1.SELECTION SORT:

def selectionsort(arr):

n=len(arr)

for i in range(n):

mini=i

for j in range(i+1,n):

if arr[j]<arr[mini]:

mini=j

arr[i],arr[mini] = arr[mini],arr[i]

return arr

arr=[1,8,9,7,2]

sortedarray=selectionsort(arr)

print("sorted array",sortedarray)

2.BUBBLE SORT:

def bubble\_sort(arr):

n = len(arr)

for i in range(n):

for j in range(0, n-i-1):

if arr[j] > arr[j+1]:

arr[j], arr[j+1] = arr[j+1], arr[j]

return arr

arr = [64, 34, 25, 12, 22, 11, 90]

sorted\_arr = bubble\_sort(arr)

print("Sorted array:", sorted\_arr)

3.INSERTION SORT:

def insertion\_sort(arr):

for i in range(1, len(arr)):

key = arr[i]

j = i - 1

while j >= 0 and key < arr[j]:

arr[j + 1] = arr[j]

j -= 1

arr[j + 1] = key

arr = [12, 11, 13, 5, 6]

insertion\_sort(arr)

print("Sorted array is:", arr)

4.N-QUEENS PROBLEM:

import matplotlib.pyplot as plt

import numpy as np

def draw\_board(queens\_positions, n):

board = np.zeros((n, n))

for row, col in enumerate(queens\_positions):

board[row, col] = 1

plt.imshow(board, cmap='binary')

plt.xticks(range(n))

plt.yticks(range(n))

plt.gca().invert\_yaxis()

plt.grid(False)

plt.show()

def n\_queens(n):

def is\_safe(queens, row, col):

for r in range(row):

if queens[r] == col or \

queens[r] - r == col - row or \

queens[r] + r == col + row:

return False

return True

def solve\_n\_queens(row, queens):

if row == n:

solutions.append(queens[:])

return

for col in range(n):

if is\_safe(queens, row, col):

queens[row] = col

solve\_n\_queens(row + 1, queens)

solutions = []

solve\_n\_queens(0, [-1] \* n)

return solutions

# Visual representations for N = 4, N = 5, and N = 8

n\_values = [4, 5, 8]

for n in n\_values:

solutions = n\_queens(n)

for solution in solutions:

draw\_board(solution, n)

5.MERGE SORT:

def merge\_sort(arr):

if len(arr) <= 1:

return arr;

mid = len(arr) // 2

left = arr[:mid]

right = arr[mid:]

left = merge\_sort(left)

right = merge\_sort(right)

return merge(left, right)

def merge(left, right):

result = []

i = j = 0

while i < len(left) and j < len(right):

if left[i] < right[j]:

result.append(left[i])

i += 1

else:

result.append(right[j])

j += 1

while i < len(left):

result.append(left[i])

i += 1

while j < len(right):

result.append(right[j])

j += 1

return result