Exercise: Recursion and Combinatorial Problems

Please submit your solutions (source code) of all below-described problems in Judge.

1. Reverse Array

Write a program that reverses and prints an array. Use **recursion**.

Examples

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

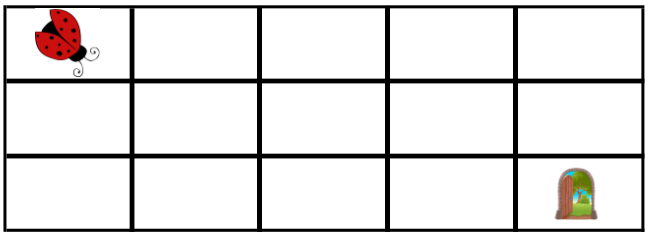
2. Nested Loops To Recursion

Write a program that simulates the execution of n nested loops **from 1 to n** which prints the values of all its iteration variables at any given time on a single line. **Use recursion.**

Examples

Input	Output
2	1 1 1 2 2 1 2 2
3	1 1 1 1 1 2 1 1 3 1 2 1 1 2 2 ... 3 2 3 3 3 1 3 3 2 3 3 3

3. Move Down / Right



Marinette has the ability to transform into Ladybug. She is stuck on a grid. Her initial location is at the **top-left corner** and tries to move to the **bottom-right corner**. However, she can only **move either down or right** at any point in time.

Write a program that prints the number of **all possible unique paths** that Marinette can take to reach the **bottom-right corner**.

Input

- On the first line you will receive an integer – **m** – number of rows.
- On the second line you will receive an integer – **n** – number of cols.

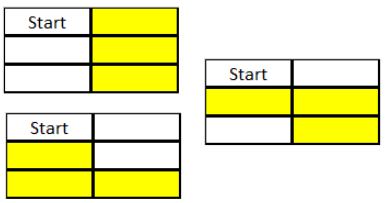
Output

- Print the number of all possible unique paths to the bottom-right corner.

Constraints

- **m** and **n** will always be in the range [0… 1000].

Examples

Input	Output	Comments
3 2	3	All possible unique paths: 
3 5	15	

4. Connected Areas in a Matrix

Let's define a **connected area** in a matrix as an area of cells in which there is a **path between every two cells**.

Write a program to find **all** connected areas in a matrix.

Input

- On the first line, you will get the **number of rows**.
- On the second line, you will get the **number of columns**.
- The rest of the input will be the **actual matrix**.

Output

- Print on the console the **total number of areas found**.
- On a separate line for each area print its **starting coordinates** and **size**.
- **Order** the areas by size (in descending order) so that the **largest area is printed first**.
    - If several areas have the same size, order them **by their position**, first by the row, then by the column of the top-left corner.
    - If there are two connected areas of the same size, the one which is above and/or to the left of the other will be printed first.

Examples

Example Layout	Output
4 9 ---*---*- ---*---*- ---*---*- ----*-*--	Total areas found: 3 Area #1 at (0, 0), size: 13 Area #2 at (0, 4), size: 10 Area #3 at (0, 8), size: 5
5 10 *--*---*-- *--*---*-- *--*****-- *--*---*-- *--*---*--	Total areas found: 4 Area #1 at (0, 1), size: 10 Area #2 at (0, 8), size: 10 Area #3 at (0, 4), size: 6 Area #4 at (3, 4), size: 6

Hints

- Create a method to find the first traversable cell which hasn't been visited. This would be the top-left corner of a connected area. If there is no such cell, this means all areas have been found.
- You can create a class to hold info about a connected area (its position and size). Additionally, you can implement Comparable and store all areas found in a TreeSet.

5. Word Cruncher

Write a program that receives some **strings** and **forms another** string that is required. On the **first line,** you will be given **all of the strings** separated by **comma and space**. On the next line, you will be given the **string** that needs to be **formed from the given strings**. For more clarification see the examples below.

Input

- On the first line, you will receive the **strings** (separated by comma and space **", "**)
- On the next line, you will receive the **target string**

Output

- Print each of them found ways to form the required string as shown in the examples

Constrains

- There might be **repeating elements** in the input

Examples

Input	Output
text, me, so, do, m, ran somerandomtext	so me ran do m text
Word, cruncher, cr, h, unch, c, r, un, ch, er Wordcruncher	Word c r un ch er Word c r unch er Word cr un c h er Word cr un ch er Word cr unch er Word cruncher
tu, stu, p, i, d, pi, pid, s, pi stupid	s tu p i d s tu pi d s tu pid stu p i d stu pi d stu pid