Subject Name: Artificial Laboratory

subject code: U19CS505

EXAM: Mid-semester Practical examinations - AUG/SEP 2021

Student name: T.D.Sarvesh

ROLL NO: 19CS142

class: III - C

Date: 26-8-2021 AN.

Aim:

To apply depth first algorithm to solve 8 queen problem.

Procedure:

For solving n queens problem, we will try placing queen into different positions of one row. No queens should be placed horizontally, vertically or diagonally. In initial stage 0 queens on the board. In successor function add a queen in any square. In goal test 8 queens on the board, none attacked.

Coding:

```
Class NQueens:
    def __init__(self,size):
        self.size =size
        self.solutions=0
        self.solve()
    def solve(self):
        Positions =[-1]* self.size
        self.Put_queen(Positions,0)
        Print("Found", self.solutions,"solutions.")
    def Put_queen(self,Positions,target_row):
        if target_row == self.size:
            self.show_full_board(Positions)
            self.solutions+=1
        else:
            for coloumn in range(self.size):
                if self.check_Place(Positions,target_row,coloumn):
                    Positions[target_row]= coloumn
                    self.Put_queen(Positions,target_row+1)
    def check_Place(self,Positions,occupied_rows,coloumn):
        for i in range(occupied_rows):
            if positions[i] == coloumn or\
                positions[i]-1 == coloumn-occupied_rows or\
                positions[i]+1 == coloumn+occupied_rows:
```

```
                return False
            return True
    def show_full_board(self, positions):
        for row in range(self.size):
            line = ""
            for column in range(self.size):
                if positions[row] == column:
                    line += 'Q'
                else:
                    line += "."
            Print(line)
        Print("\n")
def main():
    NQueens(8)
if __name__ == "__main__":
    main()
```

Result:

Thus the above code and solution has been executed successfully.

Untitled3.ipynb - Colaboratory

colab.research.google.com/drive/159uUyC5brDzFJZibBwN1_EJQrvVnYTI9

Apps | SARVESH T.D - A4 | Dashboard | Hacker... | Reading list

+ Code + Text

RAM Disk

```
class NQueens:
    def __init__(self, size):
        # Store the puzzle (problem) size and the number of valid solutions
        self.size = size
        self.solutions = 0
        self.solve()

    def solve(self):
        """Solve the n queens puzzle and print the number of solutions"""
        positions = [-1] * self.size
        self.put_queen(positions, 0)
        print("Found", self.solutions, "solutions.")

    def put_queen(self, positions, target_row):
        """
        Try to place a queen on target_row by checking all N possible cases.
        If a valid place is found the function calls itself trying to place a queen
       on the next row until all N queens are placed on the NxN board.
        """
        # Base (stop) case - all N rows are occupied
        if target_row == self.size:
            self.show_full_board(positions)
            self.solutions += 1
        else:
            # For all N columns positions try to place a queen
            for column in range(self.size):
                # Reject all invalid positions
                if self.check_place(positions, target_row, column):
                    positions[target_row] = column
                    self.put_queen(positions, target_row + 1)

    def check_place(self, positions, ocuppied_rows, column):
        """
```

Type here to search | 34°C | ENG | 16:32 | 26-08-2021

```
        for i in range(ocuppied_rows):
            if positions[i] == column or \
                positions[i] - i == column - ocuppied_rows or \
                positions[i] + i == column + ocuppied_rows:
                return False
        return True

    def show_full_board(self, positions):
        """Show the full NxN board"""
        for row in range(self.size):
            line = ""
            for column in range(self.size):
                if positions[row] == column:
                    line += "Q "
                else:
                    line += ". "
            print(line)
        print("\n")

def main():
    """Initialize and solve the n queens puzzle"""
    NQueens(8)

if __name__ == "__main__":
    # execute only if run as a script
    main()
```

```
Q . . . . . . .
. . . . Q . . .
. . . . . . . Q
. . . . . Q . .
. . Q . . . . .
. . . . . . Q .
. Q . . . . . .
```

Untitled3.ipynb - Colaboratory

colab.research.google.com/drive/159uUyC5brDzFJZibBwN1_EJQrvVnYTI9

Apps | SARVESH T.D - A4 | Dashboard | Hacker... | Reading list

+ Code + Text

RAM Disk

```
. . . Q . . . .
Q . . . . . .
. . . . . Q .
. . . . Q . .
. . Q . . . .
. . . . . Q .


. . . . . . Q
. Q . . . . .
. . . . Q . .
. . Q . . . .
Q . . . . . .
. . . . . Q .
. . . Q . . .
. . . . . Q .


. . . . . . Q
. . Q . . . .
Q . . . . . .
. . . . . Q .
. Q . . . . .
. . . . Q . .
. . . . . Q .
. . . Q . . .


. . . . . . Q
. . . Q . . .
Q . . . . . .
. . Q . . . .
. . . . . Q .
. Q . . . . .
. . . . . Q .
. . . . Q . .


Found 92 solutions.
```

Type here to search | 34°C | ENG | 16:31 | 26-08-2021