

Сортировка слиянием

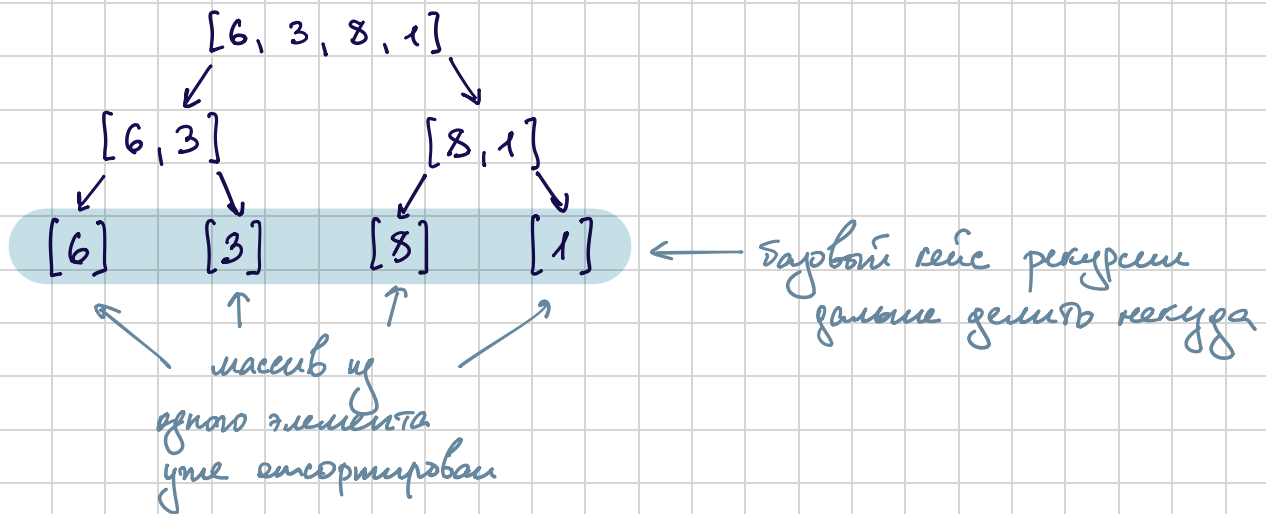
Сортировка слиянием работает по принципу **разделяй и властвуй**

Идея:

1. Раздели список на две половины
2. Отсортируй каждую половину (тем же способом — рекурсивно)
3. Объедини две отсортированные половины в одну

Пример: $[6, 3, 8, 1]$

1) Делим на 2 половины (до конца, рекурсивно)



2) Сбиваем соседние массивы в один.

Будем по очереди сравнивать элементы массивов и каждый раз брать наименьший.

Пример: $[1, 3, 5]$ $[2, 4, 6]$! оба массива отсортированы
[] — результат слияния

$[1, 3, 5]$ $[2, 4, 6]$ $[]$

$1 < 2 \Rightarrow$ добавляем 1,двигаем указатель первого массива

$[1, 3, 5]$ $[2, 4, 6]$ $[1]$

$2 < 3 \Rightarrow$ добавляем 2,двигаем указатель второго массива

$[1, 3, 5]$ $[2, 4, 6]$ $[1, 2]$

$3 < 4$

$[1, 3, 5]$ $[2, 4, 6]$ $[1, 2, 3]$

$4 < 5$

$[1, 3, 5]$ $[2, 4, 6]$ $[1, 2, 3, 4]$

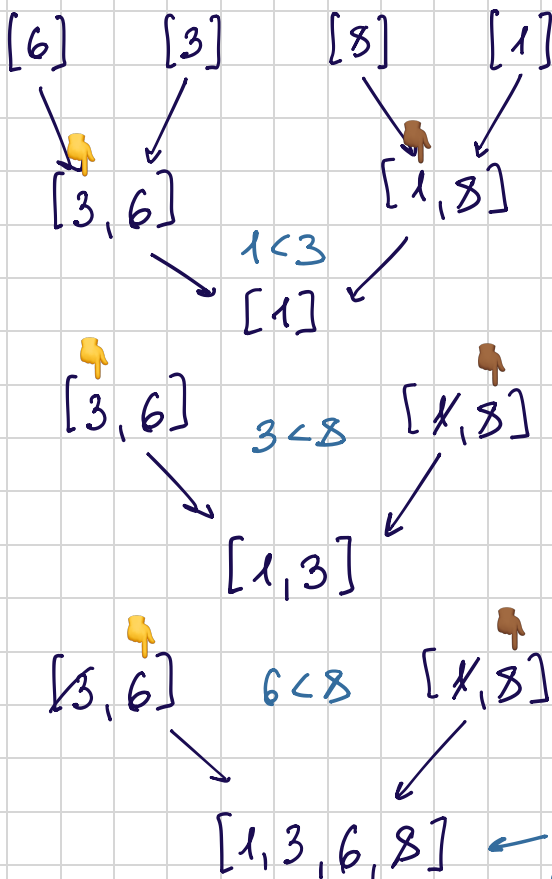
$5 < 6$

$[1, 3, 5]$ $[2, 4, 6]$ $[1, 2, 3, 4, 5]$

дошли до границы
одного из массивов \Rightarrow добавим оставшиеся элементы
другого массива
в конец.

$[1, 3, 5]$ $[2, 4, 6]$ $[1, 2, 3, 4, 5, 6]$ итог.

Вернемся к нашему примеру.



$[1, 3, 6, 8]$ \leftarrow добавим 8 в конец, т.к. в левом массиве мы дошли до конца.

```

1 def merge_sort(arr):
2     if len(arr) <= 1:
3         return arr
4
5     mid = len(arr) // 2
6     left = merge_sort(arr[:mid])
7     right = merge_sort(arr[mid:])
8
9     return merge(left, right)
10
11 def merge(left, right):
12     result = []
13     i = j = 0
14
15     while i < len(left) and j < len(right):
16         if left[i] < right[j]:
17             result.append(left[i])
18             i += 1
19         else:
20             result.append(right[j])
21             j += 1
22
23     result.extend(left[i:])
24     result.extend(right[j:])
25     return result
26
27

```

раздели массив пополам рекурсивно
после разделения массивов.
создаем функцию merge
создаем указатели
если элемент левого массива < элемента правого добавим левый и увелич. указатель левого.
аналогично для правого.