

AuD Übung 05

8

1.

iterativ

```
'''
Input: ganze Zahl n >= 0
Output: ganze Zahl Fakultät von n
'''
def factorial_iterative(n):
    result = 1
    for i in range(1, n + 1):
        result *= i
    return result

n = 5
print(f"Iterative: {n}! = {factorial_iterative(n)}")
```

rekursiv

```
'''
Input: ganze Zahl n >= 0
Output: ganze Zahl Fakultät von n
'''
def factorial_recursive(n):
    if n == 0:
        return 1
    else:
        return n * factorial_recursive(n - 1)

n = 5
print(f"Recursive: {n}! = {factorial_recursive(n)}")
```

2.

iterativ

```
def factorial_iterative(n):
    result = 1                # O(1)
    for i in range(1, n + 1): # O(n)
        result *= i           # O(1)
    return result
```

rekursiv

```
def factorial_recursive(n):
    if n == 0:                # O(1)
        return 1              # O(1)
    else:
        return n * factorial_recursive(n - 1) # O(1)
```

Die rekursive Funktion $\in O(1)$

Somit gilt:

$$T(n) = T(n - 1) + O(1)$$

mit $T(0) = O(1)$

$$\begin{aligned} T(n) &= T(n - 1) + O(1) \\ T(n - 1) &= T(n - 2) + O(1) \\ &\vdots \\ T(1) &= T(0) + O(1) \end{aligned}$$

zusammengefasst:

$$\begin{aligned} T(n) &= O(1) + n \cdot O(1) \\ &= O(n) \end{aligned}$$

9

1.

```
def maxTeilsomme_1(arr):
    n = len(arr)
    max_sum = float('-inf')
    for i in range(n):
        current_sum = 0
        for j in range(i, n):
            current_sum += arr[j]
            if current_sum > max_sum:
                max_sum = current_sum
    return max_sum
```

```
sequence = [4, -3, -2, 5, 2, 1, -5, 7]
print("Brute Force Maximum Subarray Sum:", maxTeilsomme_1(sequence))
```

2.

```
def max_crossing_sum(arr, low, mid, high):
    left_sum = float('-inf')
    right_sum = float('-inf')
    sum = 0

    for i in range(mid, low - 1, -1):
        sum += arr[i]
        if sum > left_sum:
            left_sum = sum

    sum = 0
    for i in range(mid + 1, high + 1):
        sum += arr[i]
        if sum > right_sum:
            right_sum = sum

    return left_sum + right_sum

def maxTeilsomme (arr, low, high):
    if low == high:
        return arr[low]

    mid = (low + high) // 2

    left_sum = maxTeilsomme (arr, low, mid)
    right_sum = maxTeilsomme (arr, mid + 1, high)
    crossing_sum = max_crossing_sum(arr, low, mid, high)

    return max(left_sum, right_sum, crossing_sum)

# Example usage
sequence = [4, -3, -2, 5, 2, 1, -5, 7]
print("Divide and Conquer Maximum Subarray Sum:", maxTeilsomme(sequence, 0,
len(sequence) - 1))
```

Aufrufe

[4, -3, -2, 5, 2, 1, -5, 7]

1. Initialer Aufruf:

maxTeilsomme (arr, 0, 7)

Der Mittelpunkt ist 3. Teilen in $[0, 3]$ und $[4, 7]$.

2. Linke Hälfte:

$\text{maxTeilsomme}(\text{arr}, 0, 3)$

Der Mittelpunkt ist 1. Teilen in $[0, 1]$ und $[2, 3]$.

3. Linke Hälfte der linken Hälfte:

$\text{maxTeilsomme}(\text{arr}, 0, 1)$

Der Mittelpunkt ist 0. Teilen in $[0, 0]$ und $[1, 1]$.

- Basisfall: $\text{maxTeilsomme}(\text{arr}, 0, 0) = 4$
- Basisfall: $\text{maxTeilsomme}(\text{arr}, 1, 1) = -3$
- Kreuzende Summe: $\text{max crossing sum}(\text{arr}, 0, 0, 1) = 1$

4. Rechte Hälfte der linken Hälfte:

$\text{maxTeilsomme}(\text{arr}, 2, 3)$

Der Mittelpunkt ist 2. Teilen in $[2, 2]$ und $[3, 3]$.

- Basisfall: $\text{maxTeilsomme}(\text{arr}, 2, 2) = -2$
- Basisfall: $\text{maxTeilsomme}(\text{arr}, 3, 3) = 5$
- Kreuzende Summe: $\text{max crossing sum}(\text{arr}, 2, 2, 3) = 3$

5. Linke Hälfte kombinieren:

$\text{maxTeilsomme}(\text{arr}, 0, 3)$

- Linke Summe: 4
- Rechte Summe: 5
- Kreuzende Summe: $\text{max crossing sum}(\text{arr}, 0, 1, 3) = 4$
- Ergebnis: 5

6. Rechte Hälfte:

$\text{maxTeilsomme}(\text{arr}, 4, 7)$

Der Mittelpunkt ist 5. Teilen in $[4, 5]$ und $[6, 7]$.

7. Linke Hälfte der rechten Hälfte:

$\text{maxTeilsomme}(\text{arr}, 4, 5)$

Der Mittelpunkt ist 4. Teilen in $[4, 4]$ und $[5, 5]$.

- Basisfall: $\text{maxTeilsomme}(\text{arr}, 4, 4) = 2$
- Basisfall: $\text{maxTeilsomme}(\text{arr}, 5, 5) = 1$
- Kreuzende Summe: $\text{max crossing sum}(\text{arr}, 4, 4, 5) = 3$

8. Rechte Hälfte der rechten Hälfte:

$\text{maxTeilsomme}(\text{arr}, 6, 7)$

Der Mittelpunkt ist 6. Teilen in $[6, 6]$ und $[7, 7]$.

- Basisfall: $\text{maxTeilsomme}(\text{arr}, 6, 6) = -5$
- Basisfall: $\text{maxTeilsomme}(\text{arr}, 7, 7) = 7$
- Kreuzende Summe: $\text{max crossing sum}(\text{arr}, 6, 6, 7) = 2$

9. Rechte Hälfte kombinieren:

$\text{maxTeilsomme}(\text{arr}, 4, 7)$

- Linke Summe: 3
- Rechte Summe: 7

- Kreuzende Summe: $\text{max crossing sum}(\text{arr}, 4, 5, 7) = 3$
- Ergebnis: 7

10. **Endgültiges Ergebnis kombinieren:**

$\text{maxTeilsomme}(\text{arr}, 0, 7)$

- Linke Summe: 5
- Rechte Summe: 7
- Kreuzende Summe: $\text{max crossing sum}(\text{arr}, 0, 3, 7) = 10$
- Ergebnis: 10

Somit beträgt die maximale Teilsomme mit beiden Algorithmen 10.