**Maze Report**

**The used Data Structures:**

I used a Queue in BFS and Stack in DFS and I made a class which I called it Node , it consists of all what I want to know about each point in the maze , it contains ( Parent , x , y , Boolean visited and it’s value ).

**Algorithm :**

I used this Algorithm :

S = (Stack or Queue)

l = the start point;

S.add(l) ;

Visited = true;

While S is not empty

l = S.remove();

l.visited = true;

i f l is the end

end = true;

break ;

for each valid unvisited neighbour N of l

N.SetParent (l);

S.add(N) ;

End While

T = stack or queue;

G = end;

While g.value != Start

t.add(g);

g = g.parent;

end while

**how it works:**

I used Queue in BFS as it dequeues the first element entered and so on , so it enqueue all the unvisited cells around then dequeue each cell and enqueue around unvisited cells , In BFS we check first level then go to second level and so on till the End , BFS reaches the shortest path  
I used Stack in DFS as it pop the last element entered and so on, so it push all the unvisited cells around then pop a cell and push its neighbor unvisited cells and so on till this path reaches the end or doesn’t find a way to pass through , so its time for the second path to do this mission , In DFS we go through a path till its end and then return if the Target isn’t reached .

**Sample Runs**:

3 3

S..

.#.

E##

Sol.

BFS and DFS

{0,0},{1,0},{2,0}

5 5

S..#E

..#..

.##..

.....

#####

Sol.

BFS

{0,0},{1,0},{2,0},{3,0},{3,1},{3,2},{3,3},{2,3},{1,3},{1,4},{0,4}

DFS

{0,0},{0,1},{1,1},{1,0},{2,0},{3,0},{3,1},{3,2},{3,3},{2,3},{1,3},{1,4},{0,4}