Problem

During some extraterrestrial exploration, you found evidence of alien poetry! Your team of linguists has determined hat each word in the alien language has an accent on exactly one position (letter) in the word; the part of the word st rting from the accented letter is called the accent-suffix. Two words are said to rhyme if both of their accent-suffixes are equal. For example, the words PROL and TARPOL rhyme if the accented letter in both is the O or the L, but the do not rhyme if the accented letters are the Rs, or the R in PROL and the P in TARPOL, or the O in PROL and the in TARPOL.

You have recovered a list of N words that may be part of an alien poem. Unfortunately, you do not know which is th accented letter for each word. You believe that you can discard zero or more of these words, assign accented letters o the remaining words, and then arrange those words into pairs such that each word rhymes only with the other word in its pair, and with none of the words in other pairs.

You want to know the largest number of words that can be arranged into pairs in this way.

Input

The first line of the input gives the number of test cases, T. T test cases follow. Each test case starts with a line with single integer N. Then, N lines follow, each of which contains a string Wi of uppercase English letters, representing a distinct word. Notice that the same word can have different accentuations in different test cases.

Output

For each test case, output one line containing Case #x: y, where x is the test case number (starting from 1) and y is th size of the largest subset of words meeting the criteria described above.

Limits

$1 \leq T \leq 100$.
Time limit: 20 seconds per test set.
Memory limit: 1GB.
$1 \leq$ length of $Wi \leq 50$, for all i.
Wi consists of uppercase English letters, for all i.
$Wi \neq Wj$, for all $i \neq j$. (Words are not repeated within a test case.)

Test set 1 (Visible)

$2 \leq N \leq 6$.

Test set 2 (Hidden)

$2 \leq N \leq 1000$.

Sample

Input

Output

```
4
2
TARPOL
PROL
3
TARPOR
PROL
TARPRO
```

6
CODEJAM
JAM
HAM
NALAM
HUM
NOLOM
4
PI
HI
WI
FI

Case #1: 2
Case #2: 0
Case #3: 6
Case #4: 2

In Sample Case #1, the two words can rhyme with an appropriate accent assignment, as described above, so the larg st subset is the entire input.

In Sample Case #2, no two words can rhyme regardless of how we assign accents, because any two suffixes will diff r at least in the last letter. Therefore, the largest subset is the empty one, of size 0.

In Sample Case #3, we can use the entire set of words if we accentuate CODEJAM and JAM at the Js, HAM and N LAM at their last As and HUM and NOLOM at the Ms.

In Sample Case #4, any two words can be made to rhyme, but always by making the accented letter the I. Therefore, if we add two pairs to the subset, words from different pairs will rhyme. We can, thus, only form a subset of size 2, y choosing any 2 of the input words.

Solution:

```
/**
 *    author:  tourist
 *    created: 31.03.2018 11:41:48
**/
#include <bits/stdc++.h>

using namespace std;

int main() {
  ios::sync_with_stdio(false);
  cin.tie(0);
  int tt;
  cin >> tt;
  for (int qq = 1; qq <= tt; qq++) {
    cout << "Case #" << qq << ": ";
    int n;
```

```
  cin >> n;
  const int ALPHA = 26;
  vector<vector<int>> trie;
  trie.emplace_back(ALPHA, -1);
  vector<int> visits(1, 0);
  vector<int> pv(1, -1);
  while (n--) {
    string s;
    cin >> s;
    int t = 0;
    for (char c : string(s.rbegin(), s.rend())) {
      int d = (int) (c - 'A');
      if (trie[t][d] == -1) {
        trie[t][d] = (int) trie.size();
        trie.emplace_back(ALPHA, -1);
        visits.push_back(0);
        pv.push_back(t);
      }
      t = trie[t][d];
      visits[t]++;
    }
  }
  int ans = 0;
  for (int i = (int) trie.size() - 1; i >= 0; i--) {
    if (visits[i] < 2) {
      continue;
    }
    ans++;
    int v = i;
    while (v != -1) {
      visits[v] -= 2;
      v = pv[v];
    }
  }
  cout << 2 * ans << '\n';
 }
 return 0;
}
```