**Пирамидальная сортировка:**

Пирамидальная сортировка - это эффективный алгоритм сортировки, который использует структуру данных "куча" для упорядочивания элементов. Он состоит из двух этапов: построение кучи и собственно сортировка.

Давайте рассмотрим простой массив [5, 2, 9, 1, 7, 6] и опишем шаги пирамидальной сортировки:

1. Построение кучи (Heapify):

* Преобразуем массив в кучу. Начнем с середины массива и будем проводить процедуру просеивания вниз (heapify-down) для каждого элемента.
* Начнем с последнего индекса, который имеет дочерние узлы (в данном случае - 2 и 3).
* Процедура heapify-down переносит текущий элемент вниз по куче до тех пор, пока не будет удовлетворено свойство кучи (в данном случае, максимальная куча). Это означает, что дочерние элементы всегда меньше родительского.
* После первого прохода по массиву, у нас будет корректно построенная куча.

1. Сортировка:

* После построения кучи, самый большой элемент будет находиться в корне кучи.
* Мы будем поочередно извлекать максимальный элемент из кучи и помещать его в конец массива.
* Затем уменьшаем размер кучи на 1 и восстанавливаем свойство кучи, вызвав heapify-down на оставшейся части кучи (то есть на неотсортированной части массива).
* После каждой итерации максимальный элемент будет перемещаться в конец массива.
* Повторяем этот процесс до тех пор, пока куча не опустеет.

Процесс сортировки для нашего массива будет выглядеть следующим образом:

Исходный массив: [5, 2, 9, 1, 7, 6]

1. Построение кучи: [9, 7, 6, 1, 2, 5]
2. Сортировка:

* Извлечение максимального элемента (9) и помещение его в конец: [5, 2, 6, 1, 7, 9]
* Уменьшение размера кучи: [7, 5, 6, 1, 2]
* Переупорядочивание кучи: [7, 5, 6, 1, 2] (heapify-down)
* Извлечение максимального элемента (7) и помещение его в конец: [2, 5, 6, 1, 7]
* Уменьшение размера кучи: [6, 5, 2, 1]
* Переупорядочивание кучи: [6, 5, 2, 1] (heapify-down)
* Извлечение максимального элемента (6) и помещение его в конец: [1, 5, 2, 6]
* Уменьшение размера кучи: [5, 1, 2]
* Переупорядочивание кучи: [5, 1, 2] (heapify-down)
* Извлечение максимального элемента (5) и помещение его в конец: [1, 2, 5]
* Уменьшение размера кучи: [2, 1]
* Переупорядочивание кучи: [2, 1] (heapify-down)
* Извлечение максимального элемента (2) и помещение его в конец: [1, 2]
* Уменьшение размера кучи: [1]
* Переупорядочивание кучи: [1] (heapify-down)
* Извлечение максимального элемента (1) и помещение его в конец: [1]
* Отсортированный массив: [1, 2, 5, 6, 7, 9]

**Алгоритм сортировки вставками:**

Это алгоритм представляет собой простой метод сортировки, который сначала строит отсортированную последовательность элементов, а затем по одному вставляет неотсортированные элементы в правильное место в этой последовательности.

Давайте рассмотрим пример сортировки вставками для массива [5, 2, 9, 1, 7, 6]:

1. Исходный массив: [5, 2, 9, 1, 7, 6]
2. Начнем со второго элемента массива (индекс 1), так как массив из одного элемента считается отсортированным.
3. Сравним второй элемент (2) с предыдущим элементом (5):

* Поскольку 2 меньше 5, поменяем их местами: [2, 5, 9, 1, 7, 6]

1. Теперь рассмотрим третий элемент (9) и вставим его в правильное положение относительно уже отсортированной части массива [2, 5]:

* 9 больше 5, поэтому оставим его на своем месте: [2, 5, 9, 1, 7, 6]

1. Рассмотрим четвертый элемент (1) и вставим его в правильное положение:

* Переставляем элементы, пока 1 не окажется перед 2: [1, 2, 5, 9, 7, 6]

1. Пятый элемент (7) вставляем в правильное положение:

* Он больше 5, но меньше 9, поэтому его место между ними: [1, 2, 5, 7, 9, 6]

1. Рассмотрим последний элемент (6) и вставим его в правильное положение:

* Переставляем элементы, пока 6 не окажется перед 7: [1, 2, 5, 6, 7, 9]

**Теперь весь массив отсортирован: [1, 2, 5, 6, 7, 9].**