AI

Practical 2: N-queens and Tower of Hanoi

1)Write a program to simulate

def tower\_of\_hanoi(n,a,b,c):

if n==1:

print(f"Move disk 1 from {a} to {c}")

return

tower\_of\_hanoi(n-1,a,c,b)

print(f"Move disk {n} from {a} to {c}")

tower\_of\_hanoi(n-1,b,a,c)

num\_disks=5

tower\_of\_hanoi(num\_disks,'A','B','C')

o/p  
Move disk 1 from A to C

Move disk 2 from A to B

Move disk 1 from C to B

Move disk 3 from A to C

Move disk 1 from B to A

Move disk 2 from B to C

Move disk 1 from A to C

Q2)

Code:

def print\_board(board):

for row in board:

print(" ".join(row))

print()

def is\_safe(board,row,col,n):

for i in range (row):

if board[i][col]=='Q':

return False

i,j=row,col

while i>=0 and j>=0:

if board[i][j]=='Q':

return False

i-=1

j-=1

i,j=row,col

while i>=0 and j<n:

if board[i][j]=='Q':

return False

i-=1

j+=1

return True

def solve\_queens(board,row,n):

if row == n:

print\_board(board)

return True

for col in range(n):

if is\_safe(board,row,col,n):

board[row][col]='Q'

if solve\_queens(board,row+1,n):

return True

board[row][col]='.'

return False

def four\_queens():

n=4

board=[['.' for \_ in range(n)]for \_ in range (n)]

if not solve\_queens(board,0,n):

print("No solution found")

four\_queens()

o/p

. Q . .

. . . Q

Q . . .

. . Q .