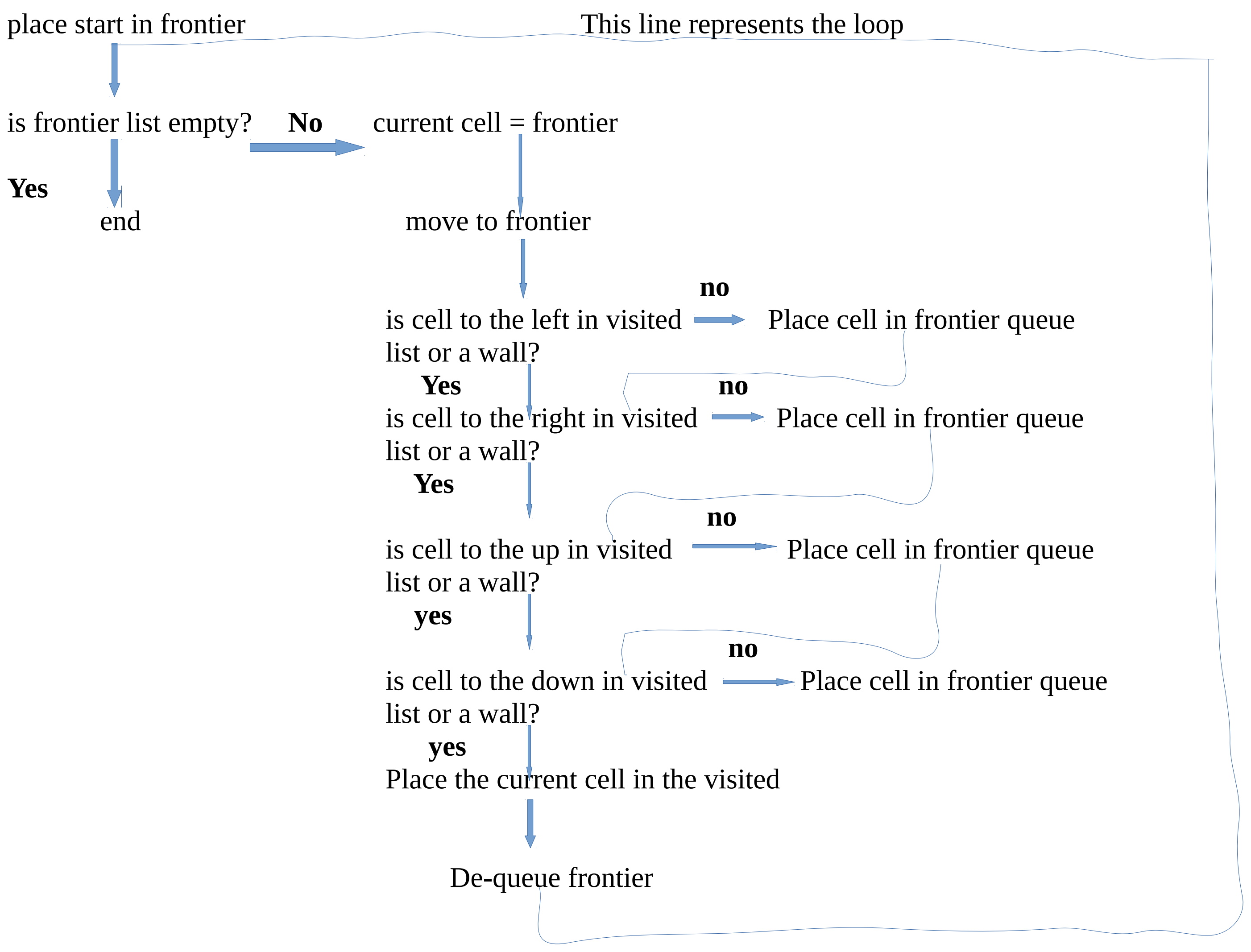
MAZE SOLVER

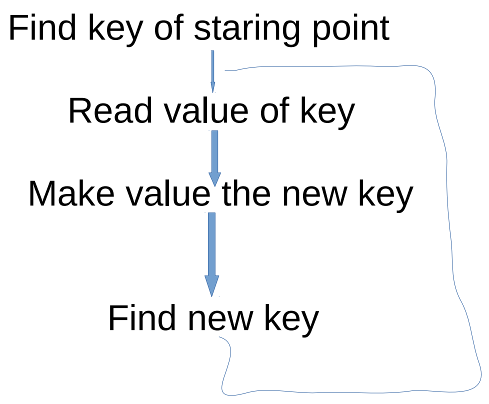
Bfs implementation:


place start in frontier
This line represents the loop
is frontier list empty?
No
current cell = frontier
Yes
end
move to frontier
no
is cell to the left in visited list or a wall?
Place cell in frontier queue
Yes
no
is cell to the right in visited list or a wall?
Place cell in frontier queue
Yes
no
is cell to the up in visited list or a wall?
Place cell in frontier queue
yes
no
is cell to the down in visited list or a wall?
Place cell in frontier queue
yes
Place the current cell in the visited
De-queue frontier

BFS: Breadth-first search (BFS) is an algorithm that is used to graph data or searching tree or traversing structures. The full form of BFS is the Breadth-first search.The algorithm efficiently visits and marks all the key nodes in a graph in an accurate breadthwise fashion. This algorithm selects a single node (initial or source point) in a graph and then visits all the nodes adjacent to the selected node. Remember, BFS accesses these nodes one by one. Once the algorithm visits and marks the starting node, then it moves towards the nearest unvisited nodes and analyses them. Once visited, all nodes are marked. These iterations continue until all the nodes of the graph have been successfully visited and marked.

Bfs implementation explanation: There are two lists frontier list and visited list and a variable called current.So at the beginning of the algo the starting point in the maze is put in the frontier list next frontier list is empty or not is checked if it is not empty than the current value is set to the frontier value. After that control goes to the frontier.Next it is checked if the cell to left of the frontier is already present in the visited list or is it a wall if it is a wall control goes to the step below it. It is checked if the cell to the right is a wall or is present in the visited list if it is not a wall and not present in the visited list then the cell is placed inside the frontier queue.Same thing will be repeated for both up and down. Then the current cell value is placed in the visited list. After that the frontier is dequed that is it removes the first entry which is achieved in the code by using popleft() as frontier list follows FIFO.Then the control moves back around the loop and starts the whole process from (is frontier list empty?). The loop exists when the frontier list is empty.

Backtracking: This is done using the python dictionary which is a key value pair. Backtracking is a algorithm.



Solution = {key:value}

This is used to identify the cell that i am on and the cell that i came from.The starting point is the place from where i want to backtrack from to the beginning cell from where bfs started in the beginning. After that the value of the key is read and the value is set as the new key. Then the value of the new key is figured out.

Turtle Graphics: This is used to generate the graphical interphase of the maze.The maze setup and the animation are using the turtle graphics library.The maze is made up of the ascii characters. Cross is a wall, (s) it is the staring point, (e) it is the end point. Turtle graphics uses a grid system so each cell will be having a x and a y parameter.