318. Maximum Product of Word Lengths

题目描述：https://leetcode.com/problems/maximum-product-of-word-lengths/

> 给定一个由n个字符串组成的数组w，求
> max(w[i].size() * w[j].size())
> 要求: w[i]和w[j]中不能有相同的元素。

解题思路1：

位操作
针对每一个字符串给定一个26位的00..00，分别代表a-z，然后如果某字符有则赋值为1,然后两两求异或，如果没有相同的1，则ok。

代码1：

```
class Solution {
public:
    int maxProduct(vector<string>& word) {
        vector<int> res(word.size(), pow(2,26));
        for(int i = 0; i < word.size(); i++){
            for(int j = i+1; j < word[i].size(); j++)
{
```

```
                int c = word[i].at(j) - 'a';
                int t = pow(2, c);
                res[i] |= t;
            }
        }
        int max = 0;
        for(int i = 0; i < word.size(); i++){
            for(int j = 0; j < word.size(); j++){
                int m = res[i] ^ res[j];
                int n = res[i] | res[j];
                if(m == n-pow(2,26)){
                    int t = word[i].size()*word[j].si
ze();
                    if(t > max){
                        max = t;
                    }
                }
            }
        }
        return max;
    }
};
```

解题思路2:

使用与操作

1 & 1 = 1

1 & 0 = 0

0 & 0 = 0

因此如果无重复的字母 a & b == 0

代码2:

```
class Solution {
public:
    int maxProduct(vector<string>& word) {
        vector<int> res(word.size(), 0);
        for(int i = 0; i < word.size(); i++){
            for(int j = 0; j < word[i].size(); j++){
                int c = word[i].at(j) - 'a';
                int t = pow(2, c);
                res[i] |= t;
            }
        }
        int max = 0;
        for(int i = 0; i < word.size(); i++){
            for(int j = i+1; j < word.size(); j++){
                int m = (res[i] & res[j]);
                int n = res[i] | res[j];
                if(!m){
                    int t = word[i].size()*word[j].si
ze();
                    if(t > max){
                        max = t;
                    }
                }
            }
        }
        return max;
    }
};
```