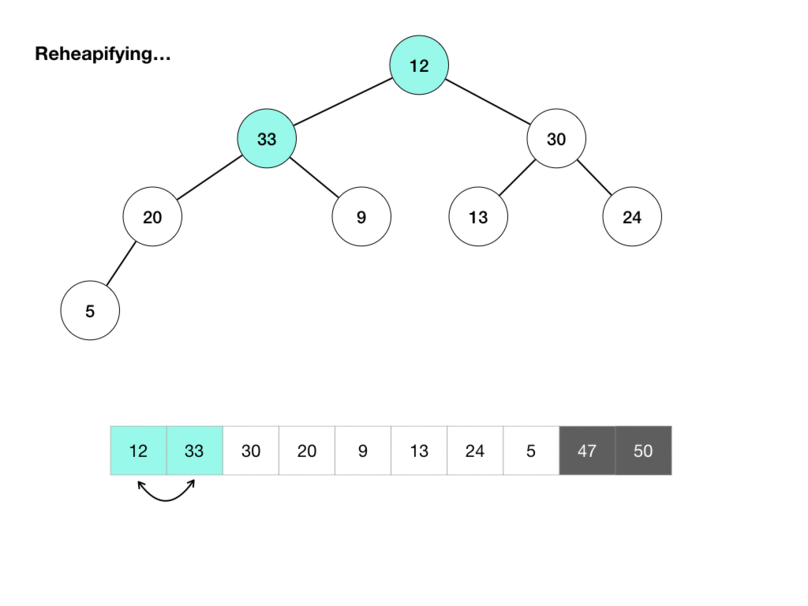
Heap Sort

### Heapsort-Algorithmus

Heapsort ist ein vergleichsbasierter Sortieralgorithmus, der auf einem Binärheap basiert. Ein Binärheap ist eine spezielle Baumstruktur, die die Heap-Eigenschaft erfüllt: In einem Max-Heap ist der Wert jedes Knotens größer oder gleich den Werten seiner Kinder. Heapsort besteht aus zwei Hauptphasen:

1. **Aufbau des Heaps**: Der Eingabe-Array wird in einen Max-Heap umgewandelt. Dies geschieht, indem die heapify-Funktion auf alle nicht-Blattknoten angewendet wird, beginnend mit dem letzten nicht-Blattknoten und endend mit der Wurzel.
2. **Entfernen der Elemente**: Das größte Element (Wurzel des Heaps) wird mit dem letzten Element des Heaps vertauscht, und der Heap wird neu aufgebaut. Dieser Prozess wird wiederholt, bis alle Elemente sortiert sind.



### Pseudocode:

Heapify(A as array, n as int, i as int)

{

max = i

leftchild = 2i + 1

rightchild = 2i + 2

if (leftchild <= n) and (A[i] < A[leftchild])

max = leftchild

else

max = i

if (rightchild <= n) and (A[max] > A[rightchild])

max = rightchild

if (max != i)

swap(A[i], A[max])

Heapify(A, n, max)

}

Heapsort(A as array)

{

n = length(A)

for i = n/2 downto 1

Heapify(A, n ,i)

for i = n downto 2

exchange A[1] with A[i]

A.heapsize = A.heapsize - 1

Heapify(A, i, 0)

}

### Detaillierte Beschreibung des Codes

1. **heapify-Funktion**:
   * Diese Funktion stellt sicher, dass der Unterbaum mit der Wurzel i die Heap-Eigenschaft erfüllt.
   * largest wird initialisiert, um den Index der Wurzel zu speichern.
   * left und right berechnen die Indizes der linken und rechten Kinder des Knotens i.
   * Wenn das linke Kind größer als die Wurzel ist, wird largest auf den Index des linken Kindes gesetzt.
   * Wenn das rechte Kind größer als das größte bisherige ist, wird largest auf den Index des rechten Kindes gesetzt.
   * Wenn largest nicht i ist, wird die Wurzel mit dem größten Element vertauscht, und heapify wird rekursiv auf dem betroffenen Unterbaum aufgerufen.
2. **heapSort-Funktion**:
   * Zuerst wird der Max-Heap aufgebaut, indem heapify auf alle nicht-Blattknoten angewendet wird, beginnend mit dem letzten nicht-Blattknoten und endend mit der Wurzel.
   * Dann wird das größte Element (Wurzel des Heaps) mit dem letzten Element des Heaps vertauscht, und der Heap wird neu aufgebaut. Dieser Prozess wird wiederholt, bis alle Elemente sortiert sind.