**PROJECT REPORT:**

MAZE RUNNER

PROJECT DESCRIPTION:

Labyrinths are regularly basic riddles for people; however, they present an incredible programming issue that we can settle utilizing the most limited way methods like Dijkstra's algorithm, Breadth First Algorithm, and Depth First Algorithm. Our goal is to create the shortest path which starts in the white and does not cross into the black boundaries.

However, we used Breadth First Search (BFS) to build our game.

OBJECTIVES:

1. Implementation of DSA techniques using BFS and backtracking.
2. Using data structures like dictionaries, lists, tuples, and queues.

LIBRARIES USED:

1. Tkinter
2. Turtle
3. Time
4. Threads

SCREENSHOTS OF OUTPUTS:

Graphical user interface, application, PowerPoint

Description automatically generated

Graphical user interface, qr code

Description automatically generated with medium confidence

Qr code

Description automatically generated

REFERNCES:

1. <https://www.makeschool.com/mediabook/oa/tutorials/trees-and-mazes/solving-the-maze/>
2. <https://www.hackerearth.com/practice/algorithms/graphs/breadth-first-search/tutorial/>
3. <https://stackoverflow.com/questions/58688299/python-maze-bfs-shortest-path>
4. <https://www.codespeedy.com/call-a-function-after-some-interval-in-python/>
5. <https://zetcode.com/tkinter/snake/>
6. <https://www.codegrepper.com/code-examples/python/python+game+over+screen>
7. <https://www.includehelp.com/algorithms/backtracking-types-and-algorithms.aspx>