Problem Statement :-

It's winter time! Time for snowmen to play some games. Two snowmen are playing a game. In this game, the first snowman must choose a subset of the set {1, 2, ..., N}, and the second one must choose a subset of the set {1, 2, ..., M}. The following two conditions must be fulfilled:

- The two sets have an empty intersection.
- The XOR of all elements in the first set is less than the XOR of all elements in the second set.

Input Specification:-

Two ints N and M will be given and $1 <= N, M <= 2000$

Output Specification:-

Let X be the total number of different ways to choose the pair of sets.Return the value X modulo 1,000,000,007

Design:-

Few xor optimisations:

Let x be the xor of elements in a subset A (derived from {1,2,..N}) and y be the xor of elements of the other subset(call it B) we need x<y

If x<y the binary representations of them would be this way:

x = pqrst0?????
y = pqrst1?????

There should be a bit position i such that the two numbers have equal bits in the higher bit positions, y has 1 in the i - th position and x has 0 in that position. The remaining bit positions can have any value.

Instead of remembering two values to verify the condition we can remember Z = x^y because it follows the pattern that if x<y
Z = 000001?????

Z must have all zero's until the i-th bit which should have 1 and the i-th bit of x should be 0.

and we will have one more optimisation i.e.. since we are concerned with only the i-th bit of Z we do Z>>i for our convenience in the implementation.This makes our required bit the lowest order bit of Z.

We can describe the final state by two numbers x,y the xor of all elements of the respective sets.Initially let x=y=0 and we have to assign all t <=T=max(N,M) to one or none of the sets.

If we add t to first set then x = x^t and this implies z = z^t and let the i-th bit of the initial x be b and the i-th bit of t be k then b = b^k.

If we add t to second set the y=y^t this implies z=z^t and b remains the same.

<u>Optimal Substructure:</u>

Any number can be in either of the subsets or in none of them.

<u>Recursive Solution:-</u>

Base Case: If t = 0,result is 1 if Z contains only zeros for bit positions higher than i and 1 for bit i and b=0

Otherwise:

If t<=N we can add t to first set. To count the number of ways to assign the remaining t-1 integers we can call dp(t-1,Z^t,b^k).

If t<=M we can add t to the other set : dp(t-1,Z^t,b).

We can choose the element to be in none of the set: dp(t-1,z,b).

We must do the DP for each bit position i and the maximum number of bit can be calculated as 2000 is the max value of M or N so maximum number of bits is 11.For a given i, the values of the bit positions smaller than i do not matter. We can actually ignore them altogether and remove the first i bits from all numbers in the dp.

Proof of Correctness:-

To prove the correctness of the recursive solution we need to prove that it:

1. Terminates
2. It satisfies the given two conditions:
    - The two sets have an empty intersection.
    - The XOR of all elements in the first set is less than the XOR of all elements in the second set.

Termination:-

The recursion terminates because t will definitely be 0 as in every recursive call we send t-1.

Empty Intersection:-

We put t in either first set or second set or none of them.This ensures that we don’t have common elements in both the sets.

XOR CONDITION:-

In the termination condition the check of i-th bit of Z having 1 and b=0 ensures that this condition is satisfied.

Sample Inputs:-

1)
2
2

2)
1
1

3)
7
4

4)
47
74

5)
1987
1789

6)
1899
1999

Outputs Of Sample Inputs:-

1) Returns: 4
2) Returns: 1
3) Returns: 216
4) Returns: 962557390
5) Returns: 553925400
6) Returns: 564171057