Here the a summary of a topcoder problem:

- You are given a vector of strings called **t**.
- Each string in **t** is the same size, and is composed of the characters `'0'` and `'1'`.
- **t** has at most 10 elements, and each string has at most 10 characters.
- **t** is considered "Nice" if there exist strings **x** and **y** such that:
  - **x.size()** equals **t.size()**.
  - **y.size()** equals **t[0].size()**.
  - Both **x** and **y** are composed of `'1'` and `'0'`.
  - If we consider all of the characters in **t**, **x** and **y** to be numbers, then for all *i* and *j*, **t[i][j]** is equal to **x[i]** XOR **y[j]**.
- Return "Nice" if **t** is Nice and "Not nice" otherwise.

```
Examples:

#     t         Answer
-   --------    ---------------------------------
0   { "01",     "Nice" -- x = y = "10" works.
      "10" }     So does  x = y = "01"

1   { "01",     "Not Nice"
      "11" }    Trust me, you can't do it.

2   {"0100",     "Nice".
     "1011",     x = "101" and y = "1011" works.
     "0100"}
```

Question 1

Which type of enumeration can solve this problem? Just put the letter of the answer into the TurningPoint text box:

- **A**: Div-Mod Enumeration.
- **B**: Power Set Enumeration.
- **C**: *n*-choose-k.
- **D**: Permutations.

Question 2:

If **t.size()** is *n*, and **t[0].size()** is *m*, then what is the running time of the enumeration?

Answers to the Clicker Questions

Question 1: You will do two enumerations of strings composed of '0' and '1' -- one for **x** and one for **y**. These are power set enumerations: **B**.

Question 2: Enumerating **x** is $O(2^n)$, and enumerating **y** is $O(2^m)$, so the answer is $O(2^{n+m})$, which can also be written $O(2^n * 2^m)$.