**Mergesort**

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 both halves

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

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

# Merge the sorted halves using the sort function

merged = left\_sorted + right\_sorted

merged.sort() # This step uses Python's built-in sort

return merged

# Example usage

arr = [38, 27, 43, 3, 9, 82, 10]

sorted\_arr = merge\_sort(arr)

print("Sorted array:", sorted\_arr)

**Quicksort**

def quick\_sort(arr):

# Base case: if the array has one or zero elements, it's already sorted

if len(arr) <= 1:

return arr

# Choose the first element as the pivot

pivot = arr[0]

left = []

right = []

# Partition the array into two sub-arrays: left (elements < pivot) and right (elements > pivot)

for x in arr[1:]:

if x < pivot:

left.append(x)

else:

right.append(x)

# Recursively sort the left and right sub-arrays and combine them with the pivot

return quick\_sort(left) + [pivot] + quick\_sort(right)

# Example usage

arr = [38, 27, 43, 3, 9, 82, 10]

sorted\_arr = quick\_sort(arr)

print("Sorted array:", sorted\_arr)