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
11th January 2017

Algorithmen & Datenstrukturen Trial Examination AS 16

This is the second trial exam task. The results of the trial exam have no weight on the final grade. We recommend to try this at home, or even at the trial exam if you have enough time.

Maximal submatrix

For a given $n \times n$ matrix $A = (a_{ij})_{0 \le i,j \le n-1}$, find a submatrix with the largest sum of entries. An $a \times b$ submatrix of a $n \times n$ matrix is a continuous $a \times b$ rectangle of the entries of A. Note that an empty 0×0 submatrix is also allowed and has sum 0. See below for an example matrix with row and column indexes and a 1×2 submatrix with the maximum sum 10.

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

There are several solutions with various running times. You can find some hints for solving this exercise below. Think about how you can solve this exercise before you read them. *(Note the final exam task may not provide a hint.)*

Input The first line of the input contains the integer $t \ge 1$, the number of test cases.

The first line of each of the test cases contains $n \le 100$, the size of the given square matrix. After that, there are n lines, each with n integer numbers representing A, separated by spaces. The i-th line corresponds to the i-th row of A, i.e., it contains $a_{i,j}$ for $0 \le j \le n-1$. All the matrix elements satisfy $-100 \le a_{i,j} \le 100$.

Output Output the largest sum of a submatrix as an integer for each test case on separate lines.

Example

Input (first case same as above):

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

Output (using 1×2 submatrix; empty submatrix; 1×1 submatrix):

```
10
0
2
```

Grading You may get up to 100 judge points. To get full points, your program should run in time $\mathcal{O}(n^3)$ (note that the input is of size n^2), but slower programs of complexity $\mathcal{O}(n^4)$ or $\mathcal{O}(n^5)$ can get partial points.

Submit your `Main.java` at `https://judge.inf.ethz.ch/team/websubmit.php?cid=18986&problem=DNA14_3`, enroll password is "`testitwell`". The judge will be open for submissions of this task at least until the final exam.

Notes For this exercise, we provide a program template as an Eclipse project in your workspace that helps you read the inputs.

As usual, the project also contains data for your local testing and `Judge.java` program that runs your `Main.java` on all the local tests – just open and run `Judge.java` in the project. Also, the local test data are of course different and generally smaller than the data that are used in the online judge.

In this exam, you *must not* `import` any Java libraries except for `java.util.Scanner`. This is not checked on submission but will be checked afterwards and penalized if your code `imports` other libraries.

Hints for solving the exercise

Note the actual exam task may not provide a hint.

A naive algorithm checks each of the $\mathcal{O}(n^4)$ possible submatrices and computes the sum of its entries in time $\mathcal{O}(n^2)$. This leads to an algorithm with running time in $\mathcal{O}(n^6)$.

To get a solution in time $\mathcal{O}(n^4)$, precompute the sums of all submatrices starting at (0,0) (top left corner) as indicated on the right figure below. Then you can quickly compute the sum of the gray rectangle as $A - B - C + D$. Then compute the sum of all submatrices.

To get the fastest solution in time $\mathcal{O}(n^3)$, reduce the problem to the *maximum subarray sum* problem: Try all the choices for the first and the last line of the submatrix. For each of those choices of `top` and `bottom` row, efficiently compute the column sums between these rows, then find a maximum subarray sum value for the column sums in time $\mathcal{O}(n)$. The left figure below illustrates this approach.