Laufzeitanalyse demogr.

```
int fiboCalc(int n) {
    if (n <= 2) then return 1;

    int result;
    result = fiboCalc(n-1) + fiboCalc(n-2);
    return result;
}
```

Die Laufzeit in Abhängigkeit von n ist

- (1) $\Theta(\log n)$
- (2) $\Theta(n)$
- (3) $\Theta(n^2)$
- (4) $2^{\Theta(n)}$
- (5) $n^{\Theta(n)}$

Laufzeitanalyse demogr.

```
int fiboCalc(int n) {
    if (n <= 2) then return 1;

    int result;
    result = fiboCalc(n-1) + fiboCalc(n-2);
    return result;
}
```

Die Laufzeit in Abhängigkeit von n ist

- (1) $\Theta(\log n)$
- (2) $\Theta(n)$
- (3) $\Theta(n^2)$
- (4) $2^{\Theta(n)}$
- (5) $n^{\Theta(n)}$

Auflösung:

Laufzeitanalyse demogr.

```
int fiboCalc(int n) {
    if (n <= 2) then return 1;

    int result;
    result = fiboCalc(n-1) + fiboCalc(n-2);
    return result;
}
```

Die Laufzeit in Abhängigkeit von n ist

- (1) $\Theta(\log n)$
- (2) $\Theta(n)$
- (3) $\Theta(n^2)$
- (4) $2^{\Theta(n)}$
- (5) $n^{\Theta(n)}$

Auflösung: (4) $2^{\Theta(n)}$

Fibonacci demogr.

Welches ist die schnellste Laufzeit, in der man die ersten n Zahlen der Fibonacci-Folge berechnen kann?

- (1) $\Theta(n)$
- (2) $\Theta(n \log n)$
- (3) $\Theta(n^2)$
- (4) $\Theta(n^{\log n})$
- (5) $2^{\Theta(n)}$

Fibonacci demogr.

Welches ist die schnellste Laufzeit, in der man die ersten n Zahlen der Fibonacci-Folge berechnen kann?

- (1) $\Theta(n)$
- (2) $\Theta(n \log n)$
- (3) $\Theta(n^2)$
- (4) $\Theta(n^{\log n})$
- (5) $2^{\Theta(n)}$

Auflösung:

Fibonacci demogr.

Welches ist die schnellste Laufzeit, in der man die ersten n Zahlen der Fibonacci-Folge berechnen kann?

- (1) $\Theta(n)$
- (2) $\Theta(n \log n)$
- (3) $\Theta(n^2)$
- (4) $\Theta(n^{\log n})$
- (5) $2^{\Theta(n)}$

Auflösung: (1) $\Theta(n)$ mit dynamischer Programmierung.

Dynamische Programmierung demogr.

Bei allgemeiner dynamischer Programmierung ist der unterliegende Aufrufgraph kein Baum (sondern ein DAG). Deshalb ist der benötigte Speicherplatz immer superlinear in der Länge der Eingabe.

- (1) Stimmt.
- (2) Stimmt nicht.
- (3) Keine Ahnung.

Dynamische Programmierung demogr.

Bei allgemeiner dynamischer Programmierung ist der unterliegende Aufrufgraph kein Baum (sondern ein DAG). Deshalb ist der benötigte Speicherplatz immer superlinear in der Länge der Eingabe.

- (1) Stimmt.
- (2) Stimmt nicht.
- (3) Keine Ahnung.

Auflösung:

Dynamische Programmierung demogr.

Bei allgemeiner dynamischer Programmierung ist der unterliegende Aufrufgraph kein Baum (sondern ein DAG). Deshalb ist der benötigte Speicherplatz immer superlinear in der Länge der Eingabe.

- (1) Stimmt.
- (2) Stimmt nicht.
- (3) Keine Ahnung.

Auflösung: (2)

Dynamische Programmierung demogr.

Bei allgemeiner dynamischer Programmierung ist der unterliegende Aufrufgraph kein Baum (sondern ein DAG). Deshalb ist der benötigte Speicherplatz immer superlinear in der Länge der Eingabe.

- (1) Stimmt.
- (2) Stimmt nicht.
- (3) Keine Ahnung.

Auflösung: (2)

Bsp: $O(n)$ Platz für gewichtetes Intervall-Scheduling.

Gewichtetes Intervall-Scheduling (1) demogr.

Welche Laufzeit hat die dynamische Programmierung für gewichtetes Intervall-Scheduling?

- (1) $\Theta(n)$
- (2) $\Theta(n \log n)$
- (3) $\Theta(n^2)$
- (4) $\Theta(n^2 \cdot 2^n)$
- (5) $\Theta(n!)$

Gewichtetes Intervall-Scheduling (1) demogr.

Welche Laufzeit hat die dynamische Programmierung für gewichtetes Intervall-Scheduling?

- (1) $\Theta(n)$
- (2) $\Theta(n \log n)$
- (3) $\Theta(n^2)$
- (4) $\Theta(n^2 \cdot 2^n)$
- (5) $\Theta(n!)$

Auflösung:

Gewichtetes Intervall-Scheduling (1) demogr.

Welche Laufzeit hat die dynamische Programmierung für gewichtetes Intervall-Scheduling?

- (1) $\Theta(n)$
- (2) $\Theta(n \log n)$
- (3) $\Theta(n^2)$
- (4) $\Theta(n^2 \cdot 2^n)$
- (5) $\Theta(n!)$

Auflösung: (2) $\Theta(n \log n)$

Gewichtetes Intervall-Scheduling (2) demogr.

Betrachte eine beliebige Instanz mit n Aufgaben. Sei

- g = Anzahl Aufgaben, die von Greedy ausgewählt werden.
- d = Anzahl Aufgaben, die von dyn. Program. ausgewählt werden.

Wie groß kann g/d werden?

- (1) $\Theta(1)$
- (2) $\Theta(\sqrt{n})$
- (3) $\Theta(n)$
- (4) $\Theta(n \log n)$

Gewichtetes Intervall-Scheduling (2) demogr.

Betrachte eine beliebige Instanz mit n Aufgaben. Sei

- g = Anzahl Aufgaben, die von Greedy ausgewählt werden.
- d = Anzahl Aufgaben, die von dyn. Program. ausgewählt werden.

Wie groß kann g/d werden?

- (1) $\Theta(1)$
- (2) $\Theta(\sqrt{n})$
- (3) $\Theta(n)$
- (4) $\Theta(n \log n)$

Auflösung:

Gewichtetes Intervall-Scheduling (2) demogr.

Betrachte eine beliebige Instanz mit n Aufgaben. Sei

- g = Anzahl Aufgaben, die von Greedy ausgewählt werden.
- d = Anzahl Aufgaben, die von dyn. Program. ausgewählt werden.

Wie groß kann g/d werden?

- (1) $\Theta(1)$
- (2) $\Theta(\sqrt{n})$
- (3) $\Theta(n)$
- (4) $\Theta(n \log n)$

Auflösung: (3) $\Theta(n)$