Eidgenössische
Technische Hochschule
Zürich

Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Federal Institute of Technology at Zurich

Departement Informatik
Markus Püschel
Peter Widmayer
Thomas Tschager
Tobias Pröger

3. November 2016

Datenstrukturen & Algorithmen — Blatt 7 — HS 16

Abgabe: Am Donnerstag, den 10.11.2016, vor Beginn der Vorlesung um 10 Uhr im Eingangsbereich vor ML D28. Bitte heften Sie Ihre Blätter zusammen und benutzen Sie dieses Blatt als Deckblatt. Füllen Sie auch die ersten zwei der untenstehenden Felder aus.

Übungsstunde (Raum & Zeit): _____

Abgegeben von: _____

Korrigiert von: _____

erreichte Punkte: ____

Hinweis: Dieses Blatt behandelt dynamische Programmierung. Eine vollständige Beschreibung eines dynamischen Programms behandelt **immer** die folgenden Aspekte (interessant auch für die Klausur!):

1) *Definition der DP-Tabelle:* Welche Dimensionen hat die Tabelle? Was ist die Bedeutung jedes Eintrags?
2) *Berechnung eines Eintrags:* Wie berechnet sich ein Eintrag aus den Werten von anderen Einträgen? Welche Einträge hängen nicht von anderen Einträgen ab?
3) *Berechnungsreihenfolge*: In welcher Reihenfolge kann man die Einträge berechnen, so dass die jeweils benötigten anderen Einträge bereits vorher berechnet wurden?
4) *Auslesen der Lösung:* Wie lässt sich die Lösung am Ende aus der Tabelle auslesen?

Die Laufzeit eines dynamischen Programms berechnet sich üblicherweise einfach aus der Grösse der Tabelle multipliziert mit dem Aufwand, jeden Eintrag zu berechnen. Manchmal überwiegt jedoch auch der Aufwand um die Lösung auszulesen.

Aufgabe 7.1 *Palindrome aufzählen.*

Ein *Palindrom* ist eine Zeichenkette, die sich von vorne wie von hinten gleich liest, also z.B. das Wort `RENTNER`. Formal ist ein Palindrom eine Zeichenkette $\langle a_1, \ldots, a_n \rangle$, wobei entweder $n = 1$ gilt, oder aber es sind $a_1 = a_n$ und $\langle a_2, \ldots, a_{n-1} \rangle$ ein Palindrom (für $n = 2$ fordern wir lediglich $a_1 = a_2$). Ein Array $A[1..n]$ speichere eine Zeichenkette der Länge n. Ein Teilarray $A[i..j]$, $1 \leq i \leq j \leq n$, heisst *Palindrom in* A, falls $\langle A[i], \ldots, A[j] \rangle$ ein Palindrom ist.

Beispiel: Das Array [`L`, `A`, `R`, `A`] enthält die Palindrome `A`, `L`, `R` sowie `ARA` (das Palindrom `A` kommt doppelt vor). Das Array [`A`, `N`, `N`, `A`] enthält die Palindrome `A`, `N`, `NN` sowie `ANNA` (die Palindrome `A` und `N` kommen doppelt vor).

Bitte wenden.

a) Sei A ein Array, das eine Zeichenkette der Länge n speichert. Entwerfen Sie einen Algorithmus nach dem Prinzip der dynamischen Programmierung, der alle Paare (i, j) ausgibt, für die $\langle A[i], \ldots, A[j] \rangle$ ein Palindrom ist. Geben Sie auch die Laufzeit Ihrer Lösung an.

Beispiel: Für die Eingabe [`L`, `A`, `R`, `A`] werden die Paare $(1, 1)$, $(2, 2)$, $(3, 3)$, $(4, 4)$, $(2, 4)$ ausgegeben. Für die Eingabe [`A`, `N`, `N`, `A`] werden die Paare $(1, 1)$, $(2, 2)$, $(3, 3)$, $(4, 4)$, $(2, 3)$, $(1, 4)$ ausgegeben. Es wird keine spezielle Reihenfolge gefordert, in der die Paare ausgegeben werden müssen.

Hinweis: Beachten Sie, dass die Aufgabe ohne dynamische Programmierung durch triviale Aufzählung aller Palindrome in Zeit $\mathcal{O}(n^3)$ lösbar ist. Gesucht wird also ein effizienterer Algorithmus.

b) Angenommen, der Algorithmus aus a) hat die DP-Tabelle bereits berechnet. Beschreiben Sie detailliert, wie Sie aus der DP-Tabelle ein längstes Palindrom in A ablesen können. Geben Sie auch die maximal benötigte Laufzeit an.

Aufgabe 7.2 *Aufsteigende Sequenzen.*

In dieser Aufgabe betrachten wir eine Matrix A, bestehend aus n Zeilen und m Spalten. Zwei Elemente in der Matrix sind *benachbart*, wenn sie sich an einer Seite berühren, d.h. das Element $A[i][j]$ ist mit den vier Elementen $A[i-1][j]$, $A[i][j-1]$, $A[i+1][j]$ und $A[i][j+1]$ benachbart, falls diese Felder existieren (Elemente am Rand der Matrix sind mit entsprechend weniger Elementen benachbart).

Eine Folge $x_1, x_2, \ldots, x_k$ von Elementen in der Matrix heisst *aufsteigende Sequenz*, wenn folgende Bedingungen erfüllt sind:

- die Folge von Elementen ist aufsteigend sortiert, und
- für jedes $i \in \{1, \ldots, k-1\}$ sind die Elemente x_i und x_{i+1} in der Matrix benachbart.

Gesucht wird eine längstmögliche aufsteigende Sequenz in der gegebenen Matrix. In der untenstehenden Beispiel-Matrix wäre $4, 6, 28, 29, 47, 49$ eine mögliche Sequenz. Entwerfen Sie einen möglichst effizienten Algorithmus der dynamischen Programmierung, der eine solche längste Sequenz findet. Beschreiben Sie Ihren Algorithmus und geben Sie dessen Laufzeit an.

Beispiel-Matrix:

9	27	42	41	48
35	39	8	3	5
12	49	2	38	4
15	47	29	28	6
19	1	25	33	10