**Term Project - Milestone 2**

**Selected Application:**

**Literature Review:**

***Application:***

***Maze Game Solving***

*Algorithms****:***

***Breadth First Search***

***Depth First Search***

***A\* Search***

***Important points of BFS***

* BFS works as queue data structure.
* Queue data structure is literally FIFO (First in First out).
* It traverses all nodes level by level.
* It explores a node then its child node on level # 2 and so on.
* BFS can be applied in a graph structure.
* It can be used to determine if path exists between two nodes in a graph.

***Important points of A\* search***

* It is informed search that uses heuristic to guide the search process towards the goal state.
* A\* search is combination of UCS and greedy search.
* It moves to goal by adding actual path cost (g (n)) and heuristic value (h (n)) which path has less addition value then it follows that path.
* It should be admissible. It means heuristic value should be less than the actual path cost.
* It gives optimal solution if heuristic function is admissible and consistent.
* A\* search finds shortest path from start state to goal state.

***Important points of Maze Game Solving***

1. **Representing the Maze**

The Maze is represented in a graph data structure, where each cell is a node and edges are paths between the cells.

1. **Start and Goal cell**

Identify the start cell and goal cell of the Maze. These cells are used as initial state and target state for search algorithm.

1. **Keeping Track of Visited cells**

It keeps track of all nodes. It means each node is visited only once. It keeps the track of all visited nodes in a dictionary.

1. **Stop Criteria**

Stop the search when goal is found or when all reachable cells have been visited.

**Note!!!**

By considering all above points, Maze game can be solved by BFS algorithm to find the shortest path from starting cell to goal cell.

***Maze Game Solving according to BFS Algorithm***

1. **BFS Algorithm usage**

BFS algorithm can be used to solve the maze game.

1. **Queue Data Structure**

Use a queue data structure to implement BFS. It holds the nodes that need to be visited in search process.

1. **Visiting Neighboring cells**

Visit all neighboring cells to reach the goal. This can be done by iterating over the adjacent cells

***Maze Game Solving according to A\* search Algorithm***

1. **Define goal**

First of all define the start cell and goal cell.

1. **Open and Close List**

Initialize the open and close list because open list contains the nodes that have been generated but not expanded yet and close list contains the nodes that have been expanded.

1. **Selecting path**

Select the path by adding actual path cost (g (n)) and heuristic value (h (n)), which path has less addition value, it will select that path.