# Insertion-Sort

```
input: number array $A$ of length $n$
for $j := 2$ to length($A$) do
    $key := A[j]$
    $i := j - 1$
    while $i > 0$ and $A[i] > key$ do
        $A[i + 1] := A[i]$
        $i := i - 1$
    $A[i + 1] := key$
```

# Heap-Sort

```
input: number array $A$ of length $n$
  build-heap
  while Heap not empty do
      extract root/last element of heap-array becomes new root
      heapify
```

# Quicksort(Idea)

input: number array $A$ of length $n$

1. choose pivot-element $A[pivot]$
2. decompose $A$ into two parts $A[1], \ldots, A[pivot-1]$ and $A[pivot+1], \ldots, A[n]$ with
   2.1 $A[i] < A[pivot]$ for all $i \in \{1, \ldots, pivot-1\}$
   2.2 $A[i] \geq A[pivot]$ for all $i \in \{pivot+1, \ldots, n\}$
3. for each subarray with more than one element use Quicksort for that array

# A version of Quicksort

```
Set $pivot := i$, $l := 1$, $r := n$.
find smallest $l' \geq l$ with $A[l'] \geq A[pivot]$
find biggest $r' \leq r$ with $(A[r'] < A[pivot]$ or $r' = pivot)$
if $r' = l'$ then STOP
swap $A[l']$ and $A[r']$
set $S := pivot$
if $S == l'$
    then $pivot := r'$, $r := r'$
    else $r := r' - 1$
if $S == r'$
    then $pivot := l'$, $l := l'$
    else $l := l' + 1$
```

# Bucket-Sort

Use array $L$ consisting of lists $L[i]$ with $i \in \{1, \dots, m\}$

input: an array of numbers $A$ with $A[i] \in \{1, \dots, m\}$ for $i \in \{1, \dots, n\}$

```
for $j := 1$ to $n$ do
    append $A[j]$ to $L[A[j]]$
construct one big list $L'$ consisting of the $L[1], \dots, L[n]$ by appending
go through $L'$ and return all values in the given order
```