating a minimum number of nodes is a difficult problem. As there are many algorithms used in solving this problem, the performance of algorithms differs accordingly. So, in this project we have used three algorithms and compared the performance between them. The maze finding problem is outlined as finding a path from purpose(place to begin) to the ending point in a very means that the trail therefore found should be the best optimal path.

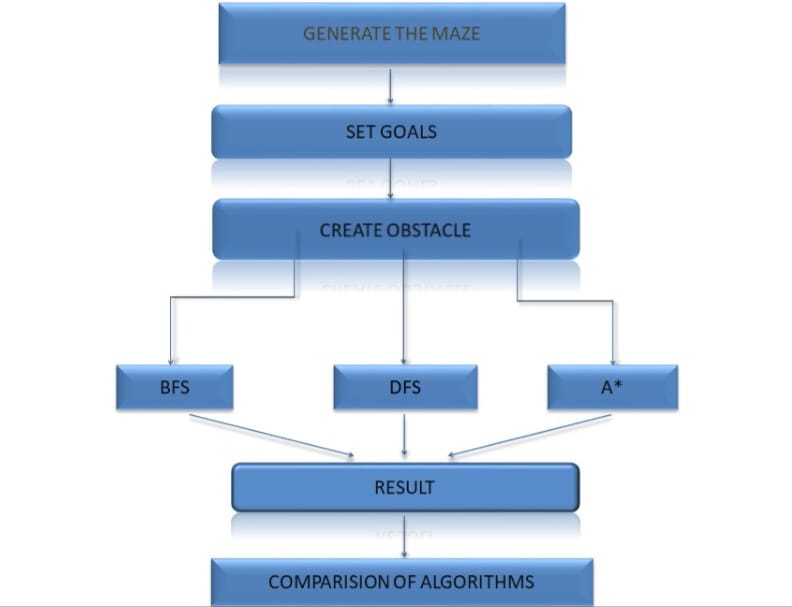


*FIG3.1 grid image.*

Here the White block in the maze image represents a path, which will correspond to a node in the graph and the black block represents a dead end or no path in the maze image. Using the above two approaches we will find the algorithm that gives us the optimal path by exploring less number of nodes. Here we have considered a set of large and small maze images as input. Based on the outputs given by A\* search algorithm and breadth first search and depth first search algorithms we can find the algorithm that gives the shortest path by exploring less number of nodes and the algorithm that is more efficient



**3.2 Proposed system Architecture:**



*Fig.3.2.Architecture*

**3.3 Advantages of Proposed methods**

Voice Input

WhatsApp

Our proposed model has the following features

* Simple and customizable.
* It can show the dfs path from the initial to goal state .
* It also show the step by step process for reaching the goal node.
* It can automatically create the maze of different forms.
* It also shows the animation of the path from current to goal node.



**3.4 Implementation of Comparative analysis of path finding algorithm**

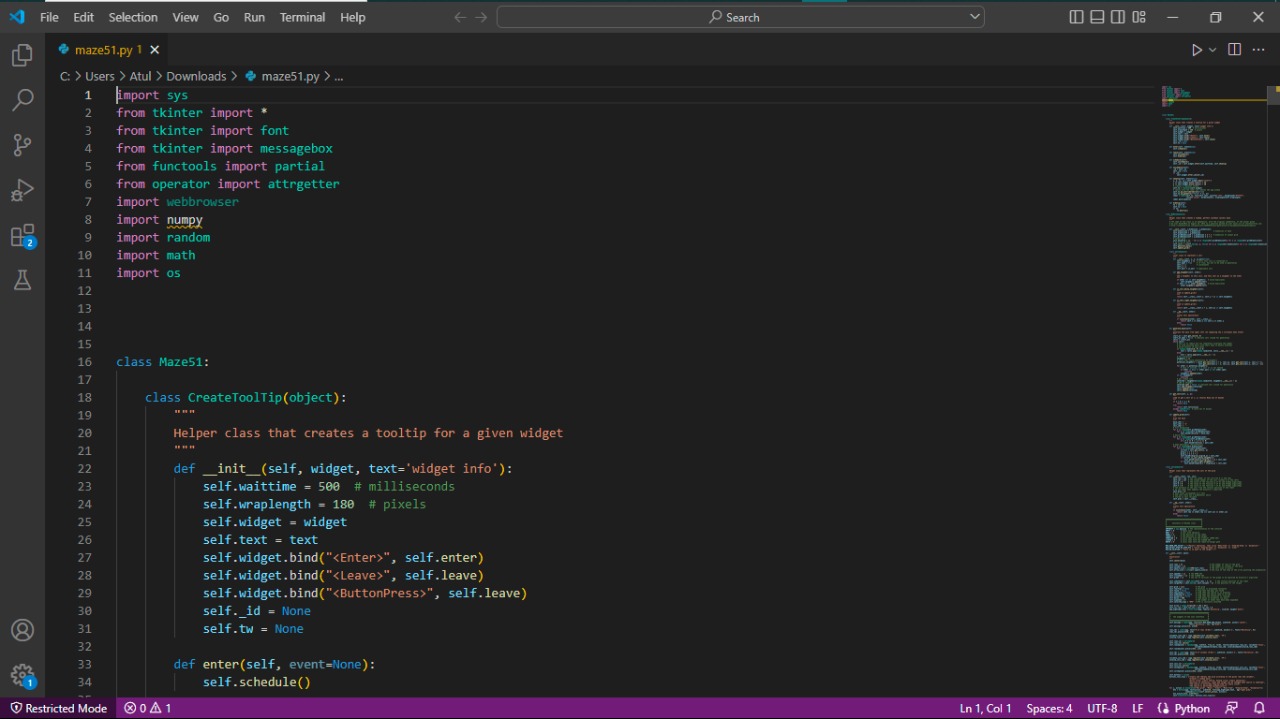
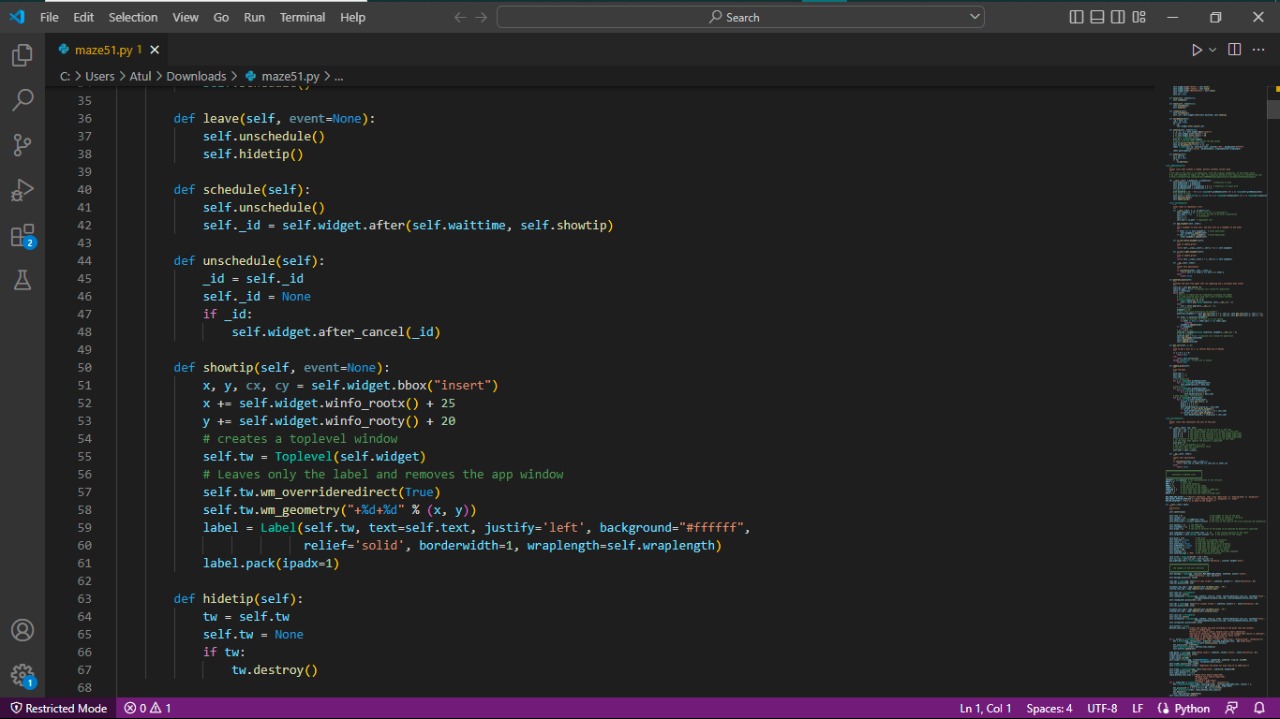
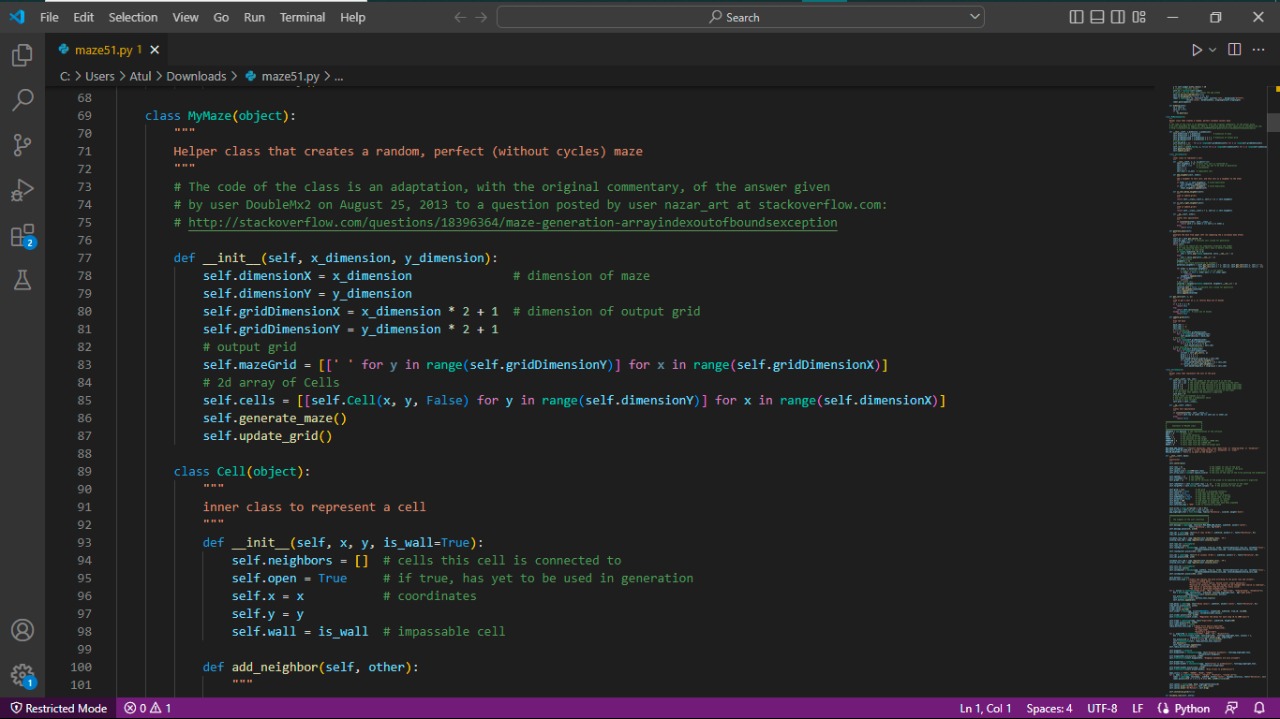
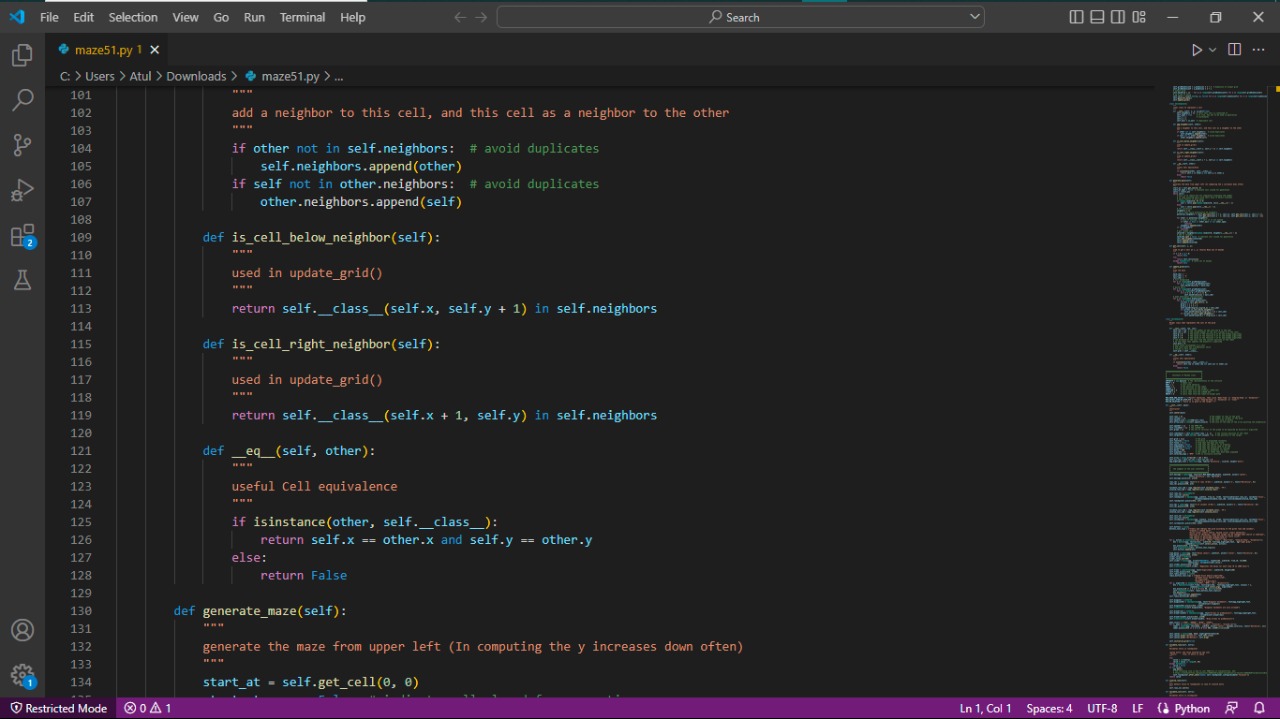
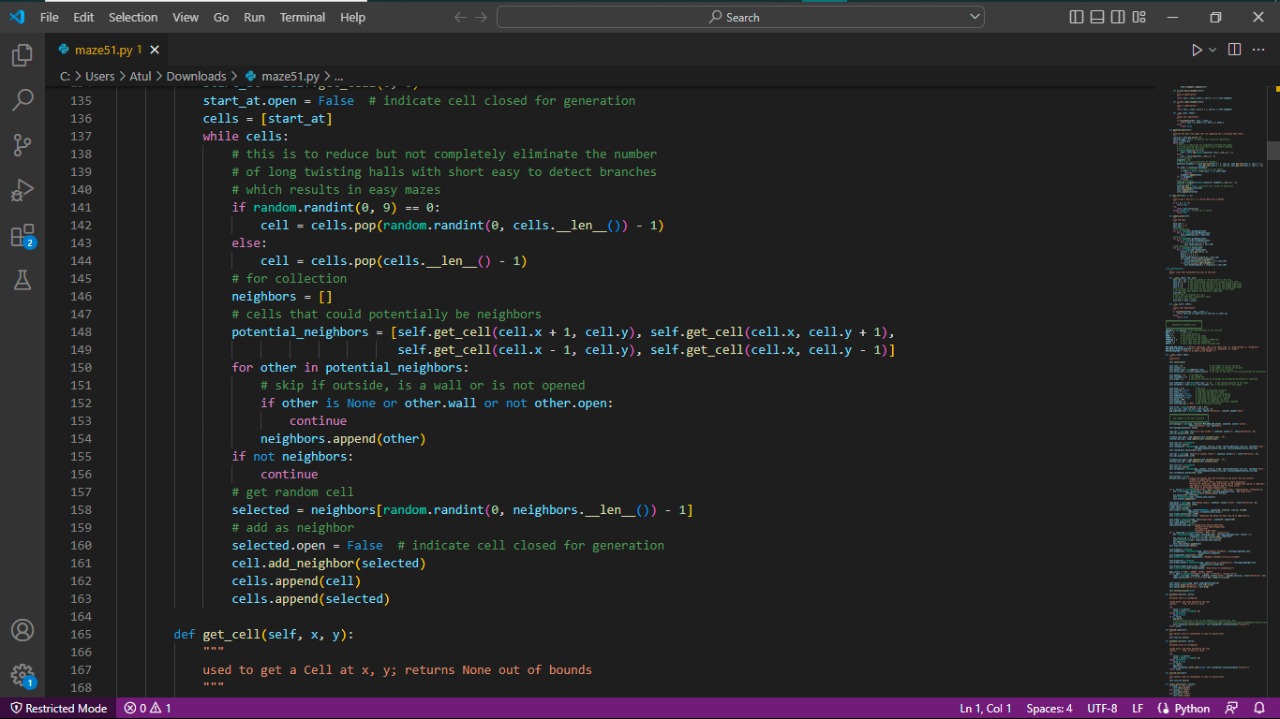
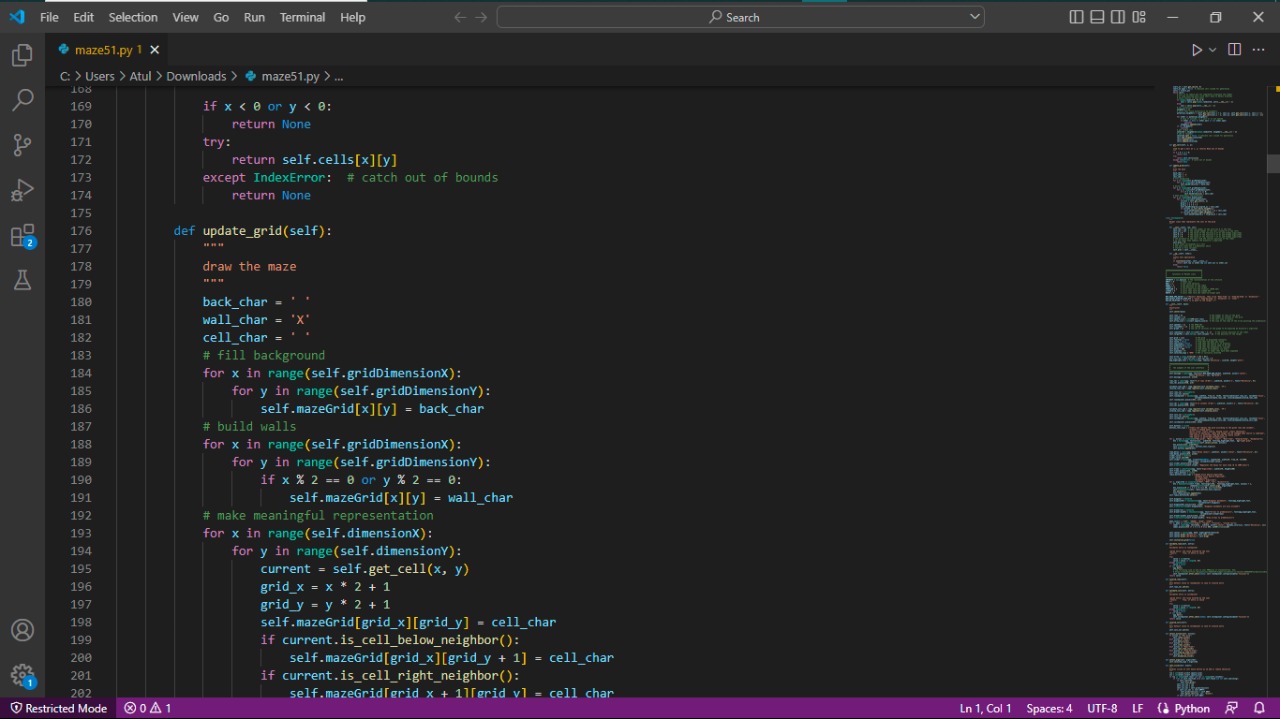
****

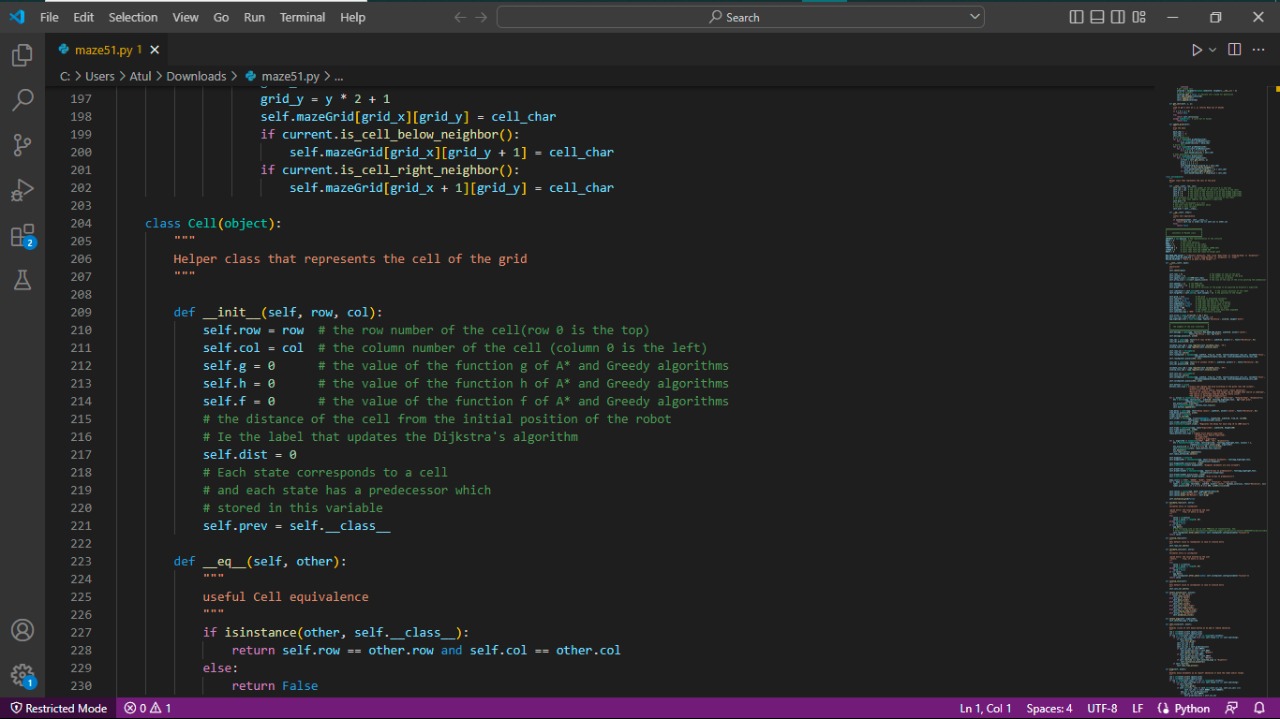
Fig.3.4(a) 

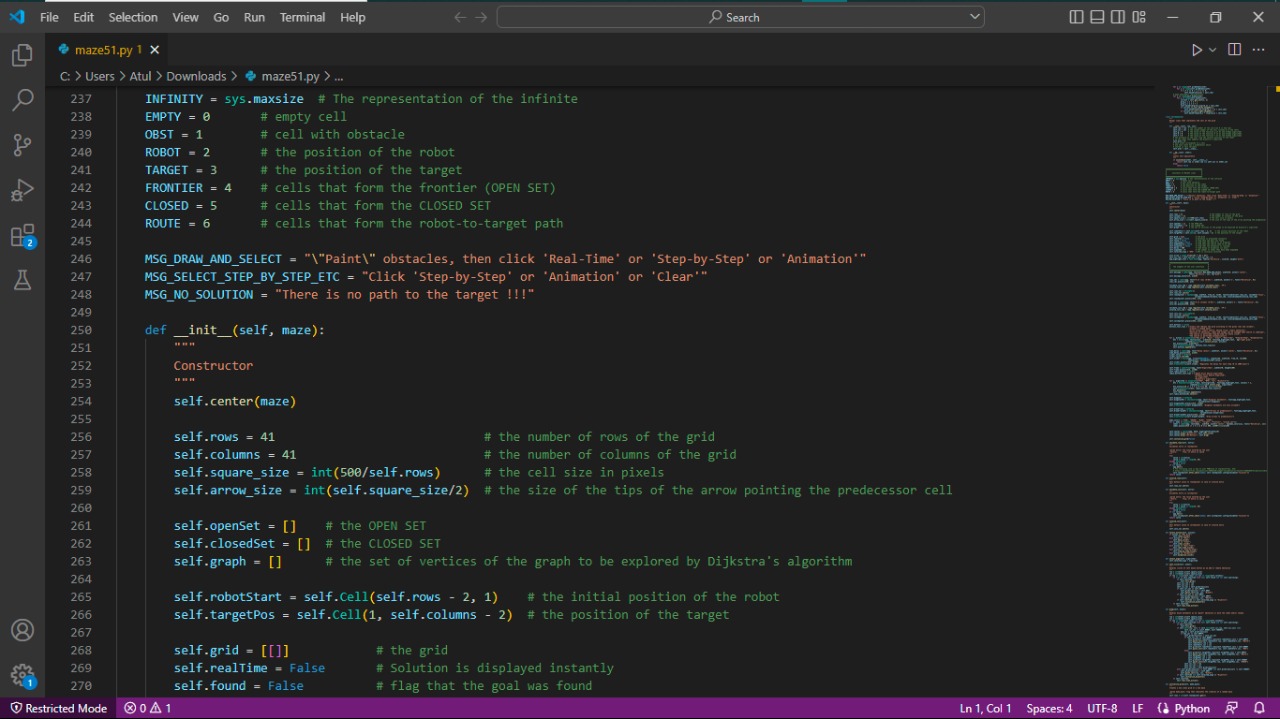




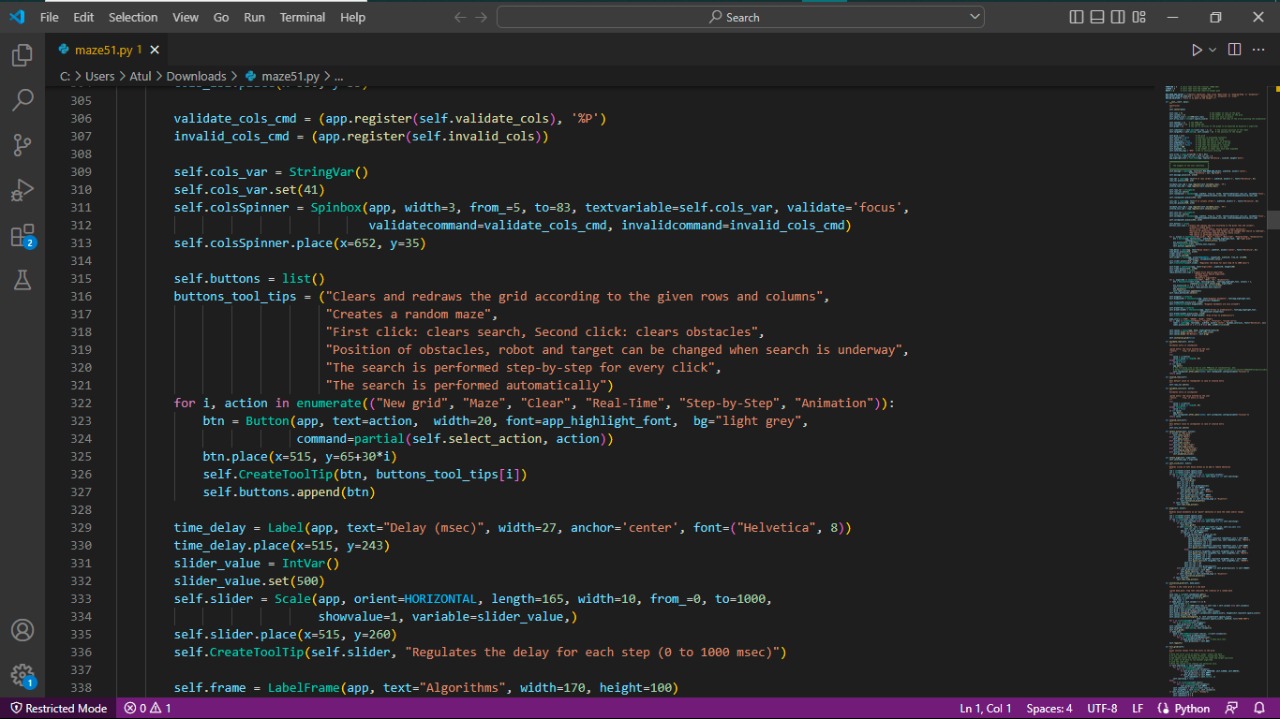


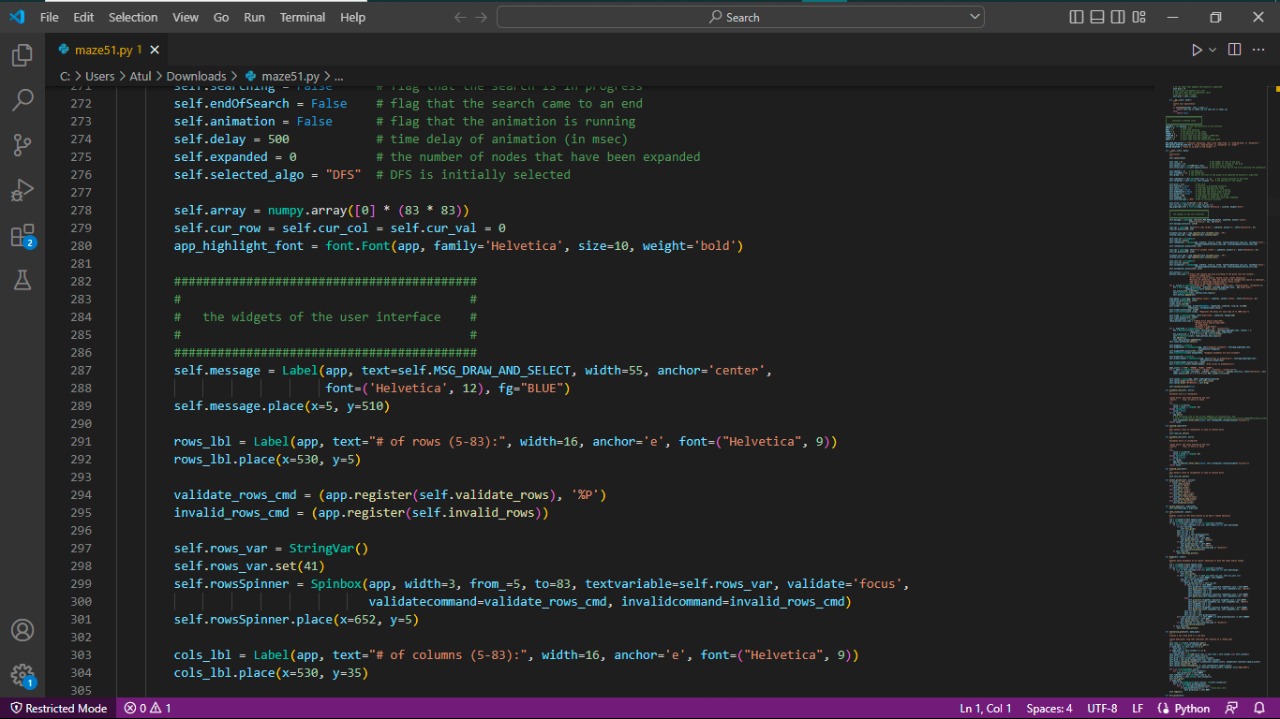


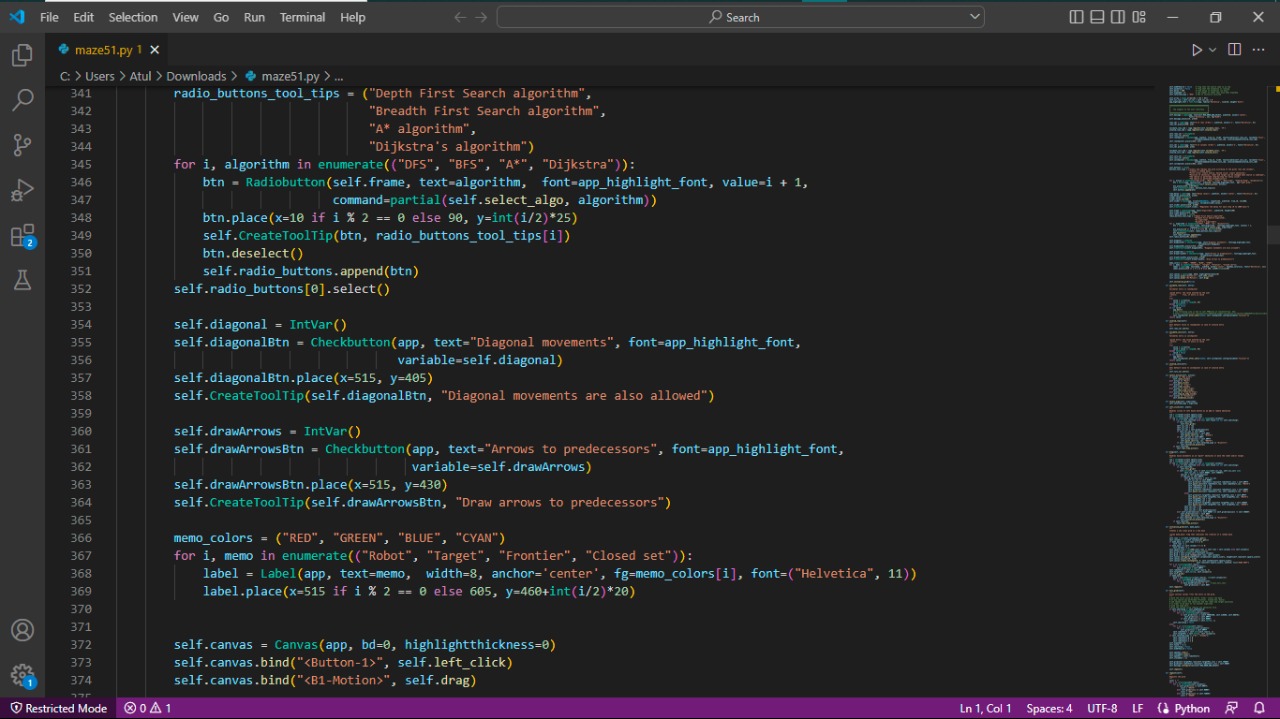


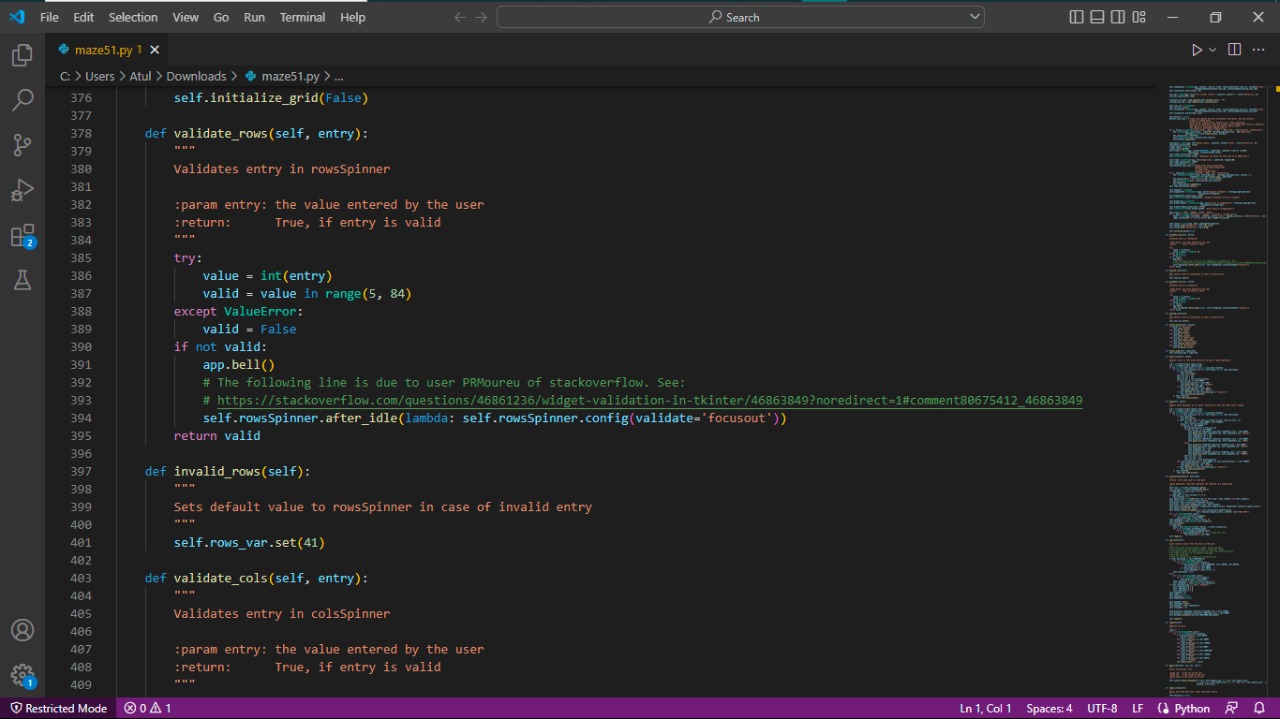


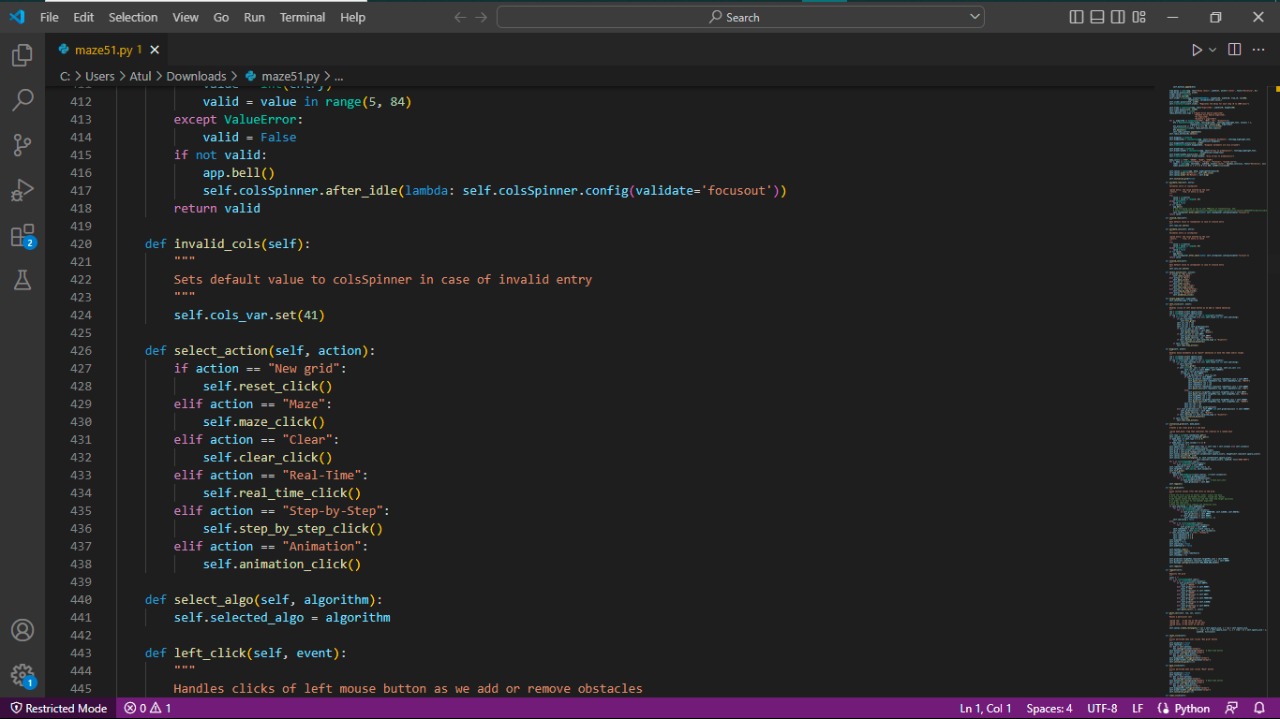


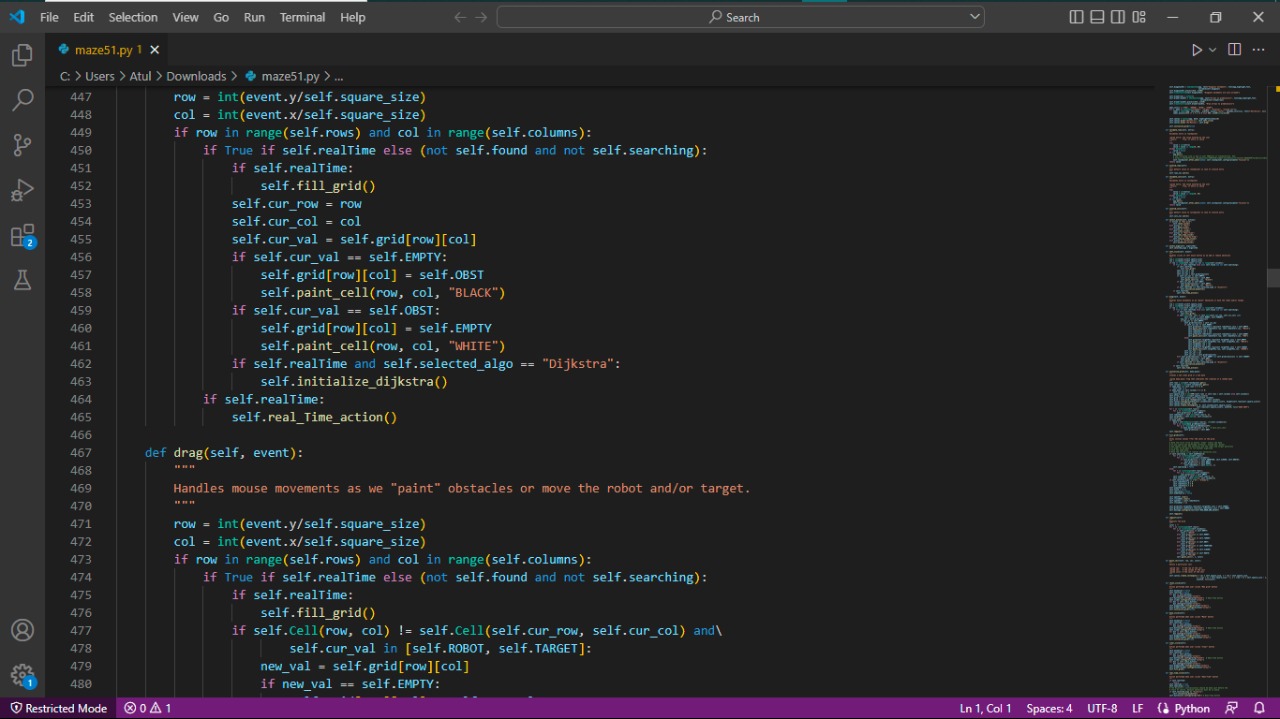


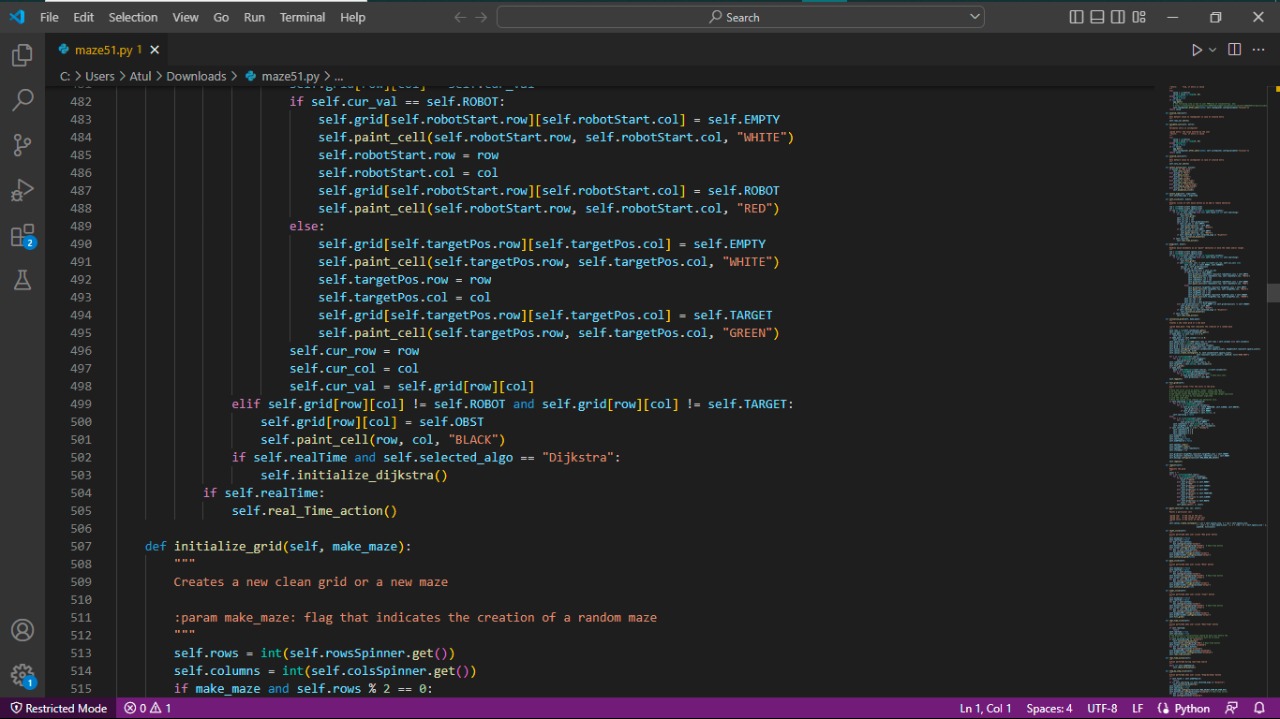


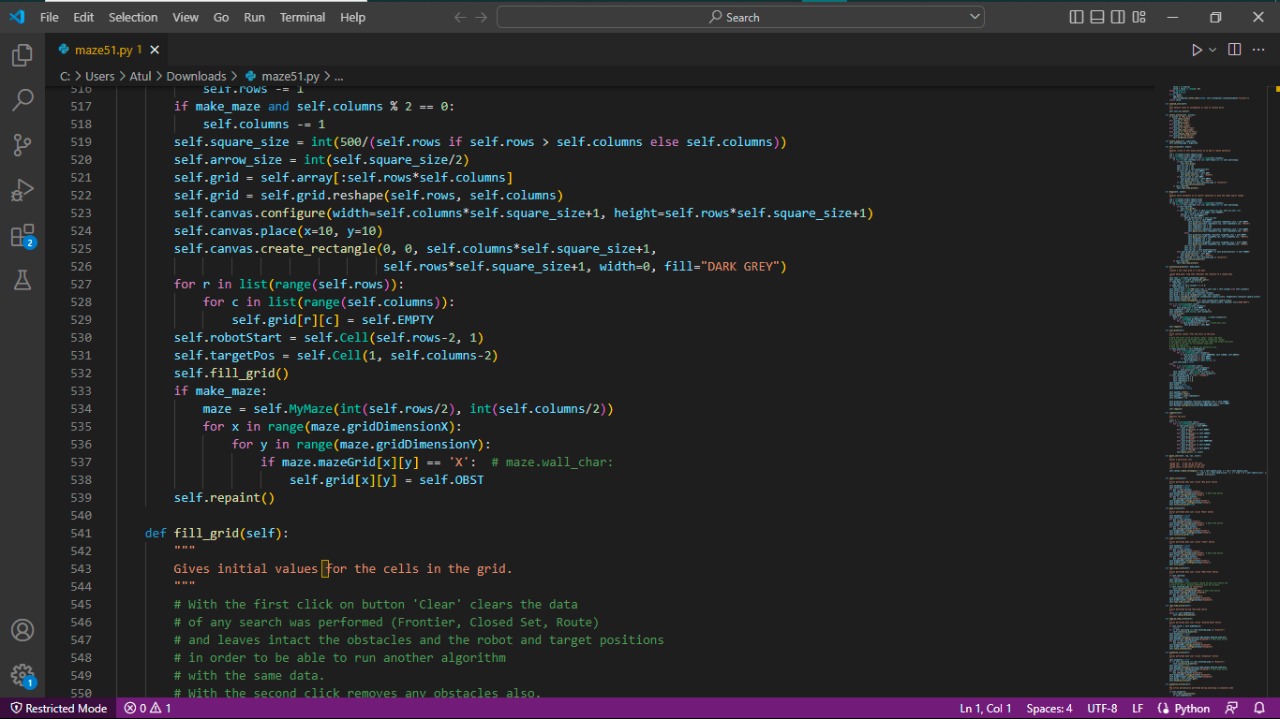


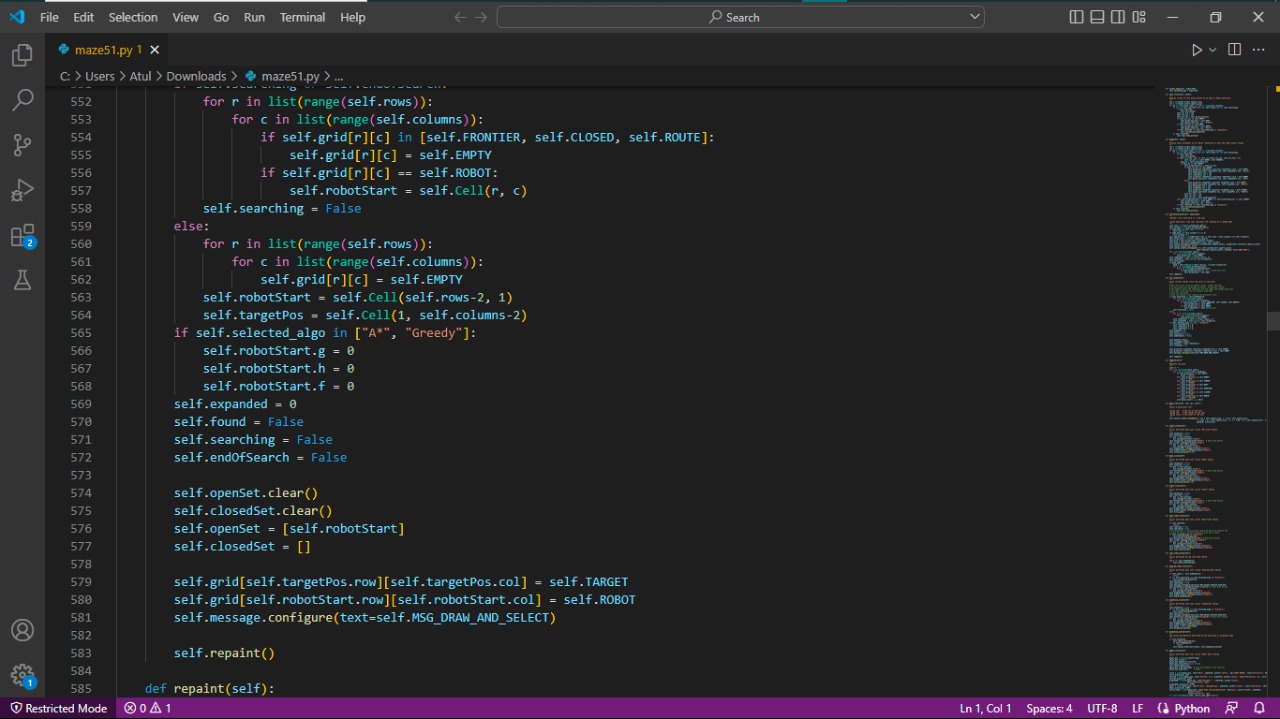


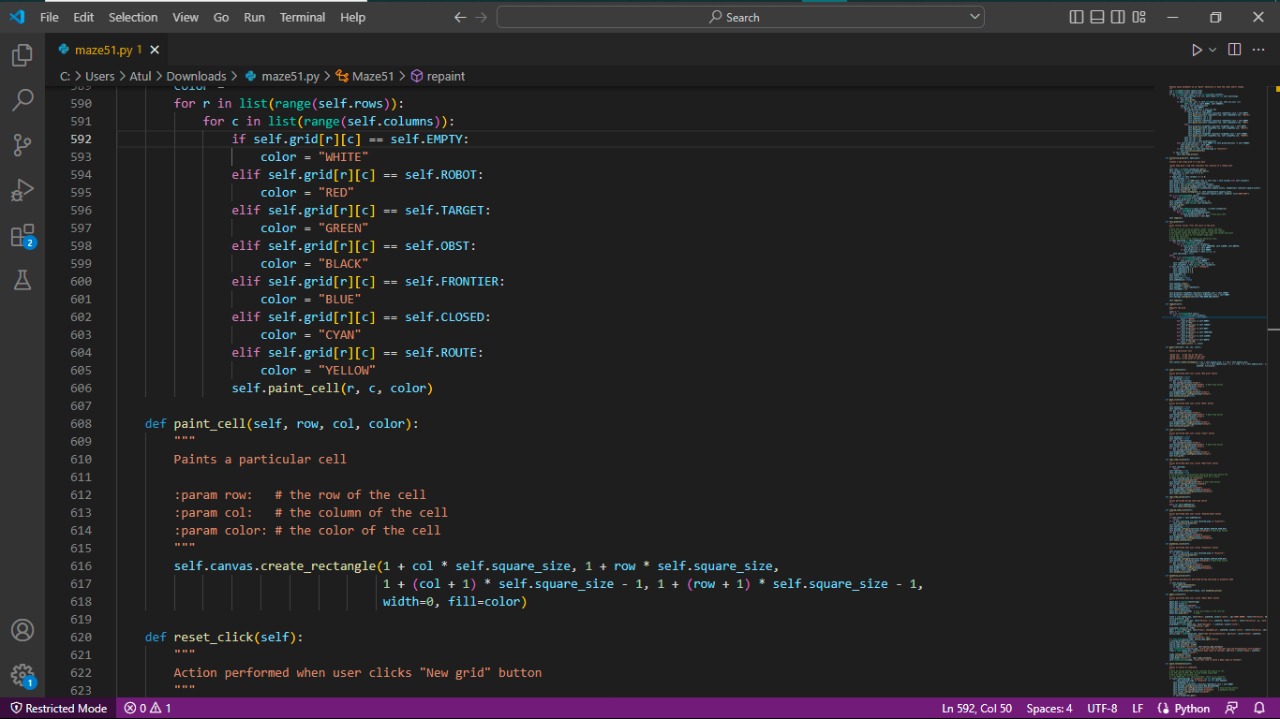


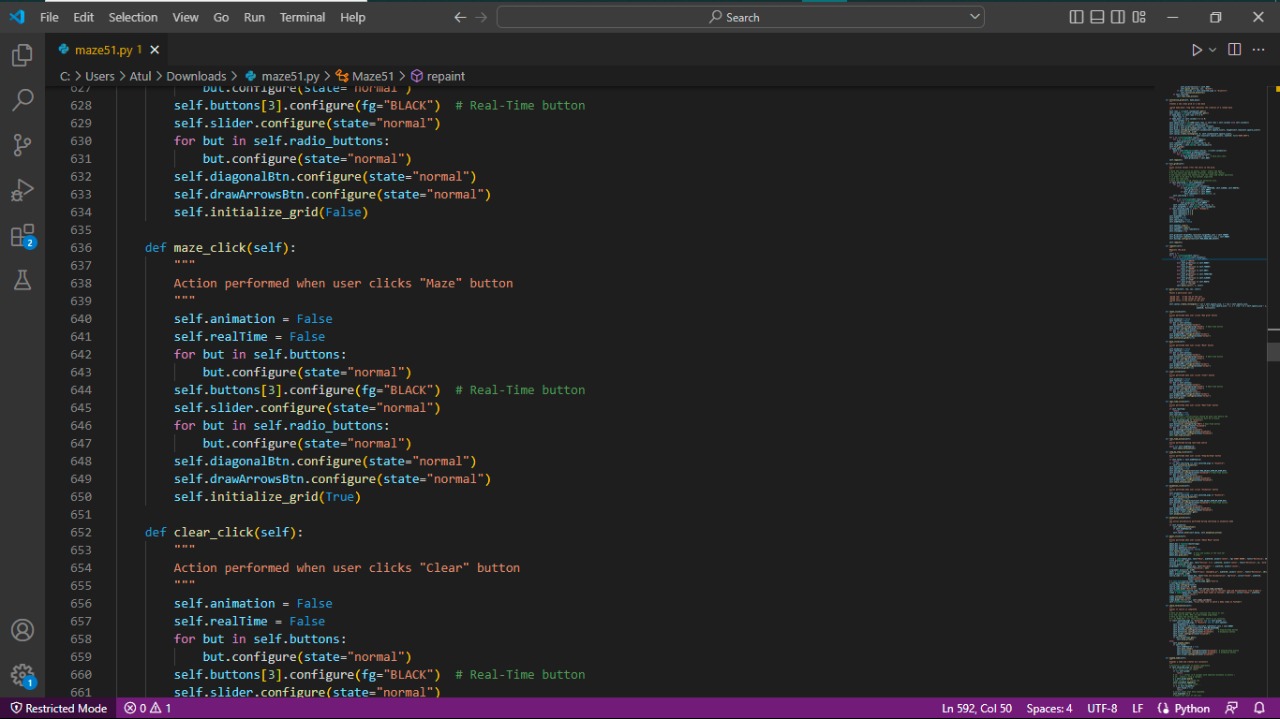


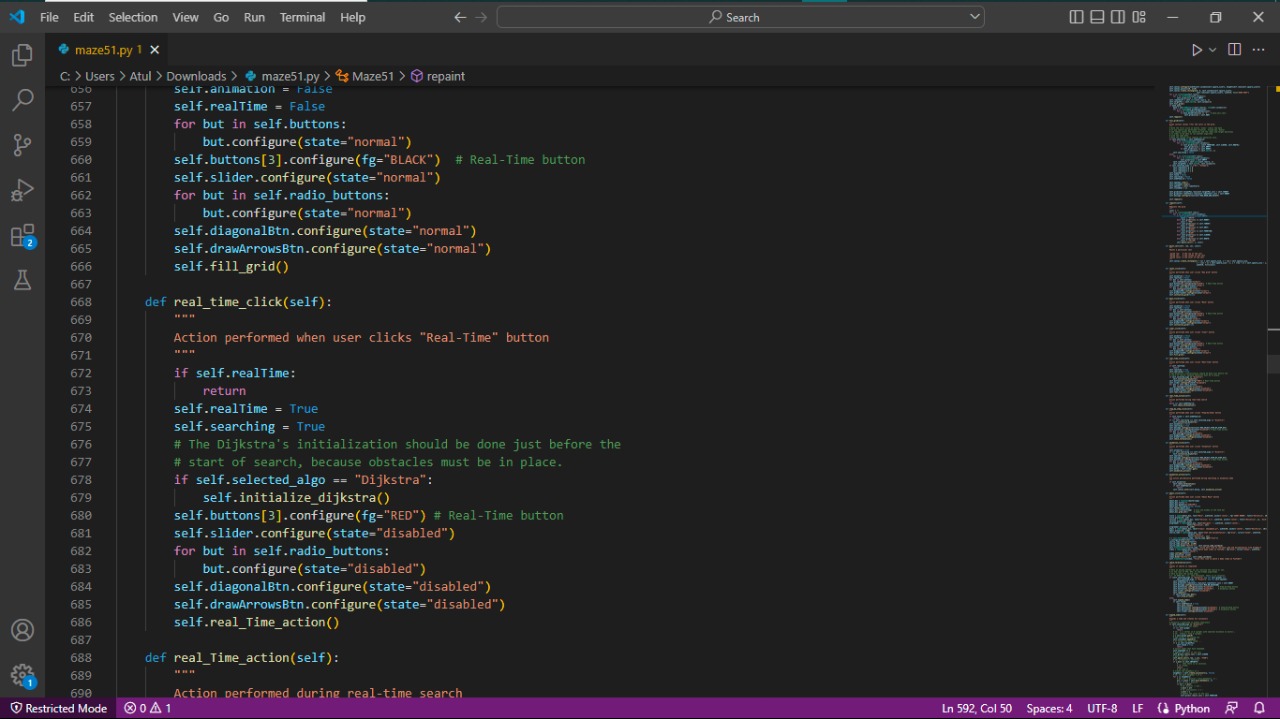


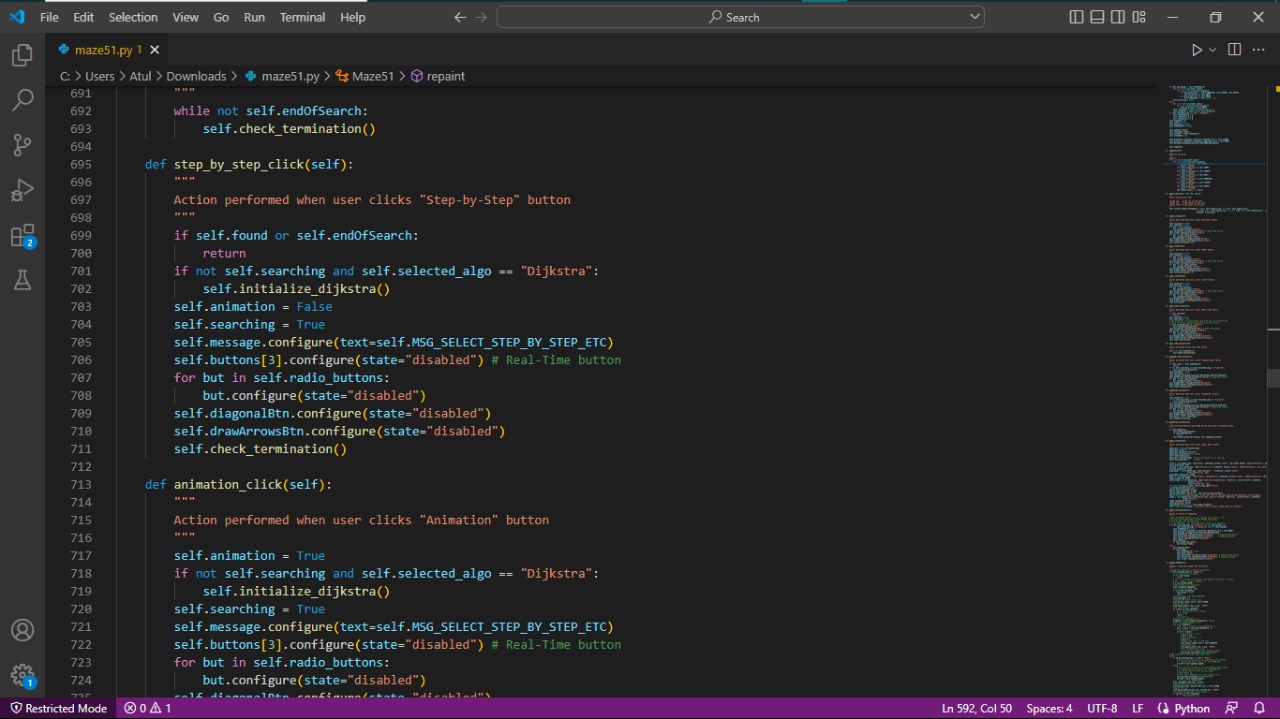


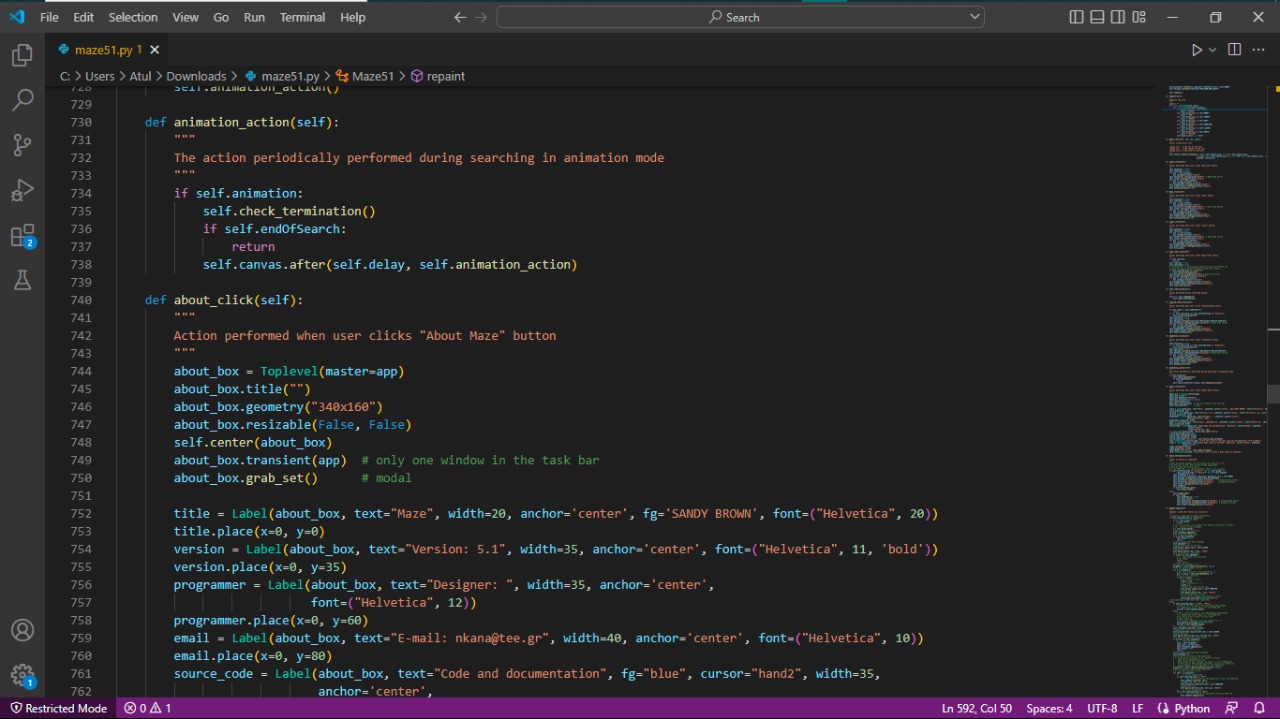


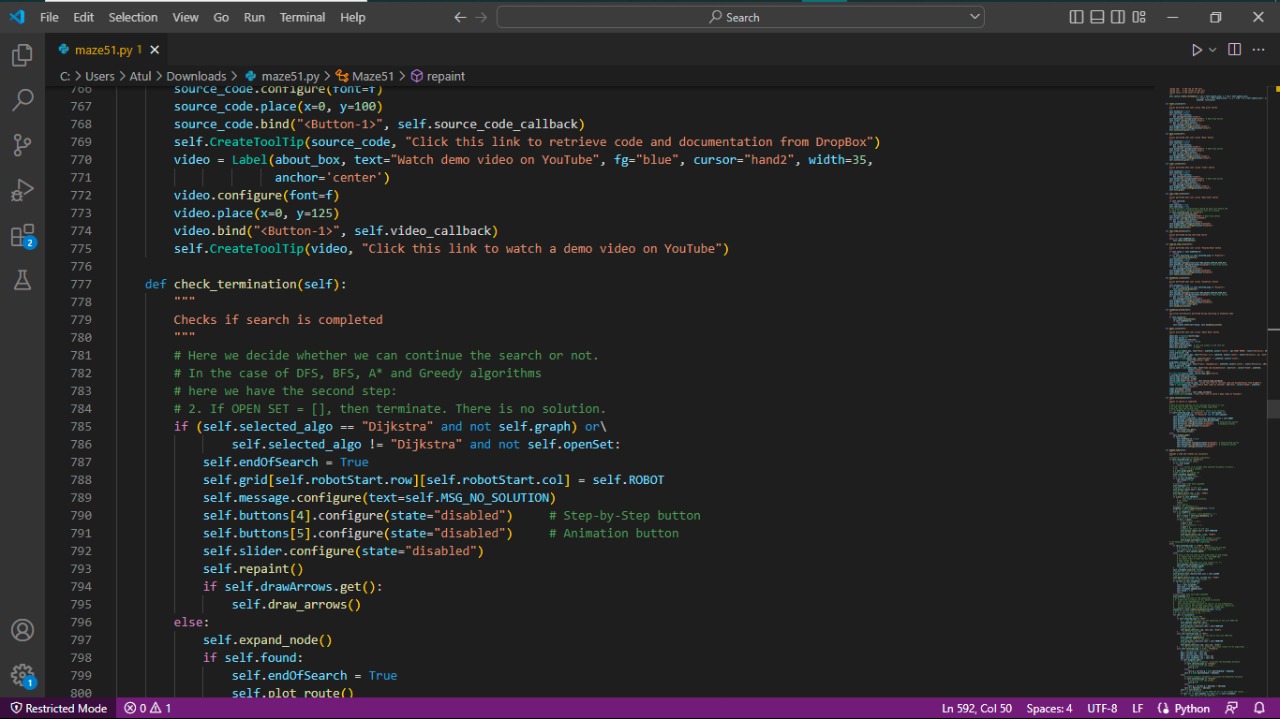


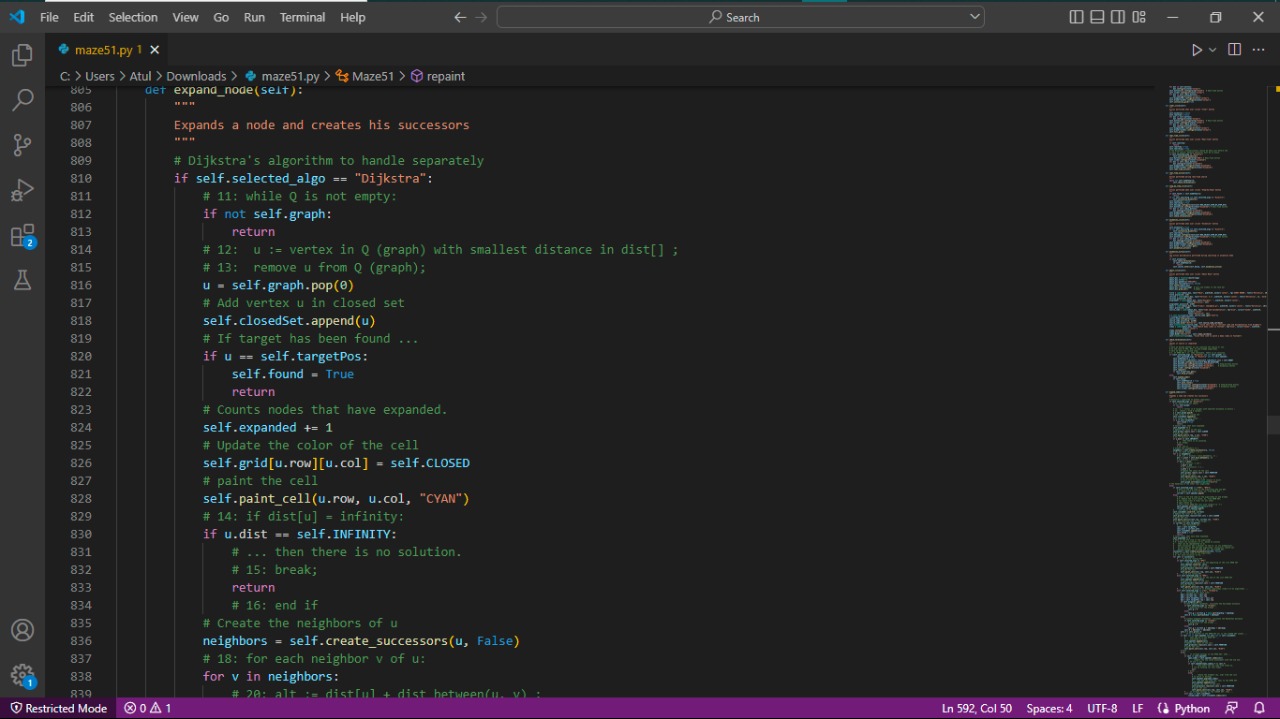


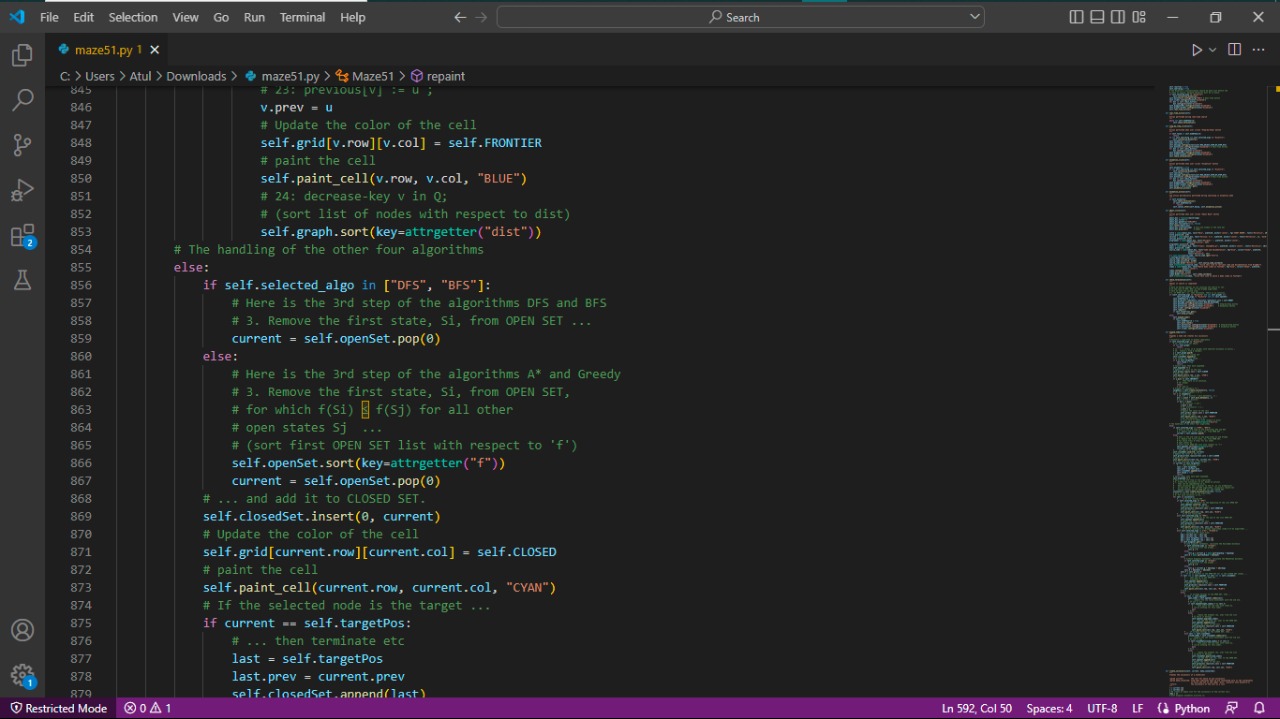




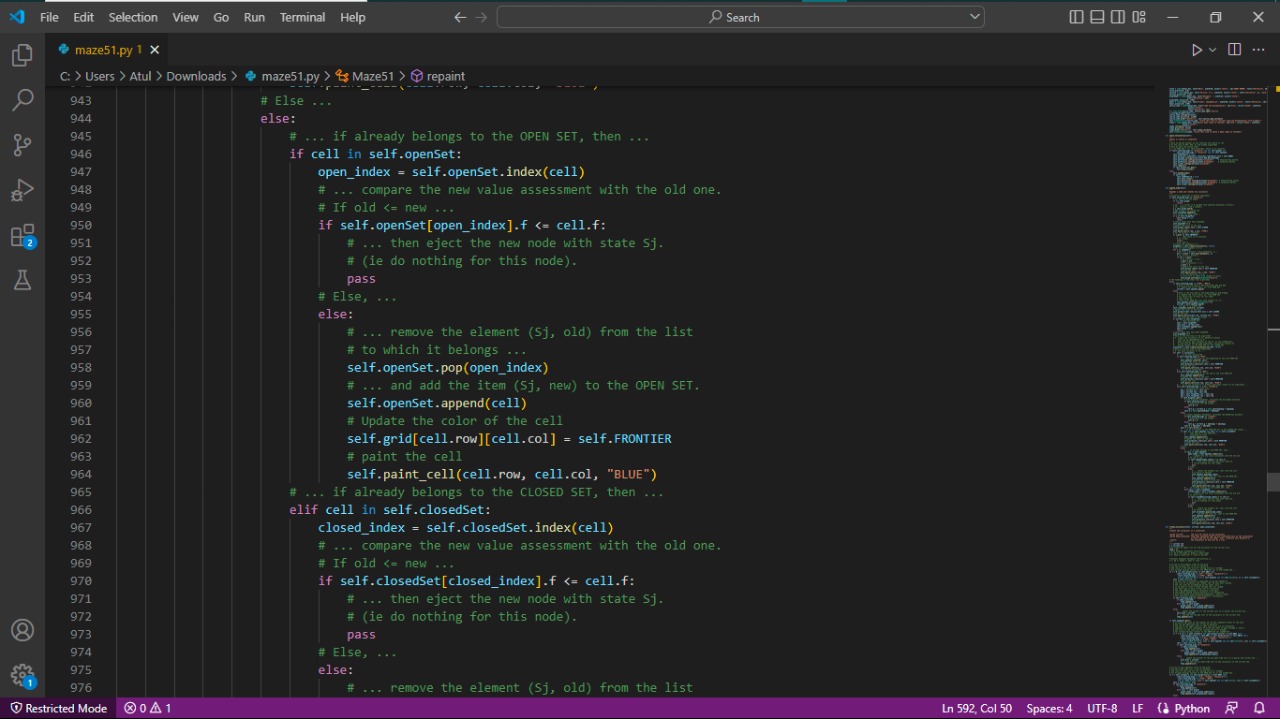


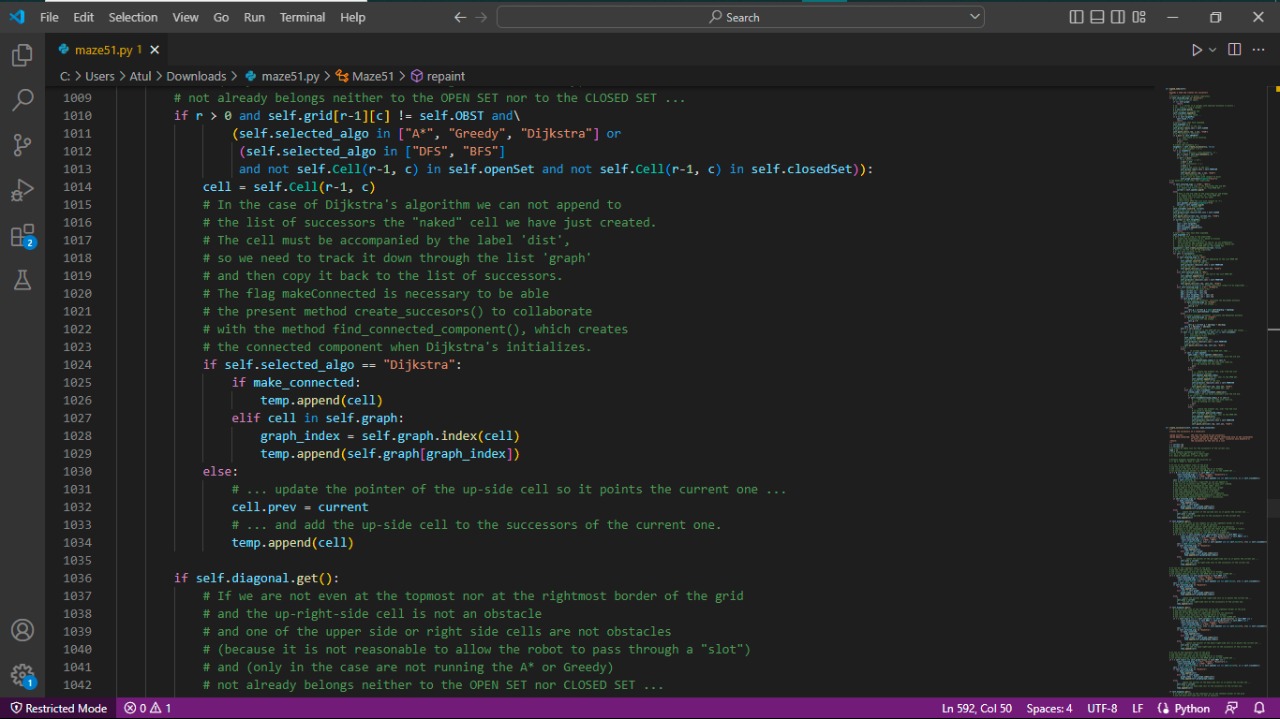


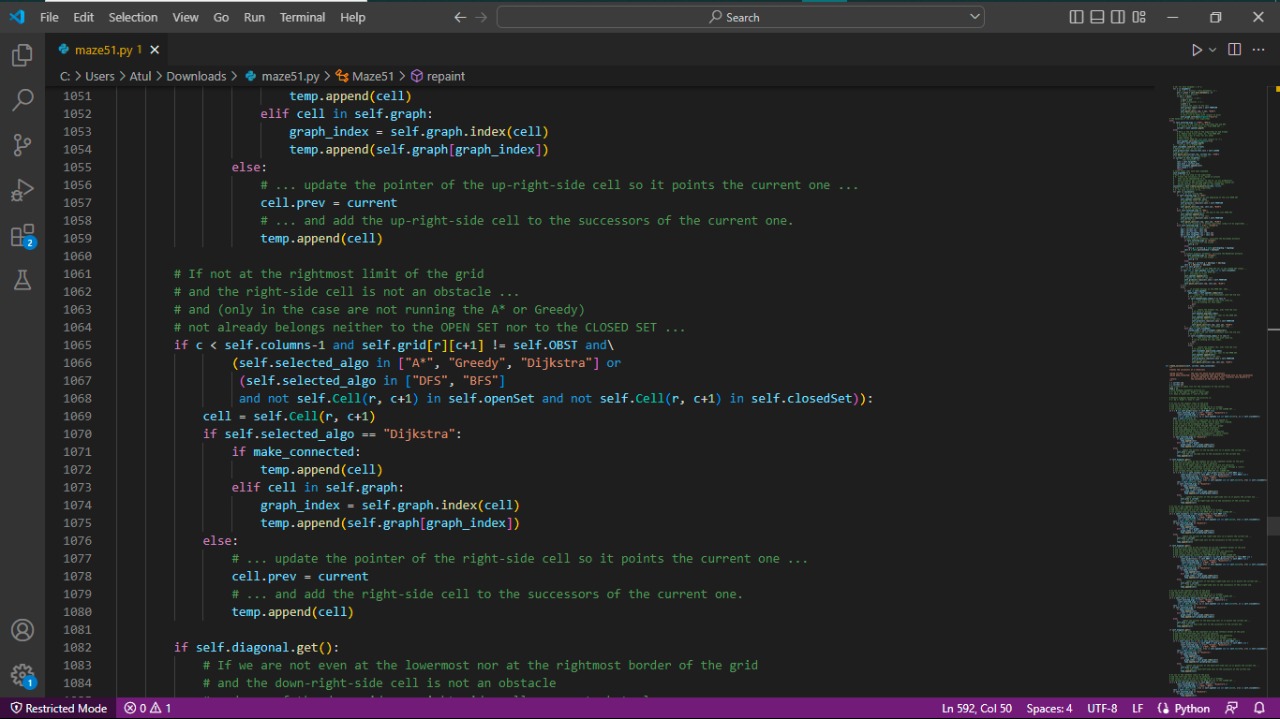












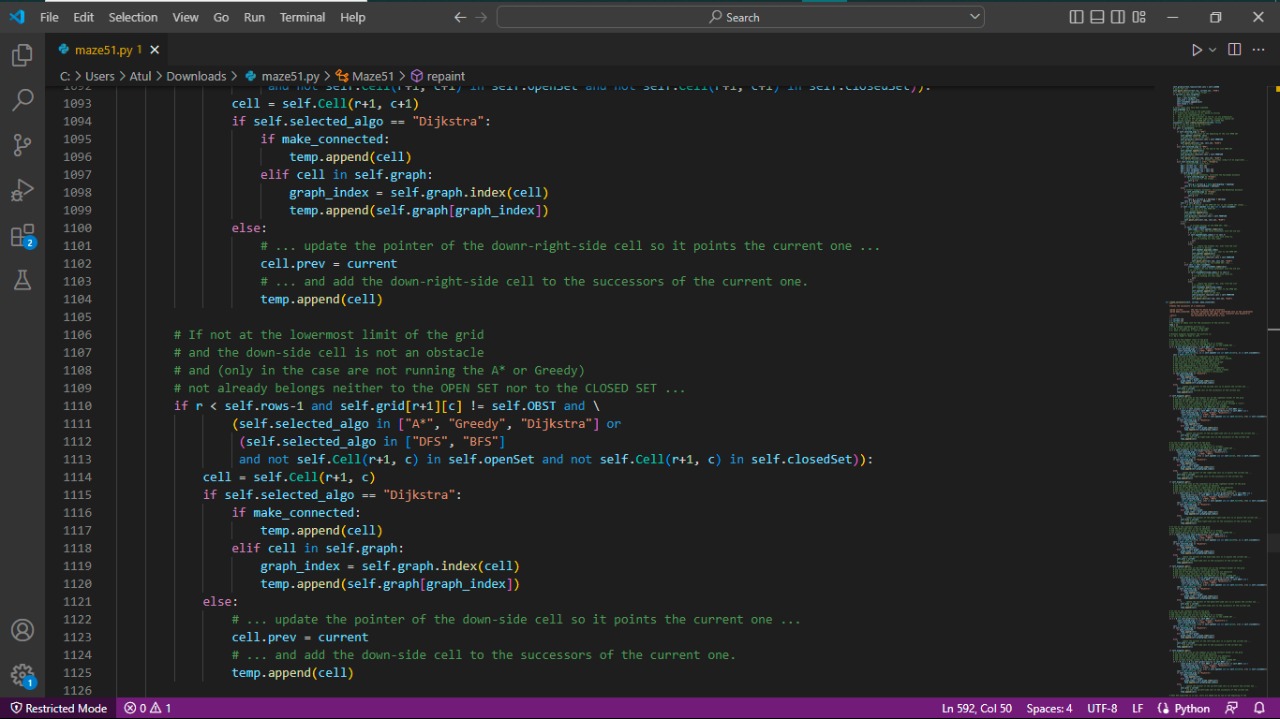


Fig 3.4.1(d)

# CHAPTER 4

# 

# RESULTS AND DISCUSSION



# 

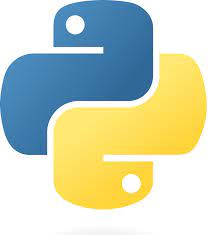
**4.1 Data Collection and Performance Metrics**

**Technologies Used**

* Programming language used: python
* Python Module: Tkinter.
* Software application on Desktop OS(Windows): command prompt/python maze
* Support application: VS code.

**Performance Metrics**

* Accuracy is about 95%
* Automated
* Can show different outputs based on the goal states and the obstacles
* Can show the animated and step performance for the algorithm.



*Fig.4.1.python*



**4.2. Comparison of Existing Solution**

* Many existing models have the comparision of the two algorithms only but the model which we have proposed will be having the most listened and most famous path finding algorithms for the maze .

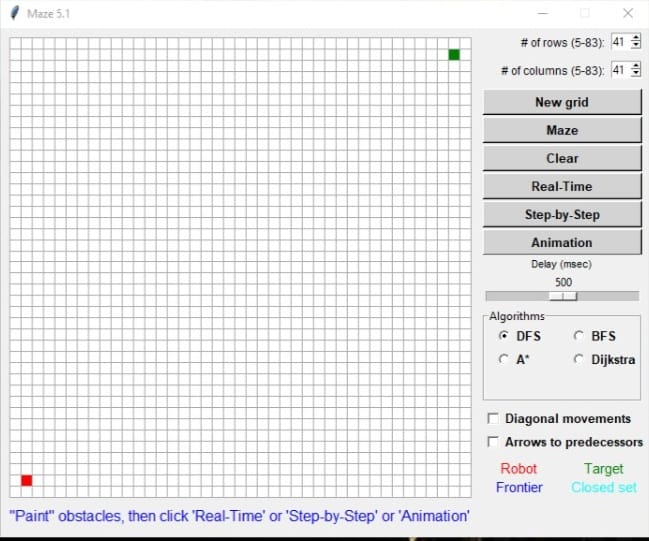
**Some other advantages of our proposed model are:**

* It can generate its own labyrinth.
* It can show the animated path of all three algorithms.
* It can show the step by step path for all three algorithms.
* We can increase or decrease the row and column of the grid which will eventually increase the path size and change the path location in the grid.
* It show the number of nodes traversed and steps taken to reach the goal state in each algorithm.

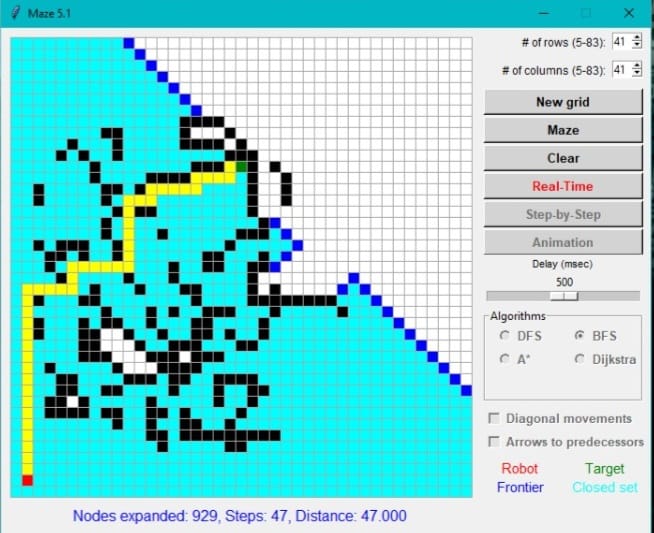


**4.3. Results and Screen Shots**

* When we run the code that will straight away lead to the out put of the maze which will allow us to run all the possibilities of the path finding algorithm

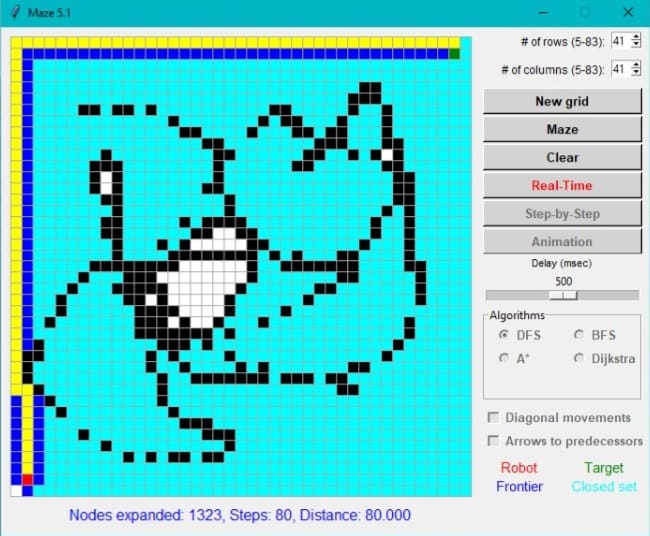


*Fig.4.3.maze*

****

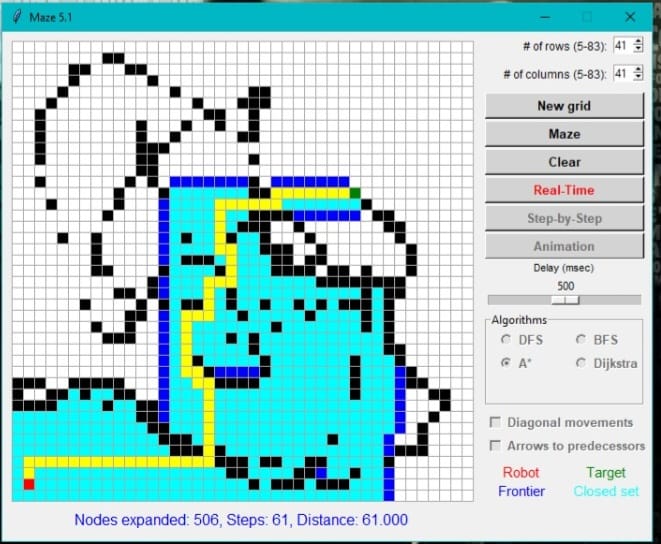


*Fig.4.3(a)BFS*



 *Fig.4.3(b)DFS*





*Fig.4.3(c) A\**





# CHAPTER 5

# CONCLUSION



### CONCLUSION:

Based on research we have done and the evaluation conducted , it can be concluded that the performance of A\* approach is better than Breath First Search(BFS) and Depth First Search (DFS) . the A\* algorithm finds the shortest path exploring less number of nodes than that of Breadth First Search and Depth First Search .In all we think that the accuracy of the model is decent and better than many already existing models and further can be improved by directly applying the game to it . the time taken by A\* is more than BFS and DFS because of the computational overhead caused due to heuristic function . In particular ,A\* algorithm is widely used for maze solving and other analogous problems as it uses straight line heuristic which cuts down the time by significant amount

Also it is more understandable for comparision of the algorithms

Our project would be effective to everyone to understand the difference between the algorithms and also understand the implementation of the algorithm

We would like to conclude that our project would be more time saving by taking the different mazes as the reference for the gaming purpose



# CHAPTER 6

# REFERENCES



## REFERENCES

1. https://www.geeksforgeeks.org/a-search-algorithm/
2. <https://en.wikipedia.org/wiki/Breadth-first_search>.
3. A review of various maze solving algorithms based on graph theory.
4. Bidirectional Graph Search Techniques for Finding Shortest Path in

Image Based Maze Problem

1. Pathfinding Algorithm Efficiency Analysis in 2D Grid” by Zarembo and Sergejs.
2. A Review of the Paper of Navigation and Pathfinding using Mobile Cellular Automata by Singhal and Harish.
3. [Artificial intelligence - Wikipedia](https://en.wikipedia.org/wiki/Artificial_intelligence)
4. Pathfinding Car Racing Game Using Dynamic Pathfinding Algorithm and Algorithm A\*
5. <https://www.sciencedirect.com/topics/mathematics/manhattan-distance>
6. N.Chandra Sekhar Reddy, Purna Chandra Rao Vemuri, A Govardhan, "evaluation of pca and k-means algorithm for efficient intrusion detection", international journal of applied engineering research, vol. 12, number 12 (2017), pp. 3370-3376, issn 0973-4562. [scopus].
7. P Devika, V Prashanthi, “RFID Based Theft Detection and vehicle monitoring system using cloud”, International Journal of Innovative technology and Exploring Engineering(IJITEE), Volume 8, Issue, Pages.737-739, ISSN:2278-3075, February 2019.(Scopus indexed)
8. N.Chandra Sekhar Reddy, Purna Chandra Rao Vemuri, A. Govardhan, "an implementation of novel feature subset selection algorithm for ids in mobile networks", international journal of advanced trends in computer science and engineering, vol. 8, number 5 (2019), pp. 2132-2141, issn 2278-3091. [scopus].