Eidgenössische
Technische Hochschule
Zürich

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

Markus Püschel
Peter Widmayer
Thomas Tschager
Tobias Pröger
Tomáš Gavenčiak

Department Informatik
11. Januar 2017

Algorithmen & Datenstrukturen — Probeprüfung — HS 16

Das ist die zweite Aufgabe der Probeprüfung. Die Probeprüfung hat keinen Einfluss auf die Endnote. Wir empfehlen, diese Aufgabe zu Hause oder sogar während der Probeprüfung zu lösen, falls die Zeit reicht.

Maximale Teilmatrix

Für eine gegebene $n \times n$ Matrix $A = (a_{ij})_{0 \le i,j \le n-1}$ soll eine Teilmatrix mit grösstmöglicher Summe der Einträge gefunden werden. Eine $a \times b$ Teilmatrix einer $n \times n$ Matrix ist ein zusammenhängendes $a \times b$ Rechteck von Einträgen in A.

Beachten Sie, dass eine leere 0×0 Teilmatrix auch erlaubt ist und dass die Summe ihrer Einträge 0 ist. Im Beispiel unten ist eine Matrix mit Zeilen- und Spaltenindizes dargestellt, sowie eine 1×2 Teilmatrix mit grösstmöglicher Summe der Einträge, nämlich 10.

	0	1	2
0	-3	3	7
1	5	-4	1
2	0	-2	1

Es gibt mehrere Lösungen mit unterschiedlichen Laufzeiten. Weiter unten finden Sie einige Hinweise zu möglichen Lösungsstrategien. Denken Sie zuerst selbst über mögliche Ansätze nach, bevor Sie die Hinweise lesen. *Beachten Sie, dass wir in der Schlussprüfung nicht notwendigerweise Hinweise geben werden.*

Eingabe Die erste Zeile der Eingabe enthält die Ganzzahl $t \ge 1$, die Anzahl der Tests.

Die erste Zeile jedes Tests enthält $n \le 100$, Anzahl der Zeilen und Spalten der gegebenen $n \times n$ Matrix. Es folgen n Zeilen, wobei jede Zeile n Ganzzahlen enthält (durch Leerzeichen getrennt). Diese Zeilen repräsentieren A. Die i-te Zeile entspricht der i-ten Zeile von A, d.h. sie enthält $a_{i,j}$ für $0 \le j \le n-1$. Für alle Einträge der Matrix gilt $-100 \le a_{i,j} \le 100$.

Ausgabe Geben Sie die grösstmögliche Summe einer Teilmatrix als Ganzzahl aus; für jeden Test in einer separaten Zeile.

Beispiel

Eingabe (erster Test entspricht dem Beispiel oben):

```
3
3
-3 3 7
5 -4 1
0 -2 1
2
-2 -3
-1 -4
2
2 -1
-2 1
```

Ausgabe (eine 1×2 Teilmatrix; eine leere Teilmatrix; eine 1×1 Teilmatrix):

```
10
0
2
```

Punkte Sie können bis zu 100 Punkte am Judge bekommen. Um alle Punkte zu bekommen, muss Ihr Programm eine Laufzeit in $\mathcal{O}(n^3)$ haben (beachten Sie, dass die Eingabe n^2 Einträge umfasst). Langsamere Programme mit Laufzeit in $\mathcal{O}(n^4)$ oder $\mathcal{O}(n^5)$ können einen Teil der Punkte bekommen.

Senden Sie Ihr `Main.java` unter folgendem Link ein: `https://judge.inf.ethz.ch/team/websubmit.php?cid=18986&problem=DNA14_3`. Das Passwort für die Einschreibung ist "`testitwell`". Der Judge wird mindestens bis zur Schlussprüfung Lösungen für diese Aufgabe annehmen. Sie können auch nach der Probeprüfung noch Lösungen einreichen.

Hinweise Wir stellen für diese Aufgabe eine Programmvorlage als Eclipse-Projektarchiv zur Verfügung. Die Vorlage hilft Ihnen beim Einlesen der Eingabe.

Wie üblich enthält das Archiv auch Testdaten, damit Sie lokal testen können. Ausserdem stellen wir ein `Judge.java` Programm zur Verfügung, das Ihr Programm `Main.java` mit allen verfügbaren Testdaten testet – öffnen und starten sie dazu einfach `Judge.java`. Die zur Verfügung gestellten Testdaten sind nicht dieselben Testdaten, welche der Online-Judge verwendet, und auch nicht so umfangreich.

In dieser Prüfung dürfen Sie *keine* Java Libraries ausser `java.util.Scanner` einbinden. Dies wird erst nach der Prüfung getestet und bestraft, falls Ihr Code andere Bibliotheken `import`iert.

Hinweise zu möglichen Lösung

Beachten Sie, dass wir bei der Schlussprüfung nicht notwendigerweise Hinweise geben werden.

Ein naiver Algorithmus überprüft jedes der $\mathcal{O}(n^4)$ möglichen Teilmatrizen und berechnet jeweils die Summe der Einträge mit Laufzeit in $\mathcal{O}(n^2)$. Das ergibt einen Algorithmus mit Laufzeit in $\mathcal{O}(n^6)$.

Für eine Lösung mit Laufzeit in $\mathcal{O}(n^4)$ können Sie die Summe aller Teilmatrizen mit linker, oberer Ecke (0,0) vorberechnen. Dieser Ansatz wird in der Abbildung rechts illustriert. Dann können Sie die Summe des grauen Rechtecks schnell mit $A - B - C + D$ berechnen. Berechnen Sie schliesslich die Summe aller möglichen Teilmatrizen.

Um das Problem noch schneller zu lösen, können Sie das Problem auf das *Maximum Subarray Sum* Problem reduzieren. Dann kann man einen Algorithmus mit Laufzeit in $\mathcal{O}(n^3)$ erreichen. Betrachten Sie jede mögliche Wahl der ersten und der letzten Zeile der Teilmatrix. Berechnen Sie für jede Wahl der Zeilen `top` und `bottom` auf effiziente Weise die Spaltensummen zwischen diesen Zeilen und finden Sie dann das Teilarray mit grösstmöglicher Summe für diese Spaltensummen in Laufzeit $\mathcal{O}(n)$. In der Abbildung links wird dieser Ansatz illustriert.