Problem 4-E4 – Image encoding

Digital photography is one of the most popular forms of visual information representation. An image is represented as a matrix of positive integers, with values in the [0; 255] range, distributed across lines and columns. Due to the significant memory need, the images are often encoded in order to reduce the transmitted information.

Requirement

Given an image, **A**[][], of size *m* (number of lines) x *n* (number of columns), encode the image as following. The image is converted into a vector, **v1**[] (the image is scanned from left to right and from top to bottom). Each **different** value from the vector, **v1**[i], is converted into two values: [*a b*], where *a* represents the actual value, and *b* the number of occurrences of this value in the vector. These new values, concatenated, will represent the encoded signal of the image **A**[][] and will be stored in **v2**[]. Display the reduced amount of data, computed as the difference between the sizes of **v1**[] and **v2**[]. This can be either a positive value or a negative one.

Entry data

The program will receive, from the keyboard (*stdin* stream), the following data:

- an integer value as the number of lines *m*, followed by *newline* (*Enter* key);
- an integer value as the number of columns *n*, followed by *newline* (*Enter* key);
- the **A** matrix values, as one value on a single line followed by *newline* (*Enter* key), scanned from left to right and from top to bottom (scroll on lines and columns).

Output data

The program will display a single integer value, representing the difference between the size of the two vectors, i.e. the initial vector and the final, encoded one, followed by *newline* character (*Enter* key).

WARNING to the compliance of the problem requirement: displaying the results should be EXACTLY on the indicated way! In other words, the standard output stream will not show anything in addition to the requirement of the problem; as a result of automatic evaluation, any additional character displayed, or viewed other than that indicated, will lead to a false result and therefore obtain a Rejection of the program.

Restrictions and specifications

1. The size of the matrix is represented by positive integers, greater than 2 and smaller than 20. The matrix values are positive integers, in the range of [0; 255].
2. **Warning**: Depending on the chosen programming language, the file containing the code must have one of the following extensions .c, .cpp, .java, or .m. The web editor **will not automatically add an extension** and its absence leads to the impossibility of compiling the program!
3. **Warning**: The source file must be named, by the candidate, in the following format: <name>.<ext>, where name is the surname of the candidate and the extension is chosen according to the previous point. Pay attention to the limitations of the Java language, related to the class name and file name!

Example

Input	Output
3 4 2 2 1 3 4 3 2 4 2 4 4 4	4

Explanation: **A**=$\begin{bmatrix} 2 & 2 & 1 & 3 \\ 4 & 3 & 2 & 4 \\ 2 & 4 & 4 & 4 \end{bmatrix}$;

v1=[2 2 1 3 4 3 2 4 2 4 4 4] (through crossing **A**);
v1=[1 2 2 2 2 3 3 4 4 4 4 4] (sorted in ascending order);
Observation: the sorting of the values is not mandatory for solving the problem.

Different values:
v1[0]=1 -> a=1, b=1 (a single occurrence), [1 1];
v1[1]=2 -> a=2, b=4 (4 occurrences), [2 4];
v1[5]=3 -> a=3, b=2 (2 occurrences), [3 2];
v1[7]=4 -> a=4, b=5 (5 occurrences), [4 5];

v2=[[1 1] [2 4] [3 2] [4 5]]=[1 1 2 4 3 2 4 5];
size(**v1**)=12;
size(**v2**)=8;
Difference=12–8=4.

Working time: 120 minutes