**Bubble sort algorithm**

The Bubble Sort algorithm is a simple algorithm used to arrange elements in a list in either ascending or descending order. The algorithm works by comparing adjacent elements in the list and swapping them if they are in the wrong order. This process is repeated until no more swaps occur between the elements.

def bubble\_sort(arr):

n = len(arr)

for i in range(n-1):

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

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

# Swap the elements if they are in the wrong order

temp = arr[j]

arr[j] = arr[j+1]

arr[j+1] = temp

# Unsorted list

my\_list = [64, 34, 25, 12, 22, 11, 90]

# Call the function to execute the Bubble Sort algorithm

bubble\_sort(my\_list)

# Print the sorted list

print("Sorted list:")

for i in range(len(my\_list)):

print(my\_list[i])

**The Merge Sort algorithm**

is an efficient algorithm used to sort elements in a list in either ascending or descending order. The algorithm works by dividing the list into approximately equal halves, then recursively dividing each half into further halves until reaching individual segments containing only one element.

These individual segments are then merged together in a sequential manner to create larger segments, and this process is repeated until reaching a single segment that contains all the elements in the correct order.

The merging of segments is done in a gradual and intelligent manner. The elements in different segments are compared, sorted, and merged into a single segment correctly. This process is repeated until reaching a final single segment that contains the fully sorted list.

python

def merge\_sort(arr):

if len(arr) <= 1:

return arr

# Divide the array into two halves

mid = len(arr) // 2

left\_half = arr[:mid]

right\_half = arr[mid:]

# Recursively sort the two halves

left\_half = merge\_sort(left\_half)

right\_half = merge\_sort(right\_half)

# Merge the sorted halves

return merge(left\_half, right\_half)

def merge(left, right):

merged = []

left\_index = 0

right\_index = 0

# Compare elements from both halves and merge them in sorted order

while left\_index < len(left) and right\_index < len(right):

if left[left\_index] < right[right\_index]:

merged.append(left[left\_index])

left\_index += 1

else:

merged.append(right[right\_index])

right\_index += 1

# Append any remaining elements from the left or right half

merged.extend(left[left\_index:])

merged.extend(right[right\_index:])

return merged

# Example usage

arr = [5, 2, 8, 1, 9, 4]

sorted\_arr = merge\_sort(arr)

print(sorted\_arr)