Institut für Theoretische Informatik
T.T.-Prof. Dr. Thomas Bläsius
https://scale.iti.kit.edu

Übungsblatt 04

Algorithmen I - Sommersemester 2022

Abgabe im ILIAS bis 18.05.2022, 14:00 Uhr

Bitte beschrifte Deine Abgabe gut sichtbar mit Deinem Namen und Deiner Matrikelnummer. Achte insbesondere bei handschriftlichen Abgaben auf Lesbarkeit und genügend Platz für Korrektur-Anmerkungen. Die Abgabe erfolgt über das Übungsmodul in der Gruppe Deines Tutoriums im ILIAS. Gib Deine Ausarbeitungen in *einer* PDF-Datei ab.

Aufgabe 1 - Quo vadis? (8 Punkte)

Betrachte die folgenden drei Algorithmen:

```
PRIMUS(A : [ℕ, n], k : ℕ): ℕ
    c : ℕ = 0
    for i ∈ {0, ..., n − 1} do
        for j ∈ {i + 1, ..., n − 1} do
            if A[i] + A[j] = k then
                c := c + 1
            end
        end
    end
    return c
```

```
SECUNDUS(A : [ℕ; n], ℓ : ℕ = 0, r : ℕ = n − 1)
    if ℓ < r then
        key : ℕ = A[ℓ]
        A[ℓ] = A[r]
        A[r] = key
        SECUNDUS(A, ℓ + 1, r − 1)
    end
```

```
TERTIUS(A : [ℕ; n])
    for i ∈ {0, ..., n − 1} do
        if A[i] ≠ 0 then
            a : ℕ = A[i]
            A[i] := 0
            b : ℕ = TERTIUS(A)
            if a > b ∧ i < n − 1 then
                return b
            end
            return a
        end
    end
    return 0
```

1. Gegeben sei das folgende Array: $A = \langle 6, 42, 3, 35, 7, 14, 5, 10, 4, 5 \rangle$
   Gib für jeden der drei obigen Algorithmen den Inhalt von A sowie falls vorhanden den Rückgabewert an, nachdem der Algorithmus darauf angewendet wurde. Verwende für PRIMUS den Parameter $k = 20$. (3 Punkte)

2. Beschreibe für jeden der drei Algorithmen, was er leistet. Bestimme und begründe seine asymptotische Laufzeit. (5 Punkte)

Aufgabe 2 - SummitSort (6 Punkte)

Wir bezeichnen ein Array $A : [\mathbb{N};\ n]$ als *summit-sortiert*, wenn für jedes Paar von Indizes $i, j \in \{0, \dots, n-1\}$ folgendes gilt:

- Wenn $i \leq \lfloor \frac{n}{2} \rfloor \wedge j \leq \lfloor \frac{n}{2} \rfloor$, dann ist $A[i] \geq A[j]$, wenn $i \geq j$
- Wenn $i > \lfloor \frac{n}{2} \rfloor \wedge j > \lfloor \frac{n}{2} \rfloor$, dann ist $A[i] \geq A[j]$, wenn $i \leq j$

1. Füge die beiden folgenden summit-sortierten Arrays A_1, A_2 zu einem summit-sortierten Array zusammen und gib das Resultat an. (2 Punkte)

   $$A_1 = \langle 1, 9, 24, 12, 9 \rangle, \quad A_2 = \langle 3, 12, 18, 18, 21, 5 \rangle$$

   Entscheide und begründe, ob dein Resultat die einzige gültige Lösung ist.

2. Wir wollen SUMMITSORT mit Hilfe von MERGESORT aus der Vorlesung entwickeln. Der Algorithmus SUMMITSORT soll statt MERGE die Hilfsroutine SUMMITMERGE aufrufen, sonst jedoch identisch zu MERGESORT sein und die gleiche asymptotische Laufzeit haben. Beschreibe SUMMITMERGE und beweise dessen Korrektheit.
   Hinweis: In der Vorlesung wurde die Korrektheit von MERGE mit Hilfe einer Invariante gezeigt. (3 Punkte)

3. Begründe, warum SUMMITSORT mit deinem SUMMITMERGE aus Teilaufgabe 2 die gleiche asymptotische Laufzeit hat wie MERGESORT. (1 Punkt)

Aufgabe 3 - Gesucht, gefunden (5+4 Punkte)

Gegeben sei ein unsortiertes Array $A : [\mathbb{N};\ n]$. Wir wollen das k-kleinste Element in A bestimmen.

1. Beschreibe einen Algorithmus, der für ein ein gegebenes Element $A[i]$ mit $i \in \{0, n-1\}$ den Index bestimmt, an dem $A[i]$ stehen würde, wenn A sortiert wäre. Nenne und begründe die Laufzeit deines Algorithmus. (1 Punkt)

2. Entscheide und begründe, welche Werte die Konstante $b \in \mathbb{R}_+$ annehmen kann, sodass für die folgende Rekurrenz $T(n) \in O(n)$ gilt. (1 Punkt)

$$T(n) = \begin{cases} \Theta(1) & \mid n = 1 \\ T(\frac{n}{b}) + \Theta(n) & \mid \text{sonst} \end{cases}$$

3. Beschreibe einen Algorithmus, welcher in $O(n)$ das k-kleinste Element eines unsortierten Arrays A der Kapazität n bestimmt. Dein Algorithmus darf dabei die Reihenfolge der Einträge in A vertauschen. Du darfst annehmen, dass du den Median eines unsortierten Arrays in $\Theta(n)$ bestimmen kannst. Begründe, warum dein Algorithmus die geforderte Laufzeit hat. (3 Punkte)

$*$ Wir nehmen nun nicht mehr an, den Median eines unsortierten Arrays in $\Theta(n)$ bestimmen zu können. Gib einen randomisierten Algorithmus an, welcher das k-kleinste Element von A in erwartet $O(n)$ findet. Begründe die Korrektheit des Algorithmus und der Laufzeit. (4 Punkte)

Hinweis: Teilaufgaben, die nicht mit einer Nummer, sondern mit $*$ versehen sind, sind Zusatzaufgaben. Hier kannst du dir Bonuspunkte verdienen.