RWTH Aachen **SS 2015**
Lehrgebiet Theoretische Informatik Blatt 1
Rossmanith–Reidl–Sanchez 15. März 2015

Übung zur Vorlesung Algorithmen und Datenstrukturen

Aufgabe T1

Ordnen Sie die folgenden Funktionen in der Reihenfolge ihres asymptotischen Wachstums (ohne lange Nachzudenken). Mit $\log n$ bezeichnen wir den Logarithmus zur Basis 2.

- $\log(n)$
- $\log(n^5)/\log\log(n)$
- $n^{\log\log n}$
- $\log(n)^{\log n}$
- n^2
- $n \log n$
- 2^n

Aufgabe T2

Für welche Funktionen $f\colon \mathbf{N} \to \mathbf{N}$ gilt $f(O(1)) = O(1)$? Beweisen Sie Ihre Behauptung mithilfe der Definition der O-Notation.

Aufgabe T3

Entwerfen Sie einen einfachen Algorithmus, der zwei doppelte verkettete Listen aneinanderhängt und auf diese Weise eine neue doppelt verkettete Liste erzeugt. Können Sie dabei auf bedingte Verzweigungen verzichten?

Schreiben Sie den Algorithmus in Pseudocode oder in einer Programmiersprache nieder.

Aufgabe H1

Gegeben sei ein Array $a[0], \ldots, a[n-1]$, welches Zahlen in aufsteigend sortierter Reihenfolge enthält. Entwerfen Sie einen effizienten Algorithmus, der die Anzahl der Zahlen in diesem Array bestimmt, welche sich in einem Interval $[l, r]$ befinden, wobei l und r ebenfalls zwei Zahlen sind.

Die Laufzeit ihres Algorithmus sollte $O(\log n)$ betragen.

Erläutern Sie die Arbeitsweise des Algorithmus und begründen Sie, warum er schnell ist.

Beachten Sie insbesondere den Fall, wenn das Array auch die Zahlen l und r mehrfach enthält. Auch dann muß das Ergebnis korrekt sein.

Implementieren Sie Ihren Algorithmus in einer geeigneten Sprache und lassen Sie ihn auf einigen aussagekräftigen Beispielen laufen. Verwenden Sie dabei auch Fälle mit $n = 10000000$. Überlegen Sie sich ein Verfahren, wie sie die Laufzeit ihrer Suche in diesem Falle möglichst gut messen können und geben Sie die dabei gefundene Zeit an.

Aufgabe H2

Effizienz spielt in dieser Vorlesung eine große Rolle. Will man ein spezielles Problem aber nur ein einziges Mal lösen, dann achtet man nicht so gerne auf Effizienz, sondern versucht es mit so wenig Aufwand wie möglich zu lösen. Insbesondere versucht man vorhandene Programme wiederzuverwenden, um möglichst wenig selbst tun zu müssen.

In dieser Aufgabe versuchen wir also ein solches Problem mit möglichst wenig Aufwand zu lösen. Sie dürfen alle Hilfsmittel, die Ihnen einfallen, verwenden, mit Ausnahme von Hilfe durch andere Personen.

In der Datei `words` auf unserer Webseite finden Sie 462983 Zeilen mit je einem Wort. Einige der Worte in dieser Datei finden sich dort auch rückwärts geschrieben, wobei wir Groß- und Kleinschreibung ignorieren wollen. Ein Beispiel eines solchen Wortes ist `redrawer` und ein weiteres ist `Abba`.

Wie können Sie diese Aufgabe *mit so wenig Aufwand* wie möglich lösen? Was ist die genaue Anzahl dieser besonderen Wörter in der Datei `words`?