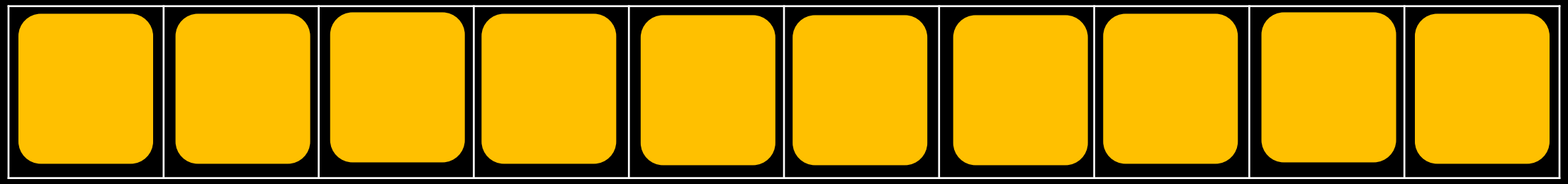
Informatik Q1 Abels

Binäre Suche

Binäre Suche

klein groß



Wo ist die 29 ?

Binäre Suche

klein

groß

23 < 29

29

... nicht an 4. Stelle ...

Binäre Suche

klein

groß

34 > 29

29

... nicht an 7. Stelle ...

Binäre Suche

klein

groß

				23	29		34		

29 = 29

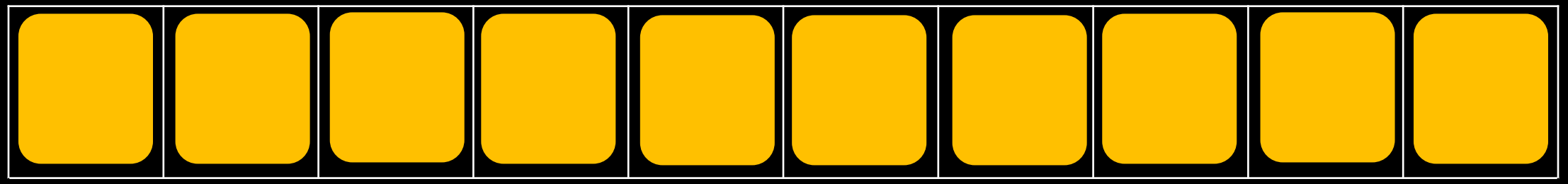
29

... an 5. Stelle!

Binäre Suche

klein

groß



0 1 2 3 4 5 6 7 8 9

links

rechts

Wo ist die 29 ?

Binäre Suche

klein

groß

0 1 2 3 4 5 6 7 8 9

mitte

$23 < 29$

29

... nicht an 4. Stelle ...

Binäre Suche

klein groß

0 1 2 3 4 5 6 7 8 9

34 > 29

... nicht an 7. Stelle ...

29

Binäre Suche

klein groß

				23	29		34		
0	1	2	3	4	5	6	7	8	9

$29 = 29$

... an 5. Stelle!

29

Übung 1

a) Formuliere in deinen eigenen Worten einen Algorithmus, ein Element in einer sortierten Liste nach dieser Strategie zu suchen.

b) Analysiere die Laufzeit des Algorithmus: Wie viele Schritte benötigst du im Worst-Case?

c) Zeichne zu deinem Algorithmus ein Struktogramm.

d) Implementiere deinen Algorithmus. Erstelle dazu ein Programm namens **`BinarySearch`**, in dem das Element `e (int)` in der Liste `liste` gesucht wird. Die Stelle, an der das Element gefunden wurde, soll im Terminal ausgegeben werden.

e) Erweitere dein Programm, sodass auch der Fall berücksichtigt wird, dass das Element nicht in der Liste vorhanden ist.

BinarySearch.py

```python
liste = [-5, 1, 3, 8, 10, 42]
e = 42

index = -1
left = 0
right = len(liste) - 1

# DEIN CODE

if index == -1:
    print("Nicht vorhanden")
else:
    print("Index:", index)
```

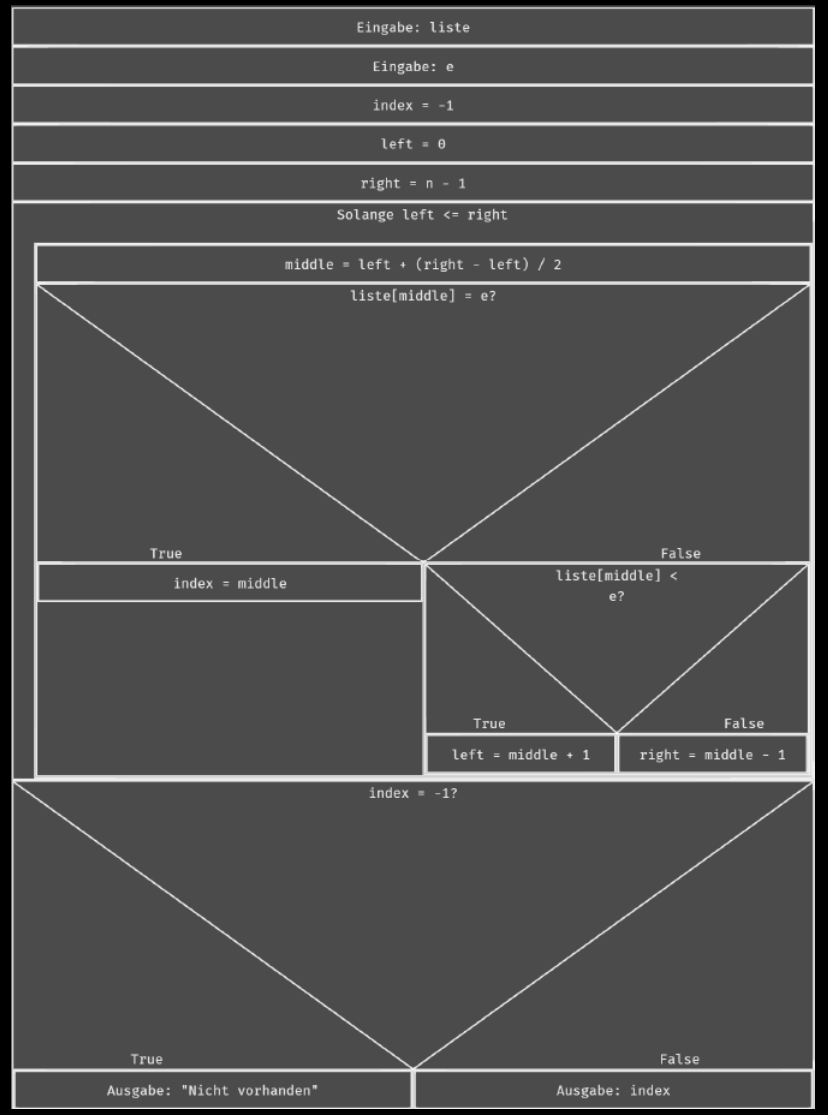
Terminal

```
Index: 5
```

Übung 1

Laufzeit: $\boldsymbol{O(\log n)}$, denn im Worst-Case halbiert der Algorithmus die Liste so oft, bis nur noch 1 Element übrig ist.

$$2^x = n$$
$$x = \log_2 n$$



SuchenUndSortieren.py

```python
liste = [-5, 1, 3, 8, 10, 42]
e = 42

index = -1
left = 0
right = len(liste) - 1
while left <= right:
    middle = (left + right) // 2
    if liste[middle] == e:
        index = middle
    elif liste[middle] < e:
        left = middle + 1
    else:
        right = middle - 1
if index == -1:
    print("Nicht vorhanden")
else:
    print("Index:", index)
```

Tagebucheintrag

Binäre Suche

Wochenübung

Erweitere dein Programm **SuchenUndSortieren** um die Funktion **binary_search** und 2 weitere relevante Testcases.

SuchenUndSortieren.py

```python
def binary_search(liste, e):
    pass

print("Testcase 1")
# ...
```

Terminal

```
Testcase 1
Liste: [-5, 1, 3, 8, 10, 42]
Element: 42
Index: 5
--------------------
Testcase 2
Liste: [-5, 1, 3, 8, 10, 42]
Element: 4
Index: Nicht vorhanden
--------------------
```