# Activity 1. Pattern matching with texts

For this activity, we need to implement a pattern matching algorithm. This pattern matching algorithm should work to compare strings containing also * and ?. * stands for any combination of characters and ? stands for any character.

To implement this we use a matrix with as many rows as the length of the target string + 1 and as many columns as the length of the string we want to compare + 1 (we have an extra row and column in case there're any empty string).

Therefore, the program uses two nested loops to iterate over the matrix and compute the values we need to compare. To do so, the first loop will have a complexity of O(n), been n the length of the string that we need to check, and the second one will have a complexity of O(m), being m the length of the target string. Therefore, we have a total complexity of O(n*m), equivalent to $O(n^2)$.

The advantage of this type of algorithm is that we can see the partial solutions for our problem, so we can know until what point of the execution we were on a good path and where it breaks down.

I think that reducing this time complexity is very complex. Although, better performance can be obtained for the best cases of this algorithm.