Algorithmen und Datenstrukturen
Übungszettel 2

Martin Avanzini <martin.avanzini@uibk.ac.at>
Thomas Bauereiß <thomas.bauereiss@uibk.ac.at>
Herbert Jordan <herbert@dps.uibk.ac.at>
René Thiemann <rene.thiemann@uibk.ac.at>

22. März, zur Besprechung am 29. März

Aufgabe 1) Suchverfahren In der Vorlesung wurden zwei Suchverfahren zur Ermittlung der Position eines Elements innerhalb einer sortierten Liste vorgestellt.

1. Vervollständigen Sie das Beispiel *Search.java* indem Sie die Binäre und Interpolations-Suche innerhalb der entsprechenden Methoden implementieren.
2. Erstellen Sie je ein Best-Case / Worst-Case Szenario für die lineare, binäre und Interpolations-Suche der Länge n=8. Geben Sie jeweils die Worst-Case Laufzeitkomplexitäten an.

Aufgabe 2) Heap Sort

1. Sei die Höhe eines Heaps durch die maximale Tiefe eines Blattes definiert. Die Tiefe der Wurzel sei dabei 1. Jeder übrige Knoten liegt genau um 1 tiefer als sein Elternknoten. Ein leerer Heap hat die Höhe 0. Wie viele Knoten enthält ein Heap der Höhe n maximal? Beweisen Sie die Gültigkeit Ihrer Aussage mittels Induktion.
2. Sortieren Sie die Zahlen 5, 3, 17, 10, 19, 6, 22 mittels des Heap-Sort Verfahrens. Illustrieren Sie dabei Zwischenschritte jeweils als Baum und Array.

Aufgabe 3) Beweisen von Programmen: Schleifeninvariante Gegeben der folgende Algorithmus zum Sortieren natürlicher Zahlen (MinSort):

Beispiel Implementierung von MinSort

```
1  public static void minSort(int data[]) {
2      int i = 0;
3      while (i < data.length) {
4          int pos = i;
5          int k = i + 1;
6          while (k < data.length) {
7              if (data[k] < data[pos]) {
8                  pos = k;
9              }
10             k++;
11         }
12
13         int tmp = data[i];
14         data[i] = data[pos];
15         data[pos] = tmp;
16
17         i++;
18     }
19 }
```

1. Beschreiben Sie die dem Algorithmus zugrunde liegende Strategie.
2. Finden Sie eine Invariante der inneren Schleife und zeigen Sie das die Zeilen 4-11 den Index eines kleinsten Elements zwischen dem i-ten und letzten Element ermitteln.
3. Finden Sie eine Invariante der äußeren Schleife und zeigen Sie, dass die Methode die Eingabedaten tatsächlich sortiert.

Aufgabe 4) Teilmengen Am Ende ein Beispiel aus der Praxis: Zwei Module eines Programms liefern je eine Menge von Daten-Element Indizes in der Form eines Integer-Arrays (ohne Duplikate, unsortiert). Seien die Arrays A und B diese beiden Mengen.

1. Beschreiben Sie einen (effizienten) Algorithmus zum Testen ob $A = B$
2. Beschreiben Sie einen (effizienten) Algorithmus zum Testen ob $A \subseteq B$

Begründen Sie das Design Ihrer Algorithmen und geben sie deren Komplexitätsklassen an.