Algorithms and Data Structures — Università della Svizzera italiana
A. Carzaniga, A. Fattaholmanan, I. Mantas, M. Bezdrighin

Graded Assignment n. 5
May 23, 2018

Graded Assignment 5

Due date: May 31, 2018 at 20:00

This is an individual assignment. You may discuss it with others, but your formulations, your code, and all the required material must be written on your own. In any case, you must acknowledge the sources used and clearly mention any help received from colleagues.

Exercise 1. *(40% grade)*

Instead of longest increasing subsequences, Antonio has been studying longest *irregular* subsequences in his free time. Given a sequence of integers $a_1, a_2, \ldots, a_n$ we define an *irregular* subsequence as any subset $a_{i_1}, a_{i_2}, \ldots, a_{i_k}$ such that $1 \leq i_1 < i_2 < \ldots < i_k \leq n$ and:

$$a_{i_1} > a_{i_2} < a_{i_3} > a_{i_4} < a_{i_5} \ldots$$

Note that a sequence of length one is trivially an *irregular* subsequence. Given a sequence of length n, find the maximal length of any irregular subsequence. The complexity of your algorithm must be $O(n^2)$. The input to your program will consist of a single line containing the numbers $a_1, a_2, \ldots, a_n$ separated by spaces. Your program should output the maximal length of an irregular subsequence to the standard output.

Examples:	Input	Output	Input	Output	Input	Output
	`0 4 2 3 3`	`3`	`1 2 3 4`	`1`	`2 1 2 1 2`	`5`

Exercise 2. *(40% grade)*

Antonio has an $n \times m$ grid containing non-negative integers. He calls a path in this grid *interesting* if the path starts in some cell, goes through a sequence of adjacent cells, and ends in some other cell of the grid without visiting a cell twice. (Two cells are called adjacent if they share a side.) In addition, an *interesting* path is such that the value in every cell is larger than the value of the previous cell of the path (except for the starting cell in the path).

Given a grid of size $n \times m$, find out the maximal length of any *interesting* path. The complexity of your algorithm should be $O(nm \log(nm))$.

The input to your program will consist of a single line containing n and m, followed by n lines that contain m numbers each. Your program should output the length of the longest interesting path to the standard output.

Examples:	Input	Output	Input	Output

	`4 4`
	`5 6 7 9`
	`9 2 4 8`
	`3 1 2 0`
	`4 5 0 3`
		`5`	`3 4`
			`1 4 4 4`
			`3 3 3 3`
			`2 2 2 2`
				`3`

A maximal interesting path in the first example is 1 2 4 8 9; one in the second example is 2 3 4.

Exercise 3. *(20% grade)*

One day Antonio found in the attic a string s of length n composed of letters of the Latin alphabet. He's a big fan of *palindromic* strings so he decided to find the longest *palindromic* subsequence of this string. We define a *palindromic* subsequence as any subset $s_{i_1}, s_{i_2}, \ldots, s_{i_k}$ such that $1 \leq i_1 < i_2 < \ldots < i_k \leq n$ and:

$$s_{i_1} s_{i_2} \ldots s_{i_k} = s_{i_k} s_{i_{k-1}} \ldots s_{i_1}$$

Given a string of length n, find out the length of the longest palindromic subsequence. The complexity of your algorithm should be $O(n^2)$.

The input to your program will consist of a single line containing the given string s. Your program should output the length of the longest palindromic subsequence to the standard output.

Examples:	Input	Output	Input	Output	Input	Output	Input	Output
	`antonio`	`3`	`mammamia`	`6`	`racecar`	`7`	`abcdef`	`1`