Weihnachtsaufgabe

Gegeben sei folgender Algorithmus. Er findet (für eine Zweierpotenz n) in einer n-elementigen Menge von gleichwertigen Elementen ein einzelnes mit höherem Wert.

Die Werte seinen in einem Array $A[1...n]$ gegeben für ein $n = 2^p$. Für die Eingabe sei garantiert, dass es ein k gibt mit $A[i] < A[k]$ für alle $i \neq k$ und $A[i] = A[j]$ für alle $i, j \neq k$.

Die Methode `weight`$(A[i, j])$ gibt dabei in $\mathcal{O}(1)$ die Summe aller Werte zwischen den Stellen i und j aus.

Algorithm 1: SockSearch

Aufruf: `sockSearch`$(A, 1, n)$

```
sockSearch(A, l, r) begin
    while l ≠ r do
        m ← (l+r−1)/2;
        if weight(A[l, m]) < weight(A[m + 1, r]) then
            l ← m + 1;
        else
            r ← m;
    return "gefunden an Stelle l";
```

1. Geben Sie seine Laufzeit in $\mathcal{O}$-Notation an. Begründen Sie!

2. Beweisen Sie die Korrektheit des Algorithmus mit Hilfe der Verifikation nach Floyd.