Given a string array words, find the maximum value of length(word[i]) \* length(word[j]) where the two words do not share common letters. You may assume that each word will contain only lower case letters. If no such two words exist, return 0.

**Example 1:**

Given ["abcw", "baz", "foo", "bar", "xtfn", "abcdef"]  
Return 16  
The two words can be "abcw", "xtfn".

**Example 2:**

Given ["a", "ab", "abc", "d", "cd", "bcd", "abcd"]  
Return 4  
The two words can be "ab", "cd".

**Example 3:**

Given ["a", "aa", "aaa", "aaaa"]  
Return 0  
No such pair of words.

师兄指点提交版

public class Solution {

public int maxProduct(String[] words) {

if (words.length == 0) return 0;

else {

int max = 0;

for (int i = 0;i < words.length;i++) {

for (int j = i+1; j < words.length;j++) {

if (max < words[i].length()\*words[j].length()){

if(!hasSameChar(words[i],words[j]))

max = words[i].length()\*words[j].length();

}

}

}

return max;

}

}

public boolean hasSameChar(String a,String b) {

char[] m = a.toCharArray();

char[] n = b.toCharArray();

int flag = 0 ;

int[] s = {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

int[] t = {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

for(int i = 0; i < m.length;i++) {

s[(int)(m[i]-'a')] = 1;

}

for(int i = 0; i < n.length;i++) {

t[(int)(n[i]-'a')] = 1;

}

for(int i = 0; i < s.length;i++){

if(s[i] == 1 && t[i] == 1) {

flag = 1;

break;

}

}

if (flag == 1) return true;

else return false;

}

}

带测试版本：

**public** **class** MaximumProductofWordLengths {

**public** **int** maxProduct(String[] words) {

**if** (words.length == 0) **return** 0;

**else** {

**int** max = 0;

**for** (**int** i = 0;i < words.length;i++) {

**for** (**int** j = i+1; j < words.length;j++) {

**if** (max < words[i].length()\*words[j].length()){

**if**(!hasSameChar(words[i],words[j])) {

max = words[i].length()\*words[j].length();

}

}

}

}

**return** max;

}

}

**public** **boolean** hasSameChar(String a,String b) {

**char**[] m = a.toCharArray();

**char**[] n = b.toCharArray();

**int** flag = 0 ;

**int**[] s = {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

**int**[] t = {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

**for**(**int** i = 0; i < m.length;i++) {

s[(**int**)(m[i]-'a')] = 1;

}

**for**(**int** i = 0; i < n.length;i++) {

t[(**int**)(n[i]-'a')] = 1;

}

**for**(**int** i = 0; i < s.length;i++) {

**if**(s[i] == 1 && t[i] == 1) {

flag = 1;

**break**;

}

}

**if** (flag == 1) **return** **true**;

**else** **return** **false**;

}

**public** **static** **void** main(String args[]) {

System.***out***.println(**new** MaximumProductofWordLengths().hasSameChar("asc", "def"));

System.***out***.println(**new** MaximumProductofWordLengths().hasSameChar("asc", "a"));

String [] words1 ={"abcw", "baz", "foo", "bar", "xtfn", "abcdef"};

String [] words2 ={"a", "ab", "abc", "d", "cd", "bcd", "abcd"};

String [] words3 ={"abcw","baz","foo","bar","xtfn","abcdef"};

System.***out***.println(**new** MaximumProductofWordLengths().maxProduct(words1));

System.***out***.println(**new** MaximumProductofWordLengths().maxProduct(words2));

System.***out***.println(**new** MaximumProductofWordLengths().maxProduct(words3));

}

}