Prof. Dr. Ir. Joost-Pieter Katoen

Christian Dehnert, Jonathan Heinen, Thomas Ströder, Sabrina von Styp

Hinweise:

- Die Übungsblätter sind in Gruppen von je 3 Studierenden aus der gleichen Kleingruppenübung zu bearbeiten.
- Die Lösungen müssen bis Montag, den 14. Mai um 11:00 Uhr in den entsprechenden Übungskasten eingeworfen werden. Sie finden die Kästen am Eingang Halifaxstr. des Informatikzentrums (Ahornstr. 55).
- Namen und Matrikelnummern der Studenten sowie die Nummer der Übungsgruppe sind auf jedes Blatt der Abgabe zu schreiben. Heften bzw. tackern Sie die Blätter!
- Am Freitag den 11. Mai, sowie am Dienstag den 15.Mai findet keine Vorlesung statt.

Aufgabe 1 (Eigenschaften von Heaps): (1 + 5 Punkte)

Beweisen Sie die folgenden, aus der Vorlesung bekannten, Lemmata:

a) Ein n-elementiger Heap hat die Höhe $\lfloor \log_2 n \rfloor$.

b) Ein Heap hat maximal $\lceil n/2^{h+1} \rceil$ Knoten mit der Höhe h.

Hinweis: Die Höhe h des Knotens entspricht der Länge des längsten Pfades zu einem Blatt des Heaps.

Aufgabe 2 (Heapsort): (2 + 2 + 4 Punkte)

a) In der Vorlesung wurden so genannte Max-Heaps vorgestellt. D.h. jedes Element ist größer/gleich seiner Kinder. Analog hierzu ist ein *Min-Heap* ein Heap, bei dem jedes Element kleiner/gleich seiner Kinder ist. Bestimmen sie, ob die folgenden Arrays Max-Heaps oder Min-Heaps sind. Für den Fall, dass es sich weder um den einen noch den anderen handelt, geben sie an, wo die Heapeigenschaft verletzt ist.

1.)

37	21	32	17	10	24	22	1	9	15	14

2.)

24	21	32	17	10	37	22	1	9	15	14

3.)

10	29	15	20	30	21	16	40	31	41	35

4.)

10	20	15	29	31	21	16	40	30	41	35

b) Stellen Sie das folgende Array als Baum dar. Skizzieren Sie (durch nummerierte Pfeile) die, zur Herstellung der Max-Heap Eigenschaft nötigen Versickerungen. Geben Sie den resultierenden Heap sowohl als Baum als auch als Array an.

9	2	1	4	3	2	1	5	7	1

c) Sortieren Sie das in Aufgabenteil **b)** gegebene Array mit Hilfe von Heapsort. Geben Sie für jeden Sortierschritt den neuentstandenen Heap als Array sowie als Baum an und skizzieren Sie nötige Versickerungen in der Baumdarstellung. Geben Sie auch die Reihenfolge der Versickerungen an.

Aufgabe 3 (Rekursiver Sortieralgorithmus): (2 + 3 Punkte)

```
sort(int [ ] A, int l, int r) {
  if (A[l] > A[r]){
    exchange(A[l], A[r]);
  }
  if (l < r-1){
    k:= (r-l+1) div 3;
    sort(A, l, r-k);
    sort(A, l+k, r);
    sort(A, l, r-k);
  }
}
```

a) Bestimmen Sie für den gegebenen Sortieralgorithmus die Komplexitätsklasse Θ im Best-, Worst- und Average-Case für den Aufruf `sort(A,0,n-1)` in Abhängigkeit von `n`, der Anzahl der zu sortierenden Elemente aus dem Array `A`.

b) Der vorgestellte Algorithmus ist nicht stabil. Ändern Sie den Algorithmus so ab, dass er stabil wird.

Aufgabe 4 (Rekursionsgleichungen): (2 + 2 + 4 + 3 Punkte)

Geben Sie für die folgenden Rekursionsgleichungen die Komplexitätsklasse (Θ) an. Begründen Sie Ihre Antwort.

a) $T(n) = 8T(\frac{n}{4}) + 2n \cdot \sqrt{n} + 10n$

b) $T(n) = 5T(\frac{n}{4}) + n \cdot \log_2 n$

c) $T(n) = 4T(\sqrt[5]{n}) + (\log_2 n)^7$

d) $T(n) = T(\frac{n}{2}) + 2T(\frac{n}{4}) + n$