快速排序

```python
def quick_sort(arr, start, end):
    """
    快速排序
    """
    if start >= end:
        return
    base = arr[start]  # 设定基准元素
    low, high = start, end
    while low < high:
        while low < high and arr[high] >= base: # arr[high] 比基准元素小则跳出
            high -= 1
        arr[low] = arr[high]

        # 移动low找到符合条件的元素放在 high 处
        while low < high and arr[low] < base: # low指向的元素比基准元素小，则 low 向右移动
            low += 1
        arr[high] = arr[low]

    # low 与 high 重合 退出循环
    arr[low] = base  # 将基准元素放到 low 位置,左边的元素都比基准元素小,右边的元素都比基准元素大
    quick_sort(arr, start, low - 1)
    quick_sort(arr, low + 1, end)
```

堆排序

```python
heapq # 最小堆
import heapq
heap = [1,2,3,4,5,8,9,6]

heapq.heapify(heap)            # 将列表转换为堆
min_x = heapq.heappop(heap)        # 弹出堆顶最小值并重建堆
heapq.heappush(heap, item)  # heap为定义堆，item增加的元素

heapq.merge(heap1, heap2) # 合并两个堆

for i in heapq.merge(heap1, heap2):
    print(i, end=" ")

heapq.nlargest(n, heap)   # n个最大元素
heapq.nsmallest(n, heap)  # n个最小元素


def heapsort(arr):
    """
    堆排序
    """
    h = []
    for value in arr:
        heappush(h, value)
    return [heappop(h) for i in range(len(h))]
```

归并排序

```python
def merge(left, right):
    """
    将两个列表left, right按顺序融合为一个列表res
    """
    res = []
    i, j = 0, 0       # i和j是位置指针，i指left，j指right
    while i < len(left) and j < len(right):
        if left[i] < right[j]:
            res.append(left[i])
            i += 1
        else:
            res.append(right[j])
            j += 1

    if i == len(left):
        for x in right[j:]:
            res.append(x)
    else:
        for x in left[i:]:
            res.append(x)

    return res

def merge_sort(arr):
    """
    归并排序
    """
    if len(arr) <= 1: return arr    # 只有一个元素 不用排序
    mid = len(arr) // 2

    left = merge_sort(arr[: mid])  # 子序列递归调用排序
    right = merge_sort(arr[mid:])

    return merge(left, right)


if __name__ == '__main__':
    a = [4, 7, 8, 3, 5, 9]
    print(merge_sort(a))


# 分解: 待排序的区间分成左右两个子序列
# 合并: 将排好序的子序列按序合并
# 递归: 使用归并排序递归排序子序列

```