LAB #06

ⓐ Implement A* Search Algorithm
ⓑ Implement Hill climbing Algorithm for 8 Queens

ⓐ A* algo {shortest path from initial point to a goal}.

'Initialize a starting point assuming to be "1"

'Also assuming endgoal to be "4"

'Refering a set of unvisited nodes. So pick the nodes on the basis of priority queues, target which seems the closest

'based on f(score) of the node.

f(score) = g(score) + h(score)

g(score): actual distn travelled so far

h(score): Estimated distance left to reach the end goal

'Look at neighbour, & keep on updating path.

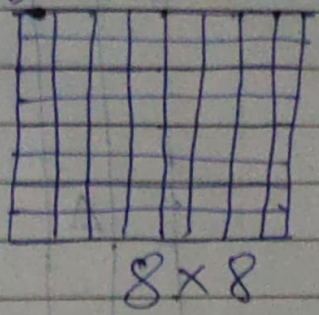
'Add the neighbour to the list if it hasn't been visited with a lower score before.

'Then track back the nodes travelled to obtain the path.

Heuristic(h) = No. of queens pair attacking each other.
g) = No. of queens placed so far.

Date 1/1
Page 35

goal: placing all queens on the board such that they do not attack each other.




```
def calculate_attacking_pairs(board):
    attacks = 0
    n = len(board)
    for i in range(n):
        for j in range(i+1, n):
            if board[i] == board[j] or
                abs(board[i] - board[j])
                == abs(i-j):
                attacks += 1
    return attacks

def a_star_8_queens(n=8):

    open_set = PriorityQueue()
    open_set.put((0, []))

    for col in range(n):
        new_board = board + [col]
        if len(new_board) <= n:
            g_score = len(new_board)
            h_score = calculate_attacking_
                pairs(new_board)
            f_score = g_score + h_score
```

So, now