Datenstrukturen und Algorithmen
Übung 1

21. Februar 2012

Diese Übung muss zu Beginn der Übungsstunde bis spätestens um 14 Uhr 15 am 28. Februar abgegeben werden. Die Abgabe der DA Übungen erfolgt immer in schriftlicher Form auf Papier. Programme müssen zusammen mit der von ihnen erzeugten Ausgabe abgegeben werden. Drucken Sie wenn möglich platzsparend 2 Seiten auf eine A4-Seite aus. Falls Sie mehrere Blätter abgeben heften Sie diese bitte zusammen (Büroklammer, Bostitch, Mäppchen).

Alle Teilnehmenden dieser Vorlesung müssen Ihre eigene Übung abgeben (einzeln oder in Zweiergruppen). Jede Übungsserie gibt 10 Punkte. Im Durchschnitt müssen Sie 7 von 10 Punkten erreichen, um die Testatbedingung zu erfüllen. Vergessen Sie nicht, Ihren Namen und Ihre Matrikelnummer auf Ihrer Abgabe zu vermerken.

Grundlagen: Summen, Produkte und Induktion

1. Leiten Sie eine einfache Formel für $\sum_{k=1}^{n}(2k-1)$ ab. **(1 Punkt)**
2. Werten Sie das Produkt $\Pi_{k=1}^{n} 2 \cdot 4^k$ aus. **(1 Punkt)**
3. Zeigen Sie mit mit mathematischer Induktion, dass die Reihe $\sum_{k=1}^{n} \frac{1}{k^2}$ nach oben durch eine Konstante beschränkt ist. (Tip: Benutzen Sie dazu $\sum_{k=1}^{n} \frac{1}{k^2} \leq 1 + 1 - \frac{1}{n}$.) **(1 Punkt)**

Theoretische Aufgaben

1. Illustrieren Sie den Ablauf von Sortieren durch Einfügen (*Insertion Sort*) anhand einer Skizze ähnlich wie in der Vorlesung gezeigt. Verwenden Sie die Eingabe $\langle 3, 15, 6, 1, 2, 11, 16, 2, 8, 4\rangle$. **(1 Punkt)**
2. Illustrieren Sie den Ablauf von Sortieren durch Mischen (*Merge Sort*) äehnlich wie in Figure 2.4 im Buch. Verwenden Sie dieselbe Eingabe wie oben. **(1 Punkt)**
3. Wir betrachten das folgende *Suchproblem*:

   Eingabe: Ein Feld von Zahlen $A = \langle a_1, a_2, \ldots a_n\rangle$ und ein Wert v.
   Ausgabe: Ein Index i so dass $v = A[i]$ oder den speziellen Wert *nil* falls v nicht in A vorkommt.

Schreiben Sie Pseudocode, welcher das Problem löst, indem das Feld Element um Element durchlaufen wird, bis v gefunden oder das Ende des Felds erreicht wird. Zeigen Sie, dass Ihr Algorithmus korrekt ist, indem Sie eine Schleifeninvariante beweisen. Zeigen Sie alle drei benötigten Kriterien. Was ist die Worst Case Laufzeit dieses Algorithmus? **(1 Punkt)**

4. Das Suchproblem oben kann effizienter gelöst werden, wenn das Eingabefeld sortiert ist. In diesem Fall kann das Element in der Mitte des Feldes mit v verglichen werden, und die eine Hälfte des Feldes kann von der Suche ausgeschlossen werden. *Binäre Suche* ist ein Algorithmus, der diese Idee wiederholt anwendet indem die verbleibende Hälfte des Feldes jeweils wieder halbiert wird. Schreiben Sie Pseudocode für binäre Suche, der iterativ (nicht rekursiv) abläuft. Zeigen Sie, dass Ihr Algorithmus korrekt ist, indem Sie eine Schleifeninvariante beweisen. Erklären Sie, warum die Worst Case Lauftzeit des Verfahrens $\Theta(\log n)$ ist. **(1 Punkt)**

Praktische Aufgaben

1. In der Vorlesung wurde für Insertion Sort die Zeitkomplexität $O(n^2)$ und für Merge Sort $O(n \log n)$ angegeben. Bestätigen Sie diese Grössenordnungen experimentell. Sie können Java Code für beide Algorithmen von der Vorlesungs-Webpage kopieren.

   Schreiben Sie ein Java Programm zur Durchführung des Experiments. Erzeugen Sie zufällige Eingabefelder verschiedener Grösse und messen Sie jeweils die Zeit, die zum Sortieren nötig ist. Zur Zeitmessung können Sie die Klasse `Timer` verwenden, die wir zur Verfügung stellen. Beachten Sie, dass die Messung von kurzen Zeitspannen relativ ungenau sein kann.

   Als Abgabe erwarten wir den Ausdruck Ihres Testprogramms. Weiter sollen Sie eine Grafik erstellen, welche die Zeitmessungen für beide Algorithmen und verschieden grosse Felder zusammenfasst. Erwähnen Sie auch die Spezifikationen des Computers, auf denen das Experiment durchgeführt wurde. Erläutern Sie, ob Ihre Messungen der Theorie entsprechen.

   Schätzen Sie zudem auf Grund Ihrer Messungen die Laufzeit von Insertion Sort bei der Eingabe von $10'000'000$ Zahlen ab.

   Hinweise: Konsultieren Sie die Java Dokumentation, um herauszufinden wie Zufallszahlen erzeugt werden können. Sie können die Grafik von Hand zeichnen oder ein Softwareprogramm wie z.B. Microsoft Excel verwenden. **(3 Punkte)**