1 Perrin-Folge

Die Zahlen der Perrin-Folge $P : \mathbb{N} \to \mathbb{N}$ sind wie folgt definiert:

$$P(n) = \begin{cases} 3 & \text{wenn } n = 0 \\ 0 & \text{wenn } n = 1 \\ 2 & \text{wenn } n = 2 \\ P(n-2) + P(n-3) & \text{sonst} \end{cases}$$

1. Entwerfen Sie einen **rekursiven** Algorithmus zur Berechnung der n-ten Zahl der Perrin-Folge. Geben Sie den Algorithmus in Pascal, C, C++, Java oder in Pseudocode an.

2. Bestimmen Sie die Rekurrenzgleichung für die maximale Laufzeit $T(n)$ von Ihrem rekursiven Algorithmus. Gehen Sie von einem uniformen Kostenmaß aus (jede Operation hat Kosten 1).

3. Beweisen Sie durch **vollständige Induktion**, dass $T(n) \leq 2^n$. Kommentieren Sie Ihr Vorgehen.

4. Geben Sie einen **iterativen** Algorithmus zur Berechnung der n-ten Zahl der Perrin-Folge an. Geben Sie den Algorithmus in Pascal, C, C++, Java oder in Pseudocode an.

5. Geben Sie die Laufzeit- und Speicherkomplexität von Ihrem iterativen Algorithmus an. Begründen Sie, wie Sie die Laufzeitkomplexität bestimmt haben.

Hinweise:

1. Beim Entwurf eines Algorithmus muss neben dem Code auch die Schnittstelle definiert werden. Was sind Eingaben, Ausgaben und Randbedingungen?

2. Die Rekurrenzgleichung für die Laufzeit setzt sich zusammen aus der Laufzeit für die Basisfälle und den allgemeinen Fall. Der allgemeine Fall enthält rekursive Aufrufe mit kleineren Eingaben.

3. Für den Beweis, dass $T(n) \leq 2^n$, brauchen Sie die Rekurrenzgleichung.