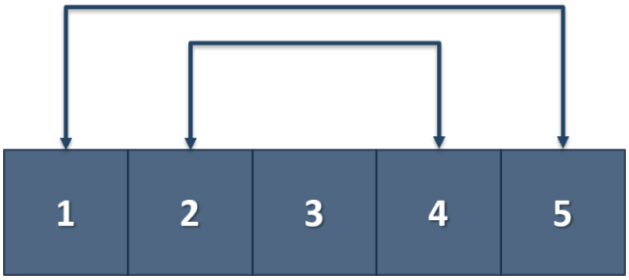
Memory Management, Pointers and References – Exercise

Submit your solutions here: https://judge.softuni.org/Contests/3014/Memory-Management-References-and-Pointers-Exercise

1. Gauss' Trick

Write a program that **sums** all of the **numbers in a list** in the following order:
first **+** *last*, *first* + *1* **+** *last* **-** *1*, *first* + *2* **+** *last* **-** *2*, … *first* + *n*, *last* - *n*.



Example

Input	Output
1 2 3 4 5	6 6 3
1 2 3 4	5 5

2. Remove Negative and Reverse

Read a **list of integers**, **remove all negative numbers** from it and print the remaining elements in **reversed order**. In case there are no elements left in the list, print "**`empty`**".

Examples

Input	Output
10 -5 7 9 -33 50	50 9 7 10
7 -2 -10 1	1 7
-1 -2 -3	empty

3. Print in Parts

Write a program that receives a 2-dimensional dynamic array with **N** rows and **M** columns and returns the first **R** rows and **C** columns.

You are not allowed to use STL containers and memory management.

Example

Input	Output
3 4 1 2 3 4 11 22 33 44	1 2 3 11 22 33 111 222 333

111 222 333 444 3 3	
4 4 2 2 2 2 5 6 7 8 0 1 0 2 9 7 5 3 2 4	2 2 2 2 5 6 7 8
4 4 2 2 2 2 5 6 7 8 0 1 0 2 9 7 5 3 4 2	2 2 5 6 0 1 9 7

4. Some Ordering

Write a program that receives a string with N elements and returns two other strings – the first one is the same string with lower-case letters only, and the second one is the same string with upper-case letters only. Write the program with the help of pointers!

You are not allowed to use STL containers and memory management.

Example

Input	Output
I love Programming.	i love programming. I LOVE PROGRAMMING.
Let’s go on a Vacation!	let's go on a vacation! LET'S GO ON A VACATION!
Just Use POINTERS	just use pointers JUST USE POINTERS

5. Compare Matrices

Write a program that reads two integer matrices (2D arrays) from the console and compares them element by element.
You are not allowed to use STL containers and memory management.

For better code reusability, you could do the comparison in a function, which returns **`true`** if they are equal and **`false`** if not.

Each matrix definition on the console will contain a line with a positive integer number **R** – the number of rows in the matrix – followed by R lines containing the numbers in the matrix, separated by spaces (each line will have an equal amount of numbers.

The matrices will have at most **`10`** rows and most **`10`** columns.

Print **`equal`** if the matrices match, and **`not equal`** if they do not match.

Examples

Input	Output
1 1 2 3 1 1 2 3	equal
2 1 2 3 2 1 3 2 1 2 3 2 1 3	equal
4 1 11 21 31 4 1 11 21 31	equal
2 1 2 3 4 5 6 2 1 3 2 4 5 6	not equal
2 1 2 3 4 5 6 2 1 2 3 4	not equal

6. Minesweeper

You are given an **N** by **M** matrix (**N** and **M** are two integers entered on the console), in which the cells contain single characters – either a **.** (dot) or a **!** (exclamation mark) – representing "empty" or "mined" positions.

Write a program that prints an **N** by **M** matrix, where each cell contains a number, representing how many adjacent cells, **including itself**, are "mined".

Each cell in a matrix has at most 8 adjacent cells – the cells directly above, below, to the left, to the right, as those diagonally – to the left and above, to the right and above, to the right and below and to the left and below.

You are not allowed to use STL containers and memory management!

Examples

Input	Output	Input	Output	Input	Output
5 5 ....!.	00111 00111 00111	5 8 ....!....	00111000 00122100 00122100	3 3 !!! !.!	353 585 353

....	00000 00000		....!... ....!....	00122100 00111000		!!!	