In the Name of God

Sharif University of Technology - Department of Computer Engineering

Artificial Intelligence - Mr. Samiei

Spring 2023

Practical Assignment(Constraint Satisfaction Problem)

Deadline: Esfand 19th - 23:59
Cheating is Strongly Prohibited

Please run all the cells.

Personal Info

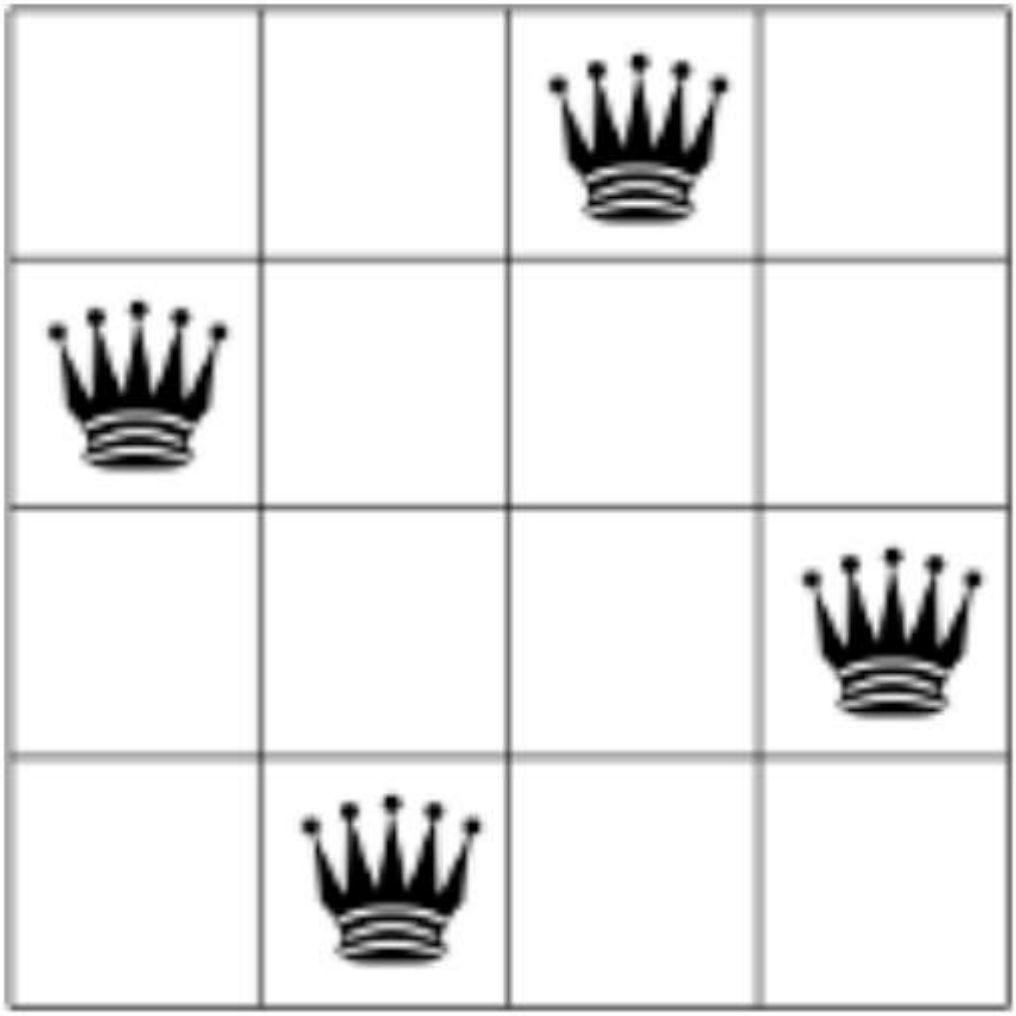
Student_number = 99105678 Name = Kiarash Last_Name =Kianian

CSP (Constraint Satisfaction Problem) (70 Points)

Author: Erfan Sadraiye

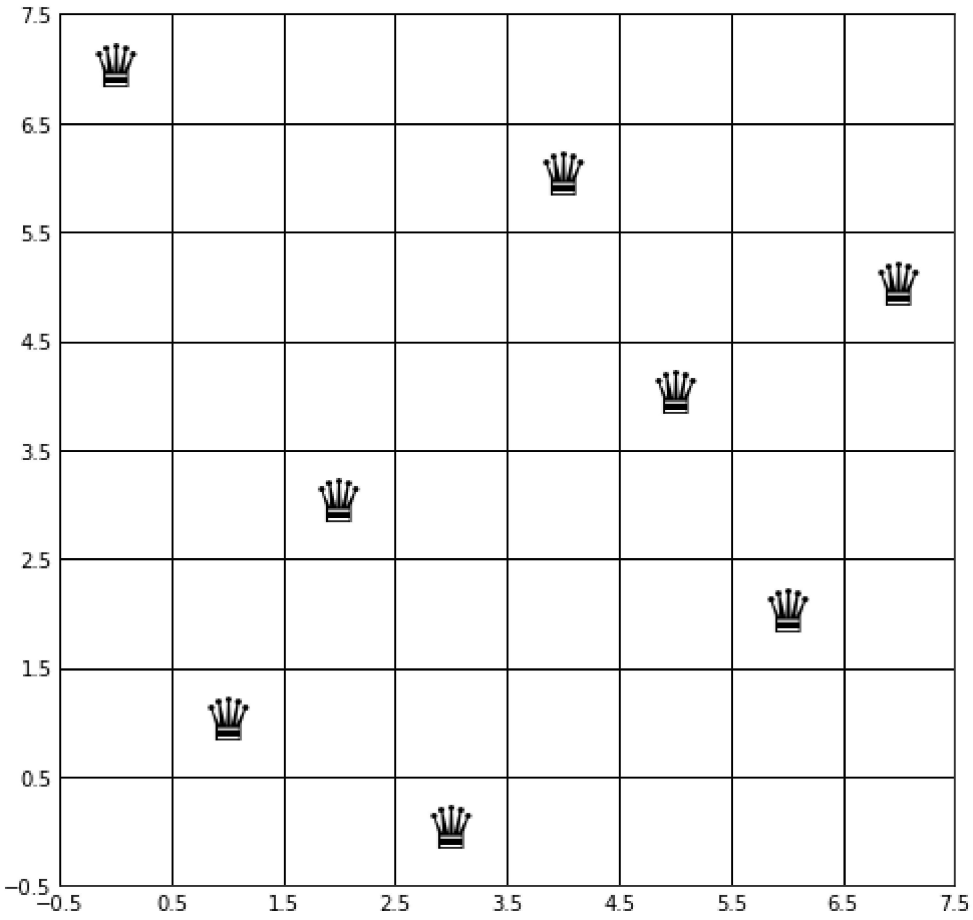
In [1]:
```
import matplotlib.pyplot as plt
import numpy as np
from tqdm import tqdm
from CSP import CSP
import pandas as pd


def plot_queens_board(board):
    n = board.shape[0]
    fig, ax = plt.subplots(figsize=(8, 8))
    ax.set_xticks(np.arange(-.5, n, 1))
    ax.set_yticks(np.arange(-.5, n, 1))
    ax.grid(color='k', linestyle='-', linewidth=1)
    ax.tick_params(axis='both', which='both', length=0)
    ax.set_axisbelow(True)
    for i in range(n):
        for j in range(n):
            if board[i][j] == 1:
                ax.text(j, n - i - 1, '♕', ha='center', va='center', fontsize
```

In this problem we are going to solve N-Queens problem with different methods. The n-queens puzzle is the problem of placing n chess queens on an n×n chessboard so that no two queens threaten each other. thus, a solution requires that no two queens share the same row, column, or diagonal.

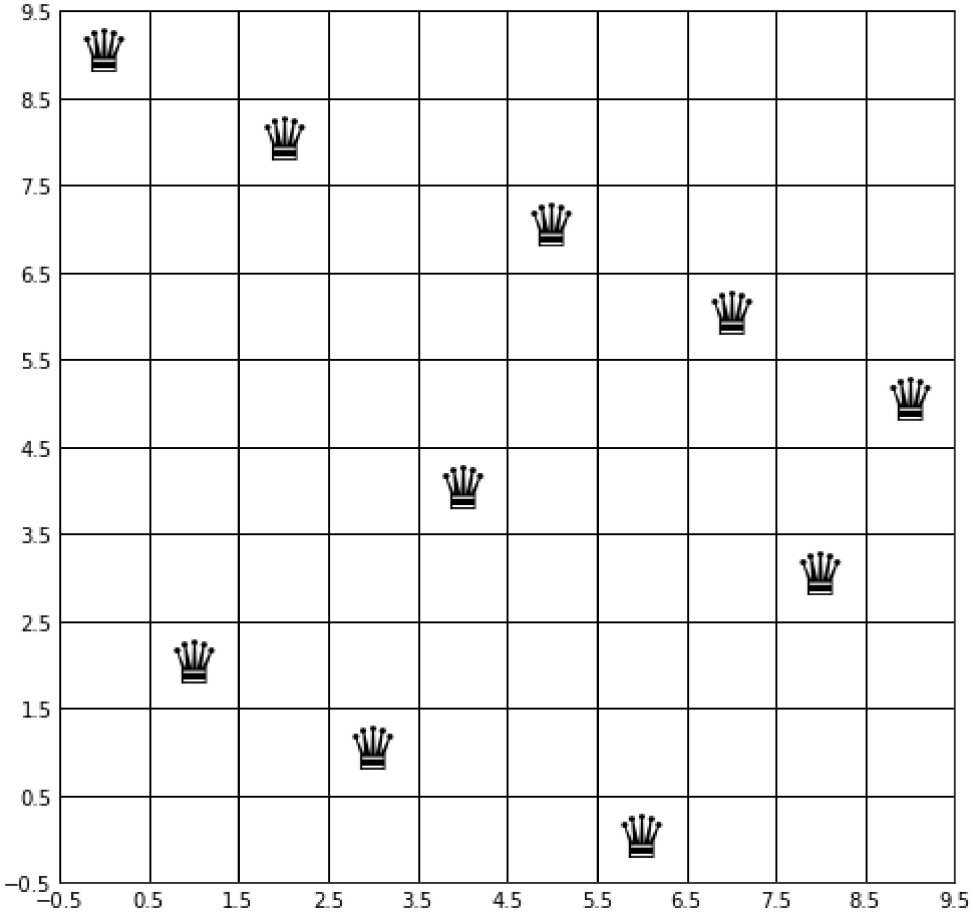


Complete the *CSP.py* file and then run the following cells to test your implementation.

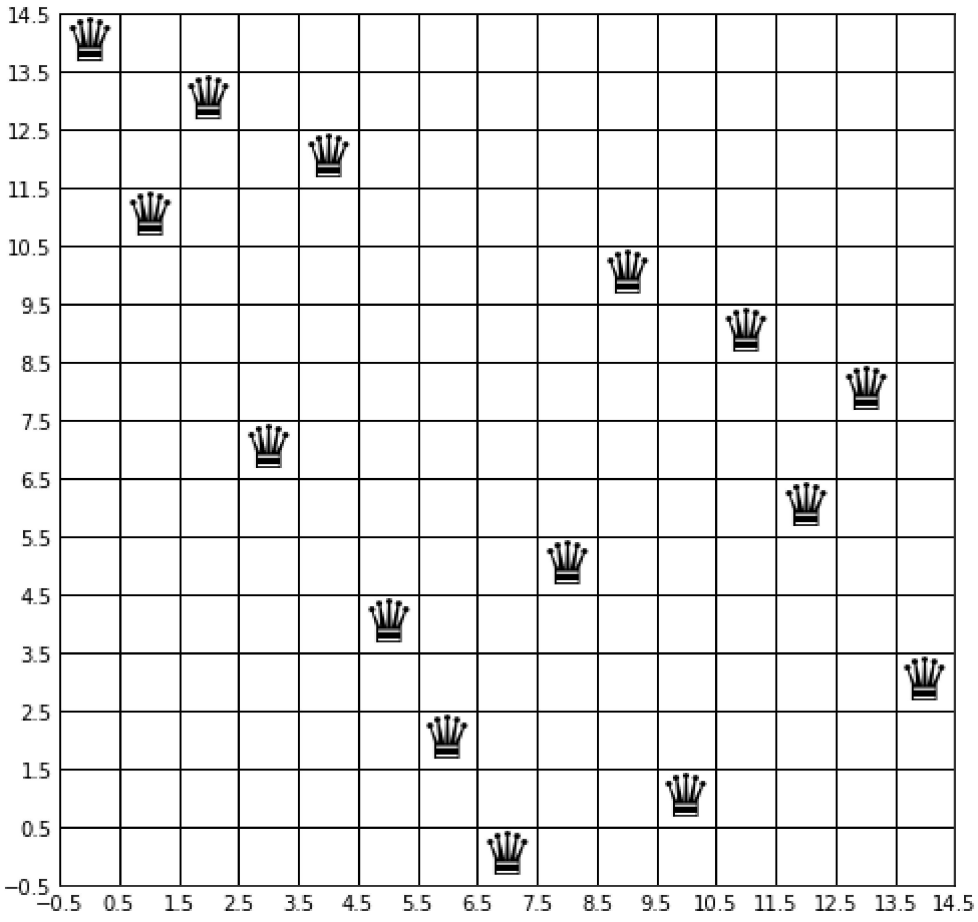
In [4]:
```
csp = CSP(8)
solution = csp.solve_problem_with_backtrack()
plot_queens_board(solution)
```



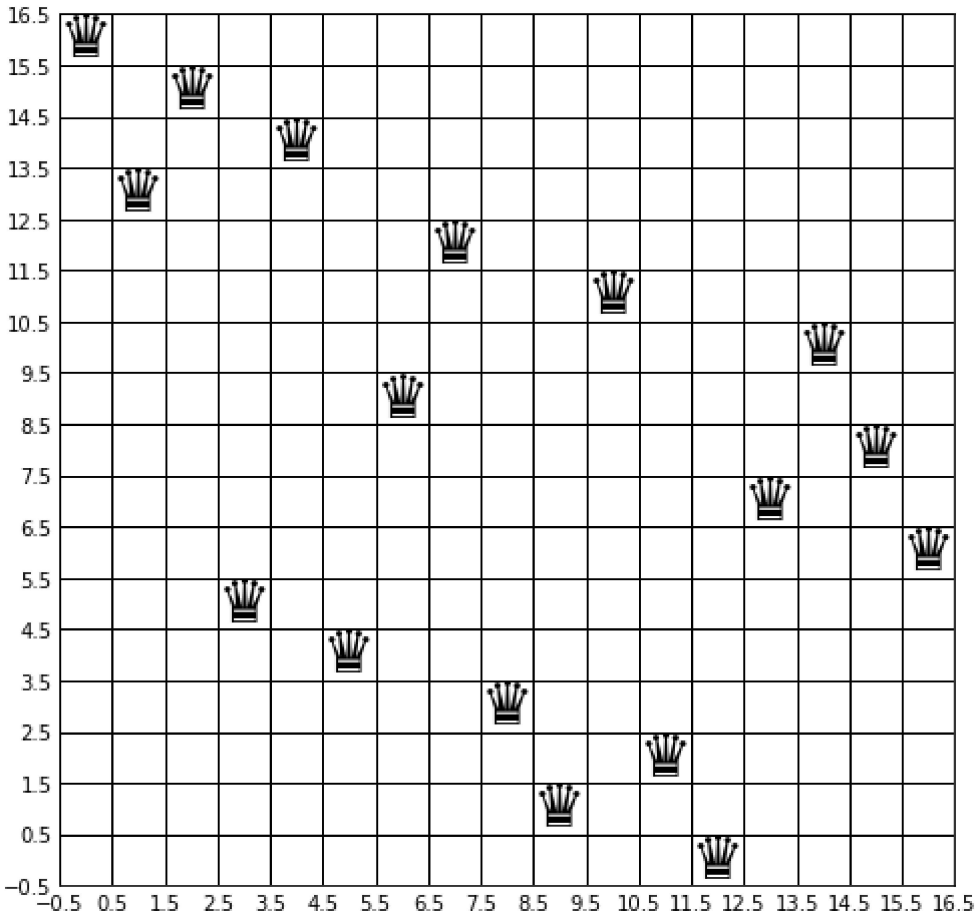
In [5]:
```
csp = CSP(10)
solution = csp.solve_problem_with_backtrack()
plot_queens_board(solution)
```



In [16]:
```
csp = CSP(15)
solution = csp.solve_problem_with_forward_check()
plot_queens_board(solution)
```



In [5]:
```
csp = CSP(17)
solution = csp.solve_problem_with_forward_check()
plot_queens_board(solution)
```



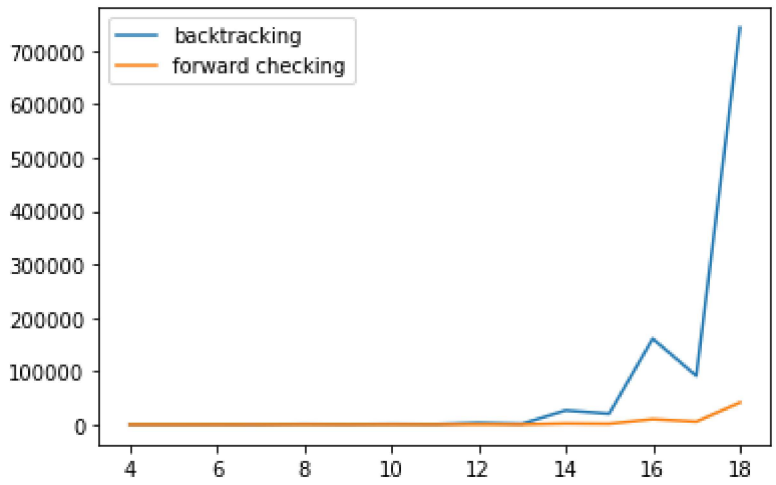
Testing Your Code

After implementation of CSP functions, we want to plot different method iteration for each N.

Don't change this cell

In [6]:
```
# You are not allowed to change this cell!
backtracking_iterations = []
forward_checking_iterations = []


def run_tests(n):
    global backtracking_iterations
    for n in tqdm(range(4, n + 1)):
        csp = CSP(n)
        csp.solve_problem_with_backtrack()
        backtracking_iterations.append(csp.number_of_iteration)
        csp = CSP(n)
        csp.solve_problem_with_forward_check()
        forward_checking_iterations.append(csp.number_of_iteration)
    x = range(4, n + 1)
    plt.plot(x, backtracking_iterations, label='backtracking')
    plt.plot(x, forward_checking_iterations, label='forward checking')
    plt.legend()
    plt.show()


num_tests = 18
run_tests(num_tests)
```

```
100%|██████████| 15/15 [05:12<00:00, 20.81s/it]
```



In [7]:
```
x = range(4, num_tests + 1)

df = pd.DataFrame(
    {"n": x, "forward checking": forward_checking_iterations, "backtracking":
print(df)
```

	n	forward checking	backtracking
0	4	8	26
1	5	5	15
2	6	31	171
3	7	9	42
4	8	113	876
5	9	41	333
6	10	102	975
7	11	52	517
8	12	261	3066
9	13	111	1365
10	14	1899	26495
11	15	1359	20280
12	16	10052	160712
13	17	5374	91222
14	18	41299	743229

In [ ]:
```
```