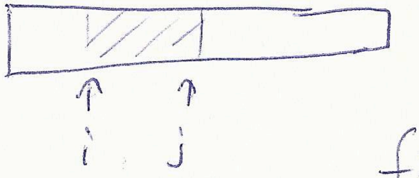
Max.-S.A

naive = brute force	$O(n^3)$
* Divide-and-Conquer	$O(n \cdot \log n)$
Scanline	$O(n)$
(halb-naive)	$O(n^2)$

$O(n) < O(n \log n) < O(n^2) < O(n^3)$



```
für i = 1 bis n          } O(n²)
    für j = 1 bis n              } O(n³)
        Summe?           O(n)
```

$n^2 = n(n)$

$n \gg \log n$

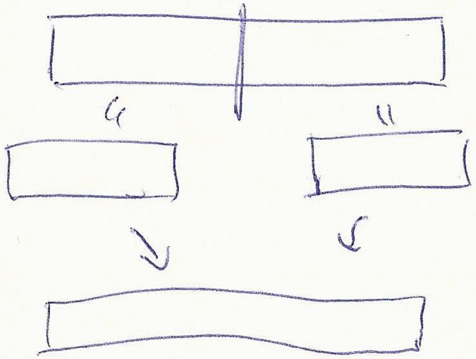
Suchen

binär $O(\log n)$ DAQ

sequentiell → $O(n)$ naiv

~~Interpol.~~
~~Fibonacci~~ } DAQ

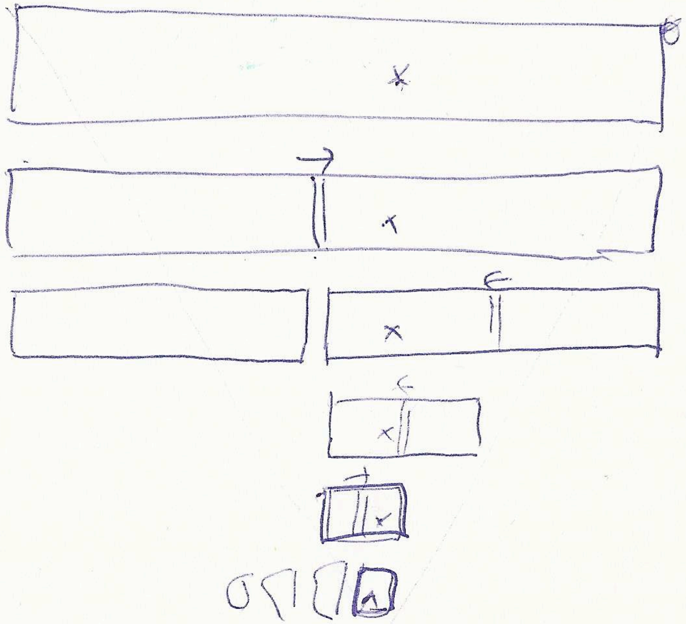
DAQ



n

$\frac{n}{2}$

$\frac{n}{4}$



$A[k] < k$

8, -4, 2, 6, -8, 6, 1, 0

$\overbrace{8, -4, 2, 6}^{12} \mid \overbrace{-8, 6, 1, 0}^{7}$

$\xleftarrow{12} \| \xrightarrow{0}$

mögliche Randergebnisse

→ 0 ←
-8
-2
-1
-1

12 } ← max
12 }
7

⇒ 8, -4, 2, 6 Σ 12 max.

-8, 6, 1, 0

Split / Divide

-8, 6 | 1, 0

Σ 6 Σ 1

$-8_0 \mid 6_6 \,\|\, 1_1 \mid 0_0$

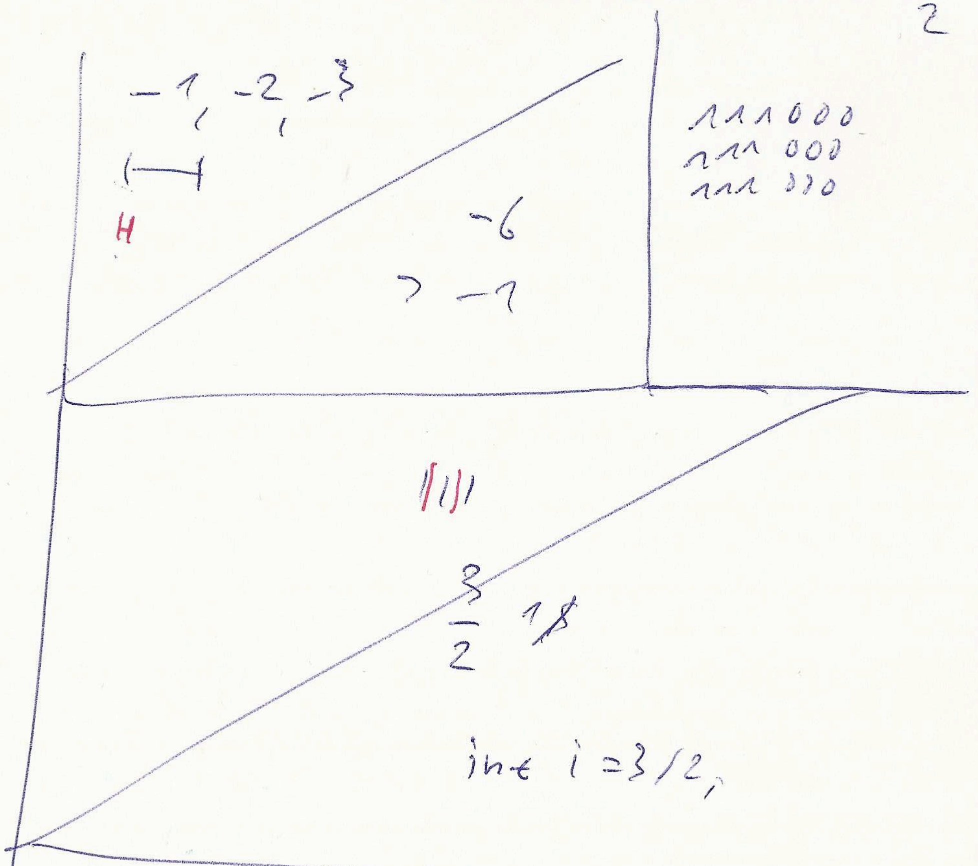
Join / Merge

$-8, 6^{6} \mid 1, 0^{1}$

Σ 7

Join / Merge

$-8, 6, 1, 0^{7}$



$\frac{3}{2}$ ~~1,8~~

int i = 3/2;

== Vergleich 1 == 0 ⇒ falsch
= Zuweisung

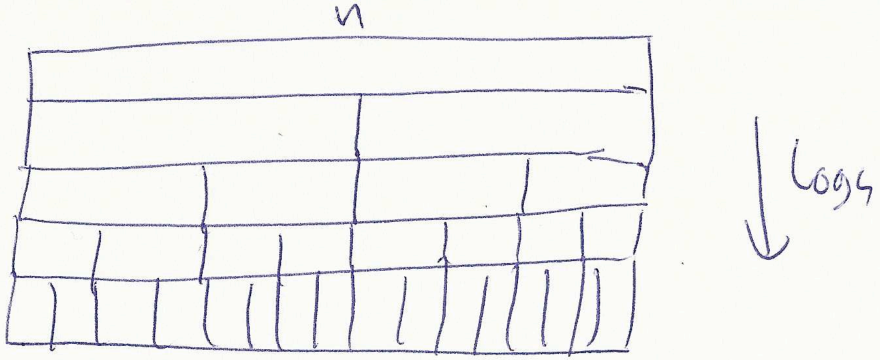
Java

= Vergleich
:= Definition

Mathe

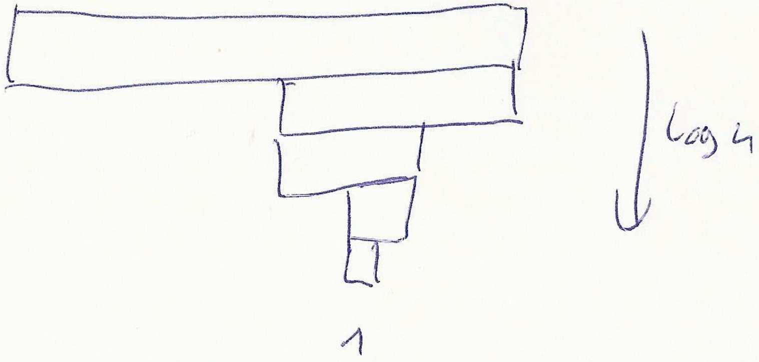
= Vergleich
← Zuweisung

Logik



DAQ
MSA

O(n · log n)



Bin. Suche

O(1 · log n)

Scanline

8, -4, 2, 6, -8, 6, 2, 0

$