Метод 1: `makeheap_n_log_n`

Добавляем элементы по одному: каждый раз кладём новый элемент в конец массива, потом просеиваем вверх (sift_up):

- Высота кучи при k элементах: не больше $\log_2 k$
- Просейка вверх для каждого из N элементов требует не более $O(\log N)$ шагов

Всего операций:

$$T(N) = O(\log 1 + \log 2 + \ldots + \log N) = O(\log N!)$$

Воспользуемся формулой Стирлинга:

$$\log N! \sim N \log N - N \Rightarrow T(N) = O(N \log N)$$

Вывод: сложность метода `makeheap_n_log_n` - $O(N \log N)$

Метод 2: `makeheap`

На уровне h кучи находится примерно 2^h узлов. Просеивание вниз (sift_down) для каждого родителя: максимальное число перестановок это высота поддерева, т.е. $H - h$, где H высота дерева (примерно $\log_2 N$). Всего операций:

$$T(N) = \sum_{h=0}^{H-1} 2^h \cdot (H - h)$$

Где 2^h - число узлов, а $(H - h)$ - сколько уровней вниз они могут уехать. Распишем сумму:

$$T(N) = \sum_{h=0}^{H-1} 2^h H - \sum_{h=0}^{H-1} 2^h h = H \sum_{h=0}^{H-1} 2^h - \sum_{h=0}^{H-1} h 2^h$$

Рассчитаем каждое слагаемое:

- $$\sum_{h=0}^{H-1} 2^h = 2^H - 1 \approx N$$

  поскольку $2^H \approx N$.

- $$\sum_{h=0}^{H-1} h\, 2^h = 2 + 2(H-1)2^{H-1} \approx O(N \log N)$$

То есть, основная (старшая) часть второго слагаемого — $O(N \log N)$. **Внимание:** Несмотря на порядок $O(N \log N)$ во втором слагаемом, главный член $H \cdot 2^H$ из первого и $2H \cdot 2^{H-1}$ внутри второго при раскрытии на самом деле равны и почти полностью компенсируют друг друга:

$$H \cdot 2^H - 2H \cdot 2^{H-1} = H \cdot 2^H - H \cdot 2^H = 0$$

Остаются только остаточные члены, порядка $O(N)$:

$$T(N) = O(N)$$

Вывод: сложность функции `makeheap` - $O(N)$