N-Queen

นายจิรายุส เสนาโนฤทธิ์

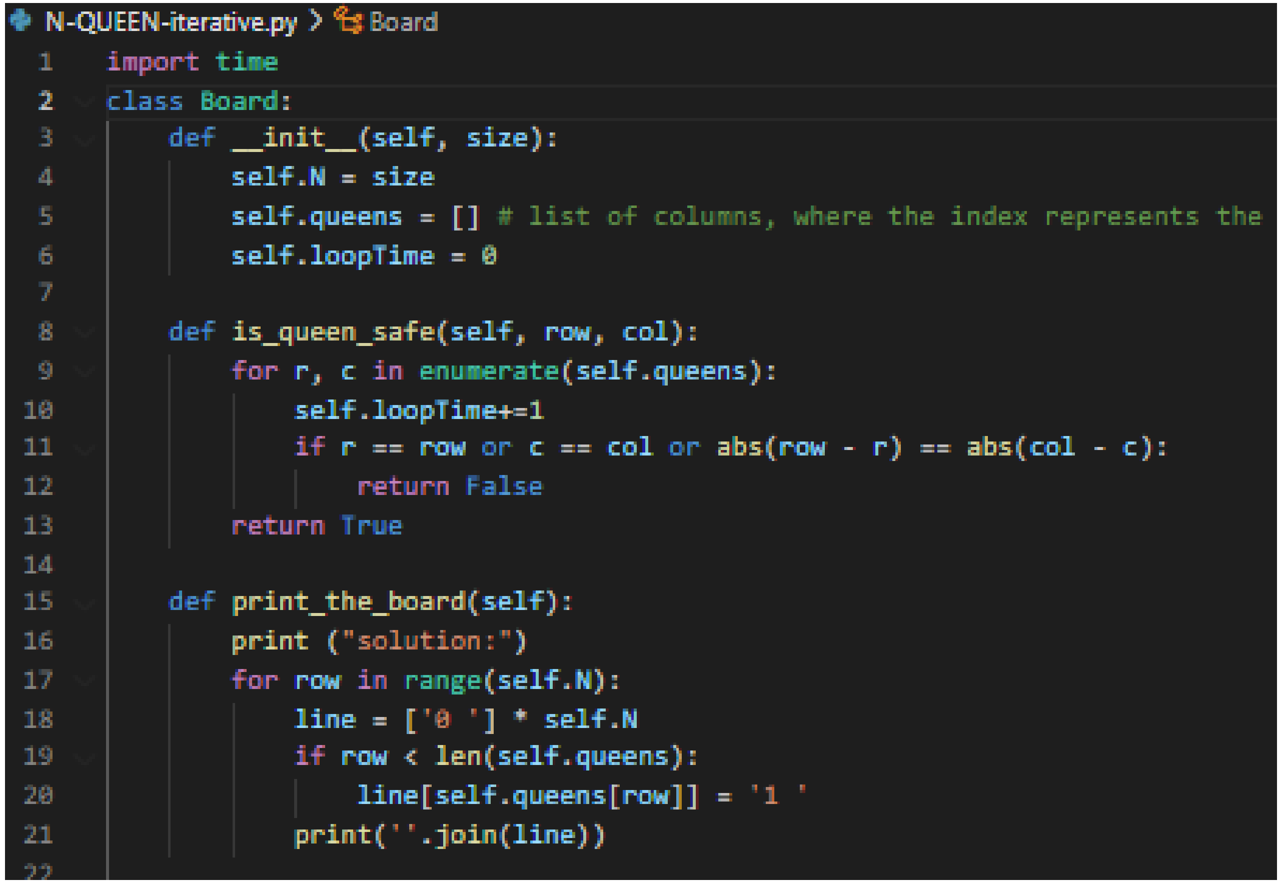
63010151

CPU Memory

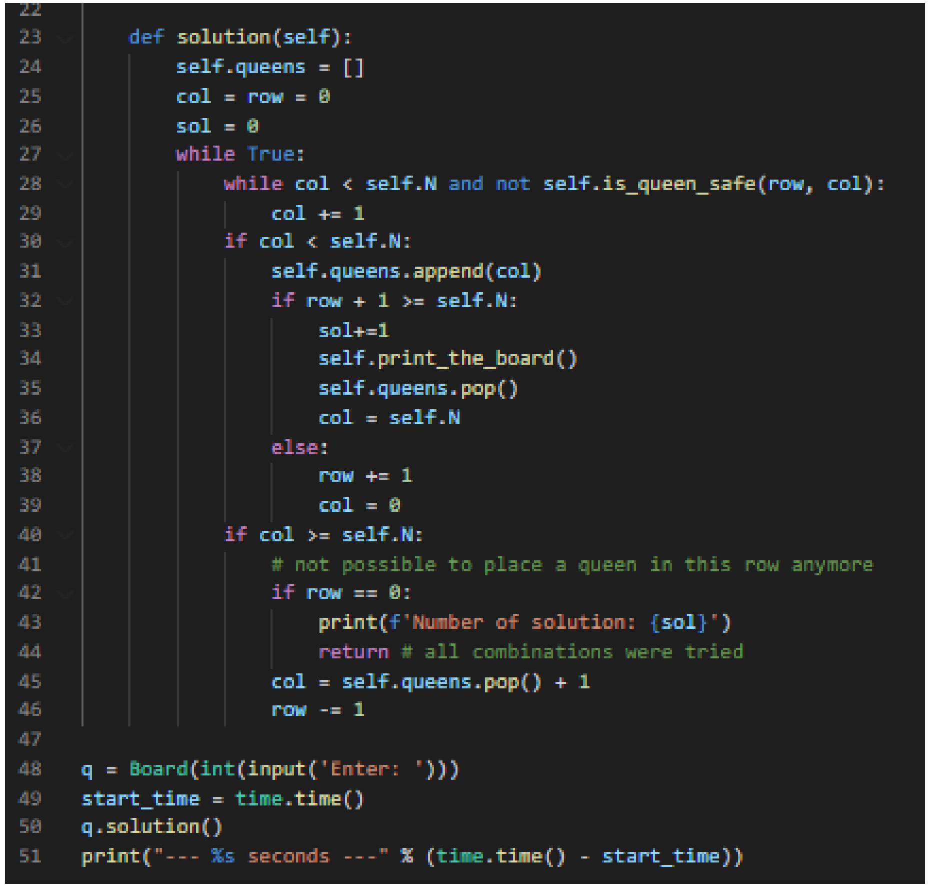
Processor : Intel(R) Core(TM) i7-9750H CPU @2.60GHz 2.59 GHz

RAM : 16 GB

Source code
Iterative



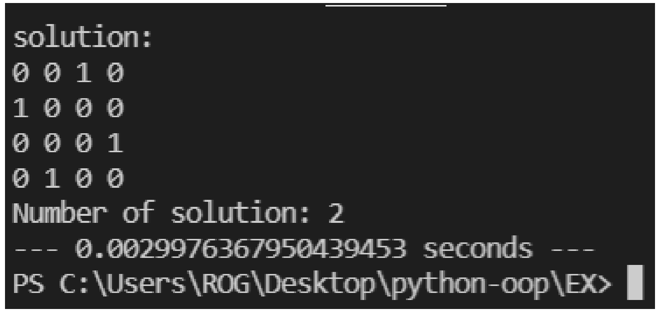
```python
import time
class Board:
    def __init__(self, size):
        self.N = size
        self.queens = [] # list of columns, where the index represents the
        self.loopTime = 0

    def is_queen_safe(self, row, col):
        for r, c in enumerate(self.queens):
            self.loopTime+=1
            if r == row or c == col or abs(row - r) == abs(col - c):
                return False
        return True

    def print_the_board(self):
        print ("solution:")
        for row in range(self.N):
            line = ['0 '] * self.N
            if row < len(self.queens):
                line[self.queens[row]] = '1 '
            print(''.join(line))
```



```python
    def solution(self):
        self.queens = []
        col = row = 0
        sol = 0
        while True:
            while col < self.N and not self.is_queen_safe(row, col):
                col += 1
            if col < self.N:
                self.queens.append(col)
                if row + 1 >= self.N:
                    sol+=1
                    self.print_the_board()
                    self.queens.pop()
                    col = self.N
                else:
                    row += 1
                    col = 0
            if col >= self.N:
                # not possible to place a queen in this row anymore
                if row == 0:
                    print(f'Number of solution: {sol}')
                    return # all combinations were tried
                col = self.queens.pop() + 1
                row -= 1

q = Board(int(input('Enter: ')))
start_time = time.time()
q.solution()
print("--- %s seconds ---" % (time.time() - start_time))
```

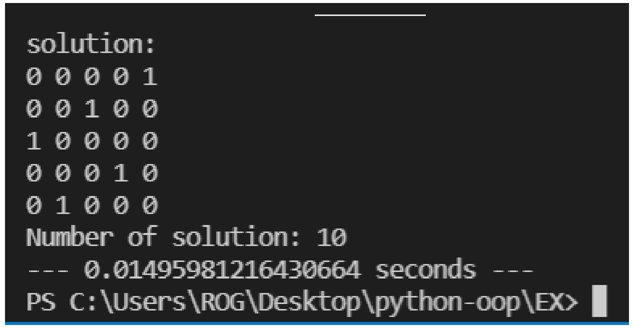
การรัน และจับเวลา ของแต่ละอินพุต (Iterative)

Input = 4



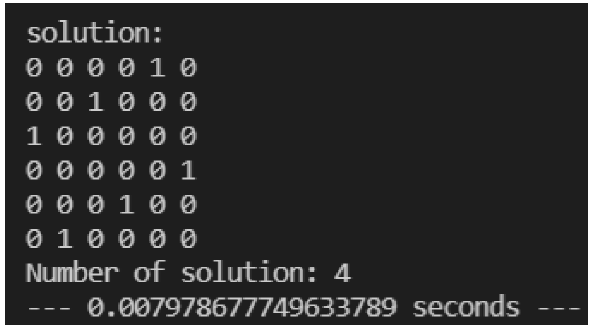
```
solution:
0 0 1 0
1 0 0 0
0 0 0 1
0 1 0 0
Number of solution: 2
--- 0.0029976367950439453 seconds ---
PS C:\Users\ROG\Desktop\python-oop\EX>
```

Input = 5



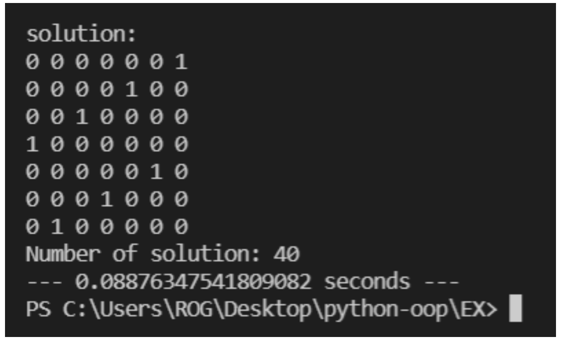
```
solution:
0 0 0 0 1
0 0 1 0 0
1 0 0 0 0
0 0 0 1 0
0 1 0 0 0
Number of solution: 10
--- 0.014959812164306640 seconds ---
PS C:\Users\ROG\Desktop\python-oop\EX>
```

Input = 6



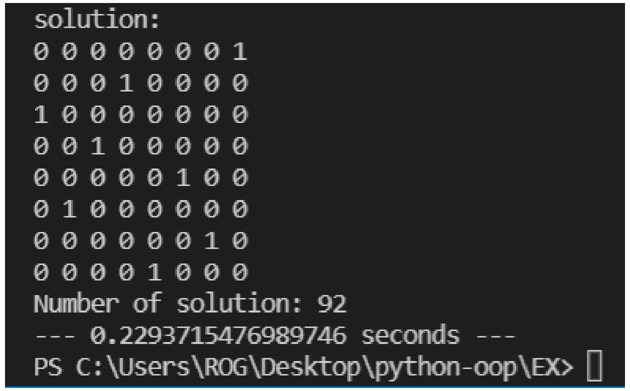
```
solution:
0 0 0 0 1 0
0 0 1 0 0 0
1 0 0 0 0 0
0 0 0 0 0 1
0 0 0 1 0 0
0 1 0 0 0 0
Number of solution: 4
--- 0.007978677749633789 seconds ---
```

Input = 7



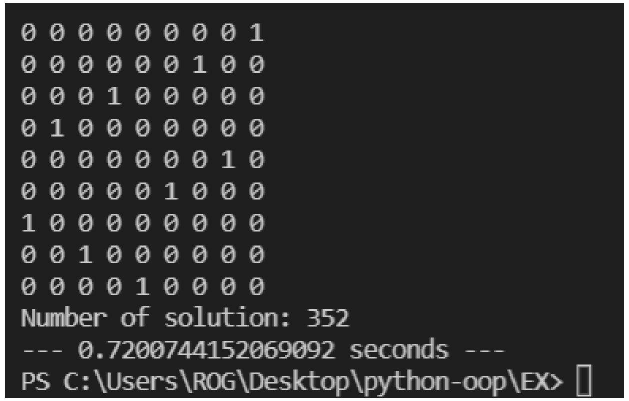
```
solution:
0 0 0 0 0 0 1
0 0 0 0 1 0 0
0 0 1 0 0 0 0
1 0 0 0 0 0 0
0 0 0 0 0 1 0
0 0 0 1 0 0 0
0 1 0 0 0 0 0
Number of solution: 40
--- 0.08876347541809082 seconds ---
PS C:\Users\ROG\Desktop\python-oop\EX>
```

Input = 8



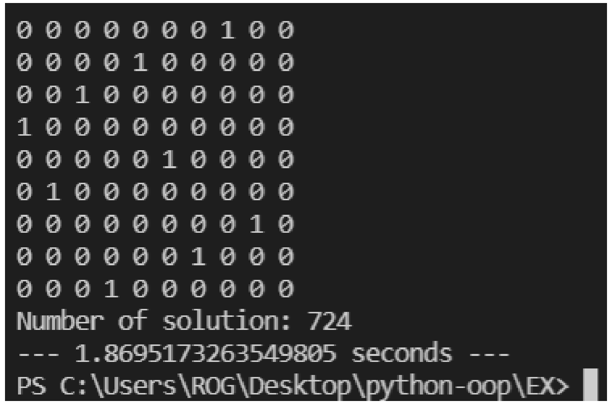
```
solution:
0 0 0 0 0 0 0 1
0 0 0 1 0 0 0 0
1 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0
0 0 0 0 0 1 0 0
0 1 0 0 0 0 0 0
0 0 0 0 0 0 1 0
0 0 0 0 1 0 0 0
Number of solution: 92
--- 0.2293715476989746 seconds ---
PS C:\Users\ROG\Desktop\python-oop\EX>
```

Input = 9



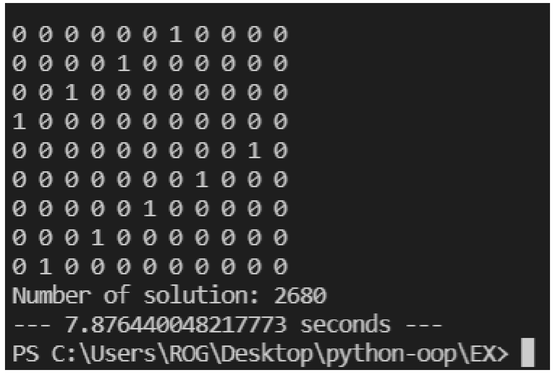
```
0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 1 0 0
0 0 0 1 0 0 0 0 0
0 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 0
0 0 0 0 0 1 0 0 0
1 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0
0 0 0 0 1 0 0 0 0
Number of solution: 352
--- 0.7200744152069092 seconds ---
PS C:\Users\ROG\Desktop\python-oop\EX>
```

Input = 10



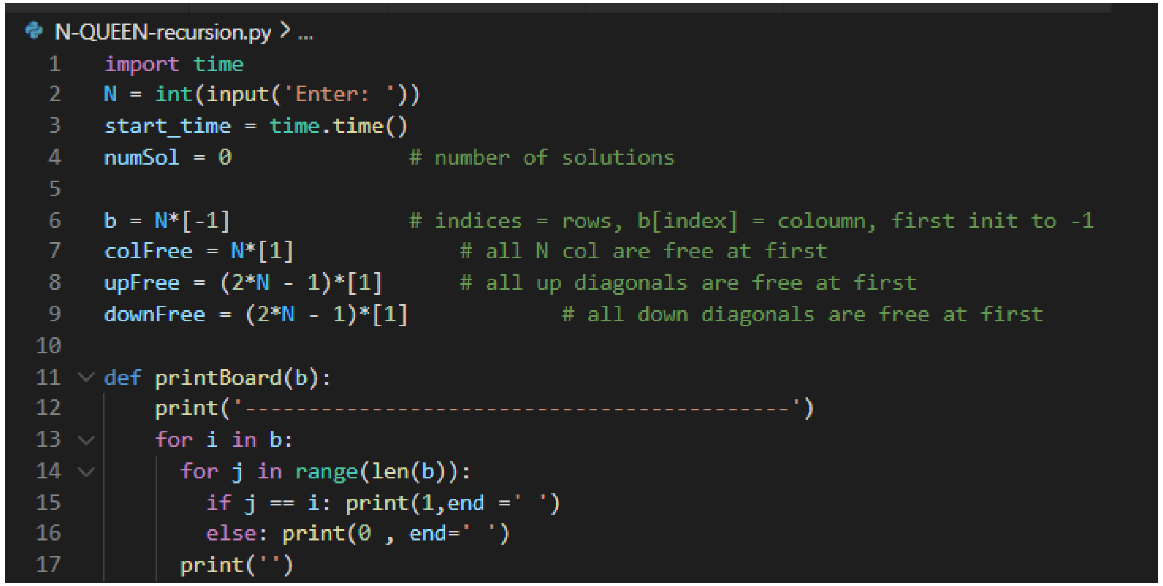
```
0 0 0 0 0 0 0 1 0 0
0 0 0 0 1 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0
0 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0
0 0 0 0 0 0 1 0 0 0
0 0 0 1 0 0 0 0 0 0
Number of solution: 724
--- 1.8695173263549805 seconds ---
PS C:\Users\ROG\Desktop\python-oop\EX>
```

Input = 11

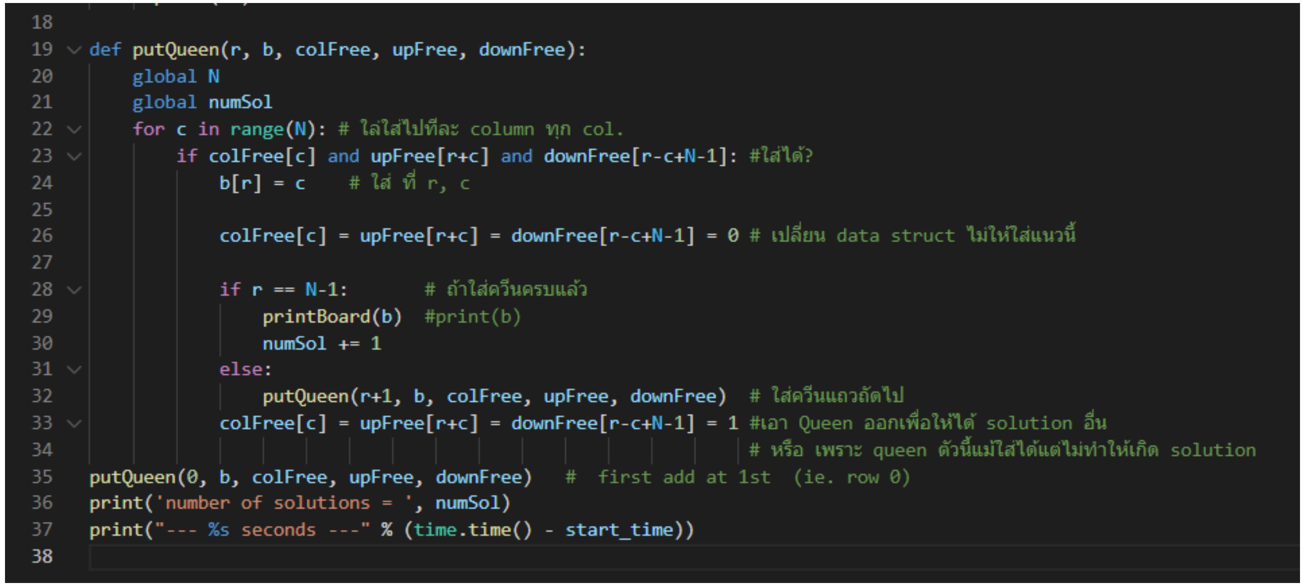


```
0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 1 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 0
0 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 1 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0 0
Number of solution: 2680
--- 7.876440048217773 seconds ---
PS C:\Users\ROG\Desktop\python-oop\EX>
```

Source code recursion



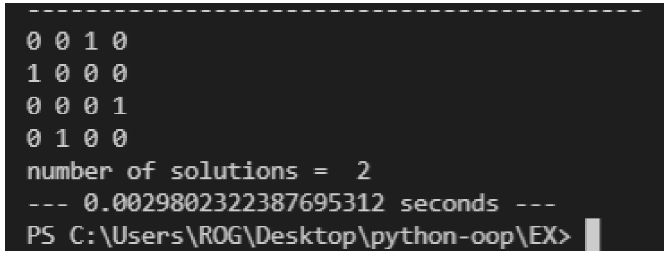
```python
import time
N = int(input('Enter: '))
start_time = time.time()
numSol = 0              # number of solutions

b = N*[-1]              # indices = rows, b[index] = coloumn, first init to -1
colFree = N*[1]             # all N col are free at first
upFree = (2*N - 1)*[1]      # all up diagonals are free at first
downFree = (2*N - 1)*[1]            # all down diagonals are free at first

def printBoard(b):
    print('-------------------------------------------')
    for i in b:
      for j in range(len(b)):
        if j == i: print(1,end =' ')
        else: print(0 , end=' ')
      print('')
```



```python

def putQueen(r, b, colFree, upFree, downFree):
    global N
    global numSol
    for c in range(N): # ไล่ใส่ไปทีละ column ทุก col.
        if colFree[c] and upFree[r+c] and downFree[r-c+N-1]: #ใส่ได้?
            b[r] = c    # ใส่ ที่ r, c

            colFree[c] = upFree[r+c] = downFree[r-c+N-1] = 0 # เปลี่ยน data struct ไม่ให้ใส่แนวนี้

            if r == N-1:       # ถ้าใส่ควีนครบแล้ว
                printBoard(b)  #print(b)
                numSol += 1
            else:
                putQueen(r+1, b, colFree, upFree, downFree)  # ใส่ควีนแถวถัดไป
            colFree[c] = upFree[r+c] = downFree[r-c+N-1] = 1 #เอา Queen ออกเพื่อให้ได้ solution อื่น
                                                             # หรือ เพราะ queen ตัวนี้แม้ใส่ได้แต่ไม่ทำให้เกิด solution
putQueen(0, b, colFree, upFree, downFree)   #  first add at 1st  (ie. row 0)
print('number of solutions = ', numSol)
print("--- %s seconds ---" % (time.time() - start_time))

```

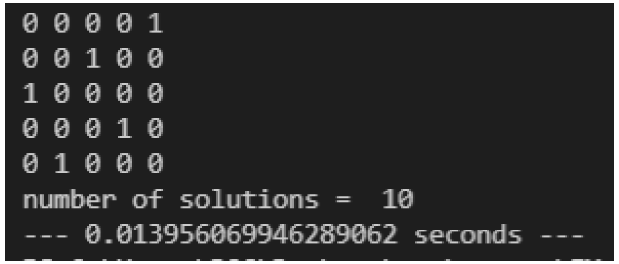
การรัน และจับเวลา ของแต่ละอินพุต (recursion)

Input = 4



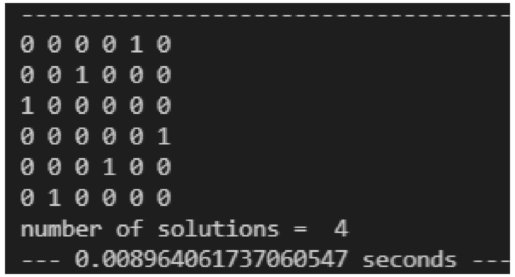
```
-------------------------------------------
0 0 1 0
1 0 0 0
0 0 0 1
0 1 0 0
number of solutions =  2
--- 0.0029802322387695312 seconds ---
PS C:\Users\ROG\Desktop\python-oop\EX> █
```

Input = 5



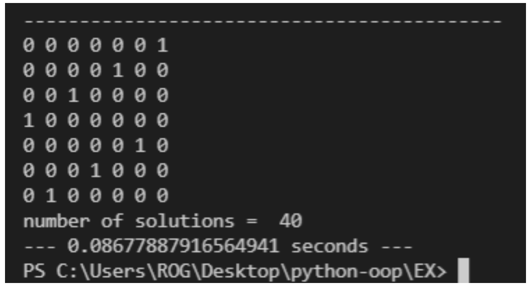
```
0 0 0 0 1
0 0 1 0 0
1 0 0 0 0
0 0 0 1 0
0 1 0 0 0
number of solutions =  10
--- 0.013956069946289062 seconds ---
```

Input = 6



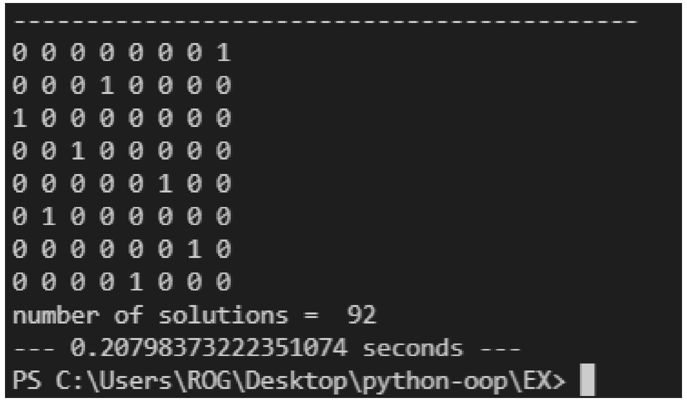
```
-------------------------------------------
0 0 0 0 1 0
0 0 1 0 0 0
1 0 0 0 0 0
0 0 0 0 0 1
0 0 0 1 0 0
0 1 0 0 0 0
number of solutions =  4
--- 0.008964061737060547 seconds ---
```

Input = 7



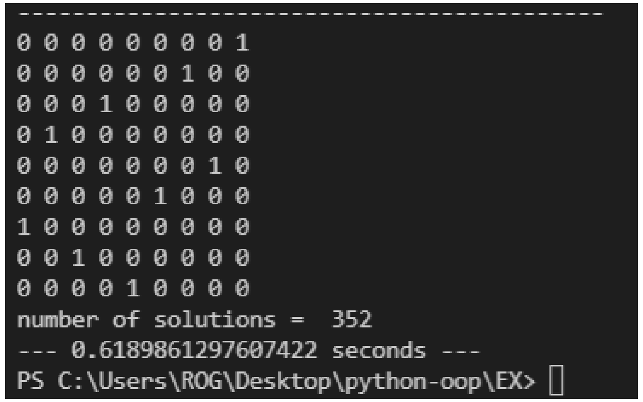
```
-------------------------------------------
0 0 0 0 0 0 1
0 0 0 0 1 0 0
0 0 1 0 0 0 0
1 0 0 0 0 0 0
0 0 0 0 0 1 0
0 0 0 1 0 0 0
0 1 0 0 0 0 0
number of solutions =  40
--- 0.08677887916564941 seconds ---
PS C:\Users\ROG\Desktop\python-oop\EX> █
```

Input = 8



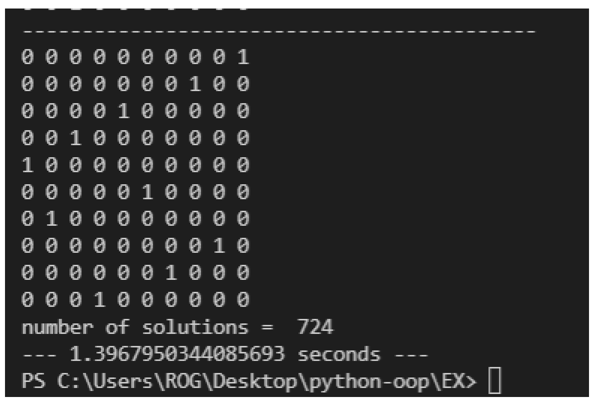
```
-------------------------------------------
0 0 0 0 0 0 0 1
0 0 0 1 0 0 0 0
1 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0
0 0 0 0 0 1 0 0
0 1 0 0 0 0 0 0
0 0 0 0 0 0 1 0
0 0 0 0 1 0 0 0
number of solutions =  92
--- 0.20798373222351074 seconds ---
PS C:\Users\ROG\Desktop\python-oop\EX> █
```

Input = 9



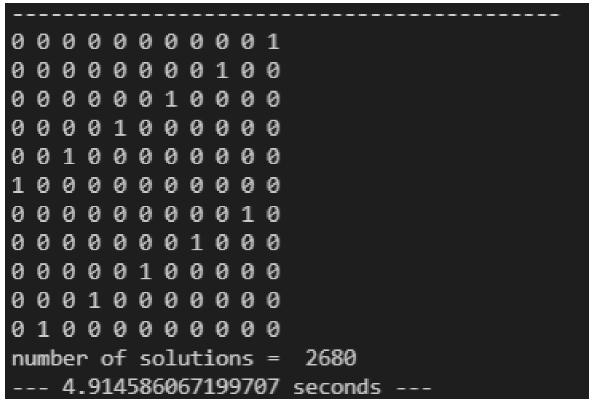
```
-------------------------------------------
0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 1 0 0
0 0 0 1 0 0 0 0 0
0 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 0
0 0 0 0 0 1 0 0 0
1 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0
0 0 0 0 1 0 0 0 0
number of solutions =  352
--- 0.6189861297607422 seconds ---
PS C:\Users\ROG\Desktop\python-oop\EX> ▯
```

Input = 10



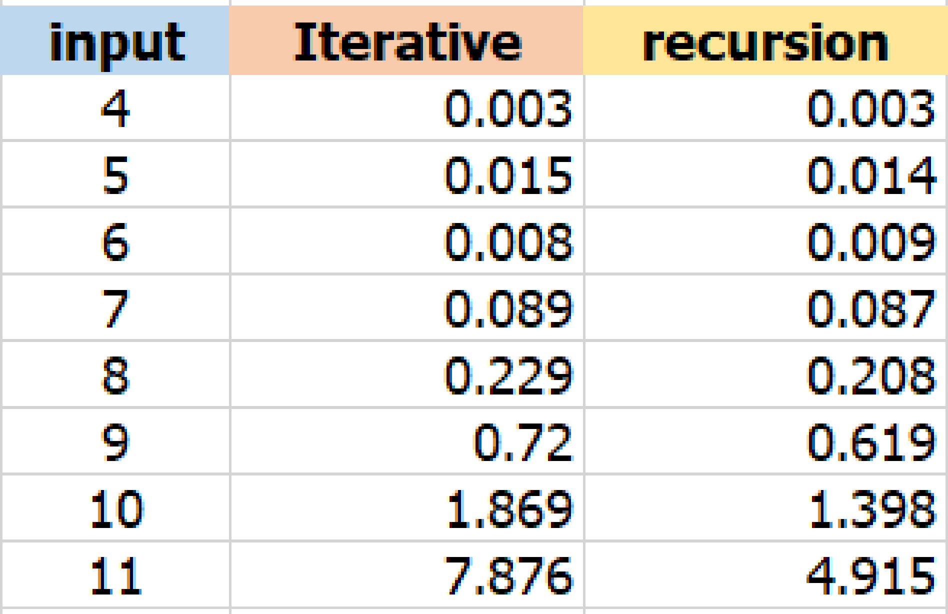
```
-------------------------------------------
0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 1 0 0
0 0 0 0 1 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0
0 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0
0 0 0 0 0 0 1 0 0 0
0 0 0 1 0 0 0 0 0 0
number of solutions =  724
--- 1.3967950344085693 seconds ---
PS C:\Users\ROG\Desktop\python-oop\EX> ▯
```

Input = 11



```
-------------------------------------------
0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 1 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 0
0 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 1 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0 0
number of solutions =  2680
--- 4.914586067199707 seconds ---
```

ตารางบันทึกผล



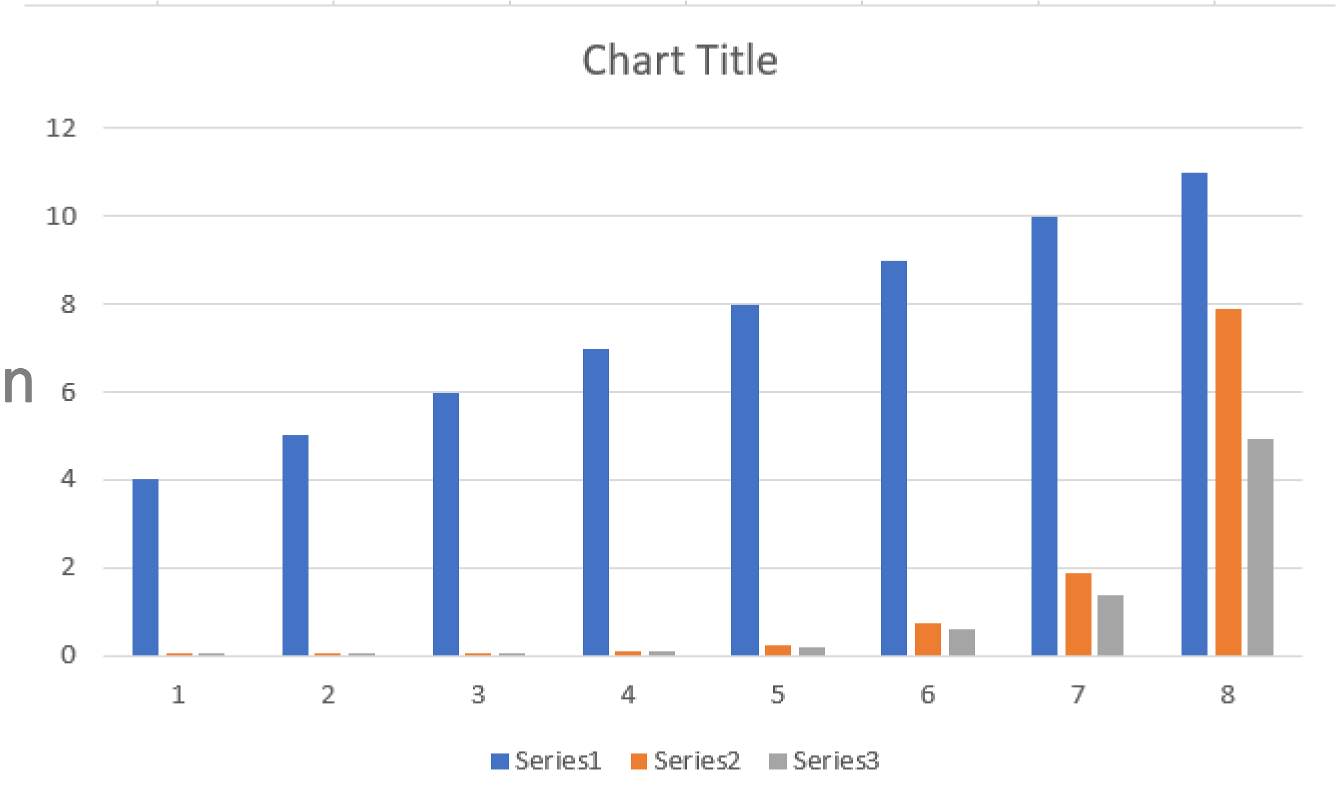
input	Iterative	recursion
4	0.003	0.003
5	0.015	0.014
6	0.008	0.009
7	0.089	0.087
8	0.229	0.208
9	0.72	0.619
10	1.869	1.398
11	7.876	4.915

กราฟเปรียบเทียบ เวลาทั้งสอง อัลกอริทึม

สีฟ้า = input

สีส้ม = เวลา Iterative

สีเทา = เวลา recursion



การวิเคราะห์ผลลัพธ์ที่ได้

จากการทดลอง

อัลกอริทึมของ recursion มีการประมวลผลที่ไวกว่า Iterative

ในช่วง input 8 ถึง 11 จะเริ่มเห็นความแตกต่างระหว่างเวลาการประมวลผล

แหล่งอ้างอิง

Iterative - https://stackoverflow.com/questions/42318343/avoid-duplicates-in-n-queen-iterative-solutions-no-recursion-allowed

Recursion - https://colab.research.google.com/drive/1nhVvTij1LuF-nB1okf9MHtyTdpmARzdG