5

Gegeben: Array A ganzer Zahlen; Element x

Gesucht: Wo kommt x in A vor?

Naives Vorgehen:

- Vergleiche x der Reihe nach mit A[0], A[1], usw.
- Finden wir i mit A[i] == x, geben wir i aus.
- Andernfalls geben wir -1 aus: „Element nicht gefunden“!

Naives Suchen

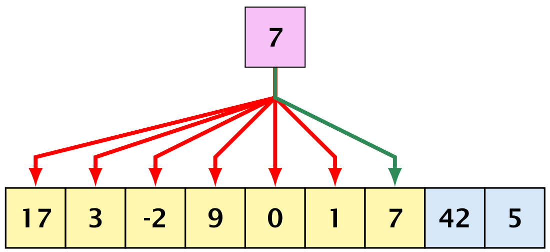
```
Input:  Array A mit Laenge n; Element x
Output: i mit A[i] == x falls existent
        sonst -1

find(A,x)
    i = 0;
    while (i < n && A[i] != x)
        i++;
    if (i == n)
        return -1;
    else
        return i;
```

Naives Suchen

Beispiel

Animation ist nur in der Vorlesungsversion der Folien vorhanden.



yes

Laufzeit Naives Suchen

Best-case:
Wenn x an Position 0.
$\Longrightarrow$ Laufzeit: $\mathcal{O}(1)$.

Worst-case:
Wenn x nicht vorkommt.
$\Longrightarrow$ Laufzeit: $\mathcal{O}(n)$.

...geht das besser?

Binäre Suche

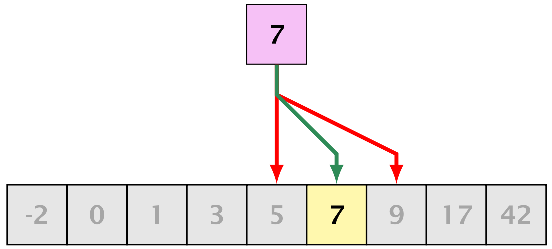
Annahme: Input ist sortiert.
Idee:

- Vergleiche x mit dem Wert, der in der Mitte steht.
- Liegt Gleichheit vor, sind wir fertig.
- Ist x kleiner, brauchen wir nur noch links weitersuchen.
- Ist x größer, brauchen wir nur noch rechts weiter suchen.

$\Longrightarrow$ binäre Suche

Beispiel

Animation ist nur in der Vorlesungsversion der Folien vorhanden.



- wir benötigen nur drei Vergleiche

Implementierung

```
Input:  sortiertes Array A; Element x;
        linker Index nl; rechter Index nr; nl<=nr
Output: Index i; nl ≤ i ≤ nr mit A[i]==x falls existent
        sonst -1
find(A, x, nl, nr) // Inputlaenge ist n=nr-nl+1
    t = (nl + nr) / 2;
    if (A[t] == x)
        return t;
    if (nl == nr)
        return -1;
    if (x > A[t])
        return find(A, x, t+1, nr);
    if (nl < t)
        return find(A, x, nl, t-1);
    return -1;
```


Laufzeit Binäre Suche

Laufzeit:

$$T(n) = \begin{cases} \mathcal{O}(1) & \texttt{A[t] == x} \\ \mathcal{O}(1) & \texttt{nl == nr} \\ \mathcal{O}(1) + T(nr - t) & \texttt{x > A[t]} \\ \mathcal{O}(1) + T(t - nl) & \texttt{nl < t} \end{cases}$$

oder

$$T(n) \leq \begin{cases} \mathcal{O}(1) & n = 1 \\ \mathcal{O}(1) + T(\lfloor n/2 \rfloor) & \text{sonst} \end{cases}$$

Laufzeit Binäre Suche

Lösen der Rekursionsgleichung:

$$T(n) \le \begin{cases} \mathcal{O}(1) & n = 1 \\ \mathcal{O}(1) + T(\lfloor n/2 \rfloor) & \text{sonst} \end{cases}$$

Üblicherweise nur für $n = 2^k$; z.B. durch vollständige Induktion.