State Space Maze Search

Firstly, the main method will be called.

Our motive here will be to read the maze in txt file and convert it into a dict. We have used ‘start’ for the starting point, ‘goal’ for the food/end point, width and height for the size of maze, ‘hasMoreLine’ to detect the last line/row of the maze and stop loop.

```
CharArr = [str(i) for i in file.readline().strip()]
    width = len(CharArr) if width == 0 else width
    if len(CharArr)==0:
      hasMoreLine = False
```

Here we are reading each line and storing in the char array

```
for k in range(len(CharArr)):
      dict[(k, height)] = CharArr[k]
      if(CharArr[k] == 'P'):
        start = (k, height)
      elif(CharArr[k] == '.'):
        goal = (k, height)
```

Here ‘k’ depicting the ‘X’ coordinate and ‘height’ being the ‘Y’ coordinate.

if we detect ‘P’ storing it as start point, ‘.’ As goal point.

This will loop till the length of char array that is width of the maze and then move to the next row once height +=1.

Once the last line is reached, in upcoming loop hasMoreLine = False which will detect the end of the maze.

```
class Node:
  # Constructor
  def __init__(self, pos:(), parent:()):
    self.pos = pos
    self.parent = parent
```

Above used for storing the current node position and parent node position

```
    # Node Comaprision
    def __eq__(self, other):
        return self.pos == other.pos
```

Above used for node comparison. For eg. if currentNode == goalNode:

DFS / BFS Logic :

```
nonVisited = []
    visited = []
```

Here creating 2 stacks for visited and not visited nodes.

And then adding the start node to nonvisited stack which will make len > 0 and activate the while loop.

`currentNode = nonVisited.pop(-1)` → Last in first out. (Stack) The top value in stack will pop

`currentNode = nonVisited.pop(0)` → First in First out. (Queue) The bottom value will pop

```
closestNodes = [(a-1, b) , (a, b+1) , (a+1, b) , (a, b-1) ]
                 Left       Top       Right     Bottom
```

writeChar and draw function is used to draw the final path using + for path nodes and else print default value associated with position.