Lehrstuhl für Informatik I
Algorithmen und Komplexität
Universität Würzburg

Würzburg, den 10. Dezember 2020

Prof. Dr. Alexander Wolff
Felix Klesen, M. Sc.

1. Kurztest zur Vorlesung Algorithmen und Datenstrukturen (Wintersemester 2020/2021)

Bitte beachten Sie die folgenden Hinweise!

1. **Personalien:** Bitte tragen Sie Ihre Daten gut lesbar ein.

Name	
Vorname	
Matrikelnummer	
Übungsgruppe	

2. **Papier:** Tragen Sie Ihre Lösungen direkt unterhalb der Aufgaben ein. Verwenden Sie kein eigenes Papier.
3. Sie dürfen ein einseitig handgeschriebenes DIN-A4-Blatt mit Notizen verwenden. **Weitere Hilfsmittel sind nicht zugelassen.**
4. Schreiben Sie **nicht** mit Bleistift oder einem roten Stift.
5. Die Bearbeitungszeit beträgt 60 Minuten.

Wir wünschen Ihnen viel Erfolg!

Aufgabe	1	2	3	4	5	6	**Gesamt**
Punkte	8	13	12	8	13	6	60
Ergebnis							

Aufgabe 1 — Laufzeitanalyse

Gegeben seien eine Methode $f(n)$ mit Laufzeit $\Theta(\log n)$ und eine Methode $g(n)$ mit Laufzeit $\Theta(n)$. Im Folgenden sind vier Schleifen gegeben, in deren Rumpf je eine dieser Funktionen aufgerufen wird. Bestimmen Sie jeweils die Gesamtlaufzeit der Schleife in Abhängigkeit von n und begründen Sie Ihre Antwort.

(a) **for** int $i = n$ **downto** 1 **do**
  $f(n)$

/ 2 P.

Laufzeit: $\Theta($__________$)$, denn:

(b) **for** int $i = 1$ **to** n **do**
  $g(i)$

/ 2 P.

Laufzeit: $\Theta($__________$)$, denn:

(c) int $i = n$
while $i > 1$ **do**
  $f(n); i = i/2$

/ 2 P.

Laufzeit: $\Theta($__________$)$, denn:

(d) int $i = 1$
while $i < n$ **do**
  $g(i); i = i \cdot 2$

/ 2 P.

Laufzeit: $\Theta($__________$)$, denn:

Aufgabe 2 — O-Notation

(a) Ordnen Sie die folgenden Funktionen nach ihrem asymptotischen Wachstum. Sie sollen Sie also in eine Reihenfolge $f_1, f_2, \ldots, f_6$ bringen, so dass $O(f_1) \subseteq O(f_2) \subseteq \cdots \subseteq O(f_6)$ gilt. **/ 5 P.**

$$5^n n^2 \qquad n(\log_3 n)^2 \qquad n\sqrt{n}$$

$$5n + \sqrt[50]{n} \qquad n + n^{1{,}03} \qquad n!/n$$

(b) In der folgenden Menge von Funktionen sind *vier Paare* von Funktionen enthalten, die asymptotisch gleich schnell wachsen, formaler ausgedrückt also Paare (f, g) von Funktionen, für die $O(f) = O(g)$ gilt. Verbinden Sie die jeweils zusammengehörenden Funktionen mit einer Linie. **/ 8 P.**

Wenn Sie mehr als 4 Linien einzeichnen, wird die Lösung nicht gewertet.

$3^{n+\log_2 n}$

$n^2 \log_2(n^2)$ $\qquad$ $n^2(\log_2 n)^2$

$n \log_2(n^5)$ $\qquad$ $n + \log_2 n$

$(n+3)^{5/2} + 2n$ $\qquad$ $50n \log_2 n$

$n^2\sqrt{n}$ $\qquad$ $n \cdot 2^n + n^5$

$(\sqrt{25})^{\log_5 n}$ $\qquad$ $2^n + 2^{\log_2 n}$

$n \cdot 3^n + n^2$

Aufgabe 3 — Sortieralgorithmen

Gegeben sei folgender Sortieralgorithmus in Pseudocode, wobei die Methode `Merge` aus `MergeSort` übernommen wurde.

```
SomeSort(int[] A)
  k = 1
  while k < A.length do
    Merge(A, 1, k, k + 1)
    k = k + 1
```

(a) Führen Sie zunächst `SomeSort` auf der folgenden Eingabe A aus. Verwenden Sie dabei je eines der vorgedruckten Felder, um den Zustand von A *nach* jedem Durchlauf der while-Schleife darzustellen. / 4 P.

Eingabe:	$A =$	5	2	1	3	4
1. Durchlauf:	$A =$					
2. Durchlauf:	$A =$					
3. Durchlauf:	$A =$					
4. Durchlauf:	$A =$					

(b) Geben Sie die maximale und die minimale Laufzeit, die `SomeSort` auf einer Eingabe A der Länge $n = A.\mathrm{length}$ benötigt, in Abhängigkeit von n an. / 2 P.

- Worst-Case: Θ(__________)
- Best-Case: Θ(__________)

(c) Beweisen Sie die Korrektheit des Algorithmus mit folgender Schleifeninvarianten: / 5 P.
Bei der k*-ten Ausführung des while-Schleifenkopfes gilt folgendes:*

(i) $A[1..k]$ *enthält dieselben Elemente wie zu Beginn der Ausführung des Algorithmus – jedoch sortiert.*

(ii) $A[(k+1)..A.\mathtt{length}]$ *hat sich seit der Ausführung des Algorithmus nicht verändert.*

Initialisierung:

Aufrechterhaltung:

Terminierung:

(d) Welchem Algorithmus, den Sie aus der Vorlesung kennen, ähnelt `SomeSort`? Begründen Sie Ihre Antwort. / 1 P.

Aufgabe 4 — Heaps

(a) Ist die Folge $B = \langle 24, 20, 22, 11, 17, 14, 9, 5, 12, 13, 15 \rangle$ ein Max-Heap? Begründen Sie Ihre Antwort. / 1 P.

(b) Gegeben ist ein Feld B der Länge n. Geben Sie einen Algorithmus *in Pseudocode* an, der *in Linearzeit* testet, ob B die Max-Heap-Eigenschaft erfüllt. / 5 P.

(c) Welcher Algorithmus aus der Vorlesung modifiziert ein beliebiges gegebenes Feld D, so dass es die Max-Heap-Eigenschaft in jedem Fall erfüllt? / 2 P.

Geben Sie die asymptotische Worst-Case-Laufzeit dieses Algorithmus an: $\Theta($__________$)$

Aufgabe 5 — Rekursionsgleichungen

Lösen Sie folgende Rekursionsgleichungen mittels der jeweils angegebenen Verfahren.

(a) Sei / 4 P.

$$T(n) = \begin{cases} 8T(\lceil n/2 \rceil) + n^2 \cdot \log_2 n & \text{falls } n > 1 \\ 1 & \text{sonst} \end{cases}$$

$T \in \Theta($__________$)$, denn mittels **Meister-Methode** (Fall ____) gilt

(b) Sei / 5 P.

$$T(n) = \begin{cases} 4T(\lfloor n/4 \rfloor) + n^3 & \text{falls } n > 1 \\ 1 & \text{sonst} \end{cases}$$

Zeigen Sie mit Hilfe der **Rekursionsbaum-Methode**, dass $T \in \Theta(n^3)$ gilt.

(c) Beweisen Sie mittels *vollständiger Induktion*, dass für

/ 4 P.

$$T(n) = \begin{cases} T(n-1) + \lfloor n/3 \rfloor & \text{falls } n > 1 \\ 1 & \text{sonst} \end{cases}$$

gilt: $T \in O(n^2)$.
Behauptung: Es gibt ein $c > 0$, so dass für alle $n \geq 1$ gilt $T(n) \leq cn^2$.
Induktionsanfang:

Induktionsvoraussetzung:

Induktionsschritt:

Aufgabe 6 — Teile & Herrsche

Gegeben sei der folgende Algorithmus:

```
AlgoX(int[] A, int ℓ = 1, int r = A.length)
  if ℓ < r then
    m = ⌊(ℓ + r)/2⌋
    AlgoX(A, ℓ, m)
    AlgoX(A, m + 1, r)
    if A[ℓ] ≤ A[m + 1] then
      Swap(A, ℓ, m + 1)                // Vertausche A[ℓ] und A[m + 1]
```

(a) Stellen Sie die *genaue* Rekursionsgleichung für die Anzahl von Vergleichen von Elementen der Eingabe auf, die der Aufruf von AlgoX(A, ℓ, r) auf einem (Teil-)Feld $A[\ell..r]$ der Länge $n = r - \ell + 1$ ausführt. Vergessen Sie den Basisfall nicht. / 2 P.

$T(n) =$

(b) Nach einem Aufruf von AlgoX(A, ℓ, r) befindet sich das Maximum des (Teil-)Feldes $A[\ell..r]$ in $A[\ell]$. Beweisen Sie dies mittels vollständiger Induktion. / 4 P.

Induktionsanfang:

Induktionsvoraussetzung:

Induktionsschritt: