class Solution {

public boolean isIsomorphic(String s, String t) {

if(s==null || t==null ||s.length()==0 || t.length()==0)

return false;

HashMap<Character,Character> smap = new HashMap<>();

HashMap<Character,Character> tmap = new HashMap<>();

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

if(!smap.containsKey(s.charAt(i)))

smap.put(s.charAt(i),t.charAt(i));

else{

if(smap.get(s.charAt(i))!=t.charAt(i))

return false;

}

if(!tmap.containsKey(t.charAt(i)))

tmap.put(t.charAt(i),s.charAt(i));

else{

if(tmap.get(t.charAt(i))!=s.charAt(i))

return false;

}

}

return true;

}

}

// time complexity : 0(N)

//space complexity:O(N)

class Solution {

public List<List<String>> groupAnagrams(String[] strs) {

if(strs==null)

return new ArrayList<>();

HashMap<Double,List<String>> hmap = new HashMap<>();

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

double a=primecal(strs[i]);

if(!hmap.containsKey(a))

hmap.put(a,new ArrayList<>());

hmap.get(a).add(strs[i]);

}

return new ArrayList<>(hmap.values());

}

private double primecal(String s){

int[] prime={2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61,67, 71, 73, 79, 83, 89, 97,101};

double res=1;

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

res\*= prime[s.charAt(i)-'a'];

}

return res;

}

}

// time complexity : 0(N)

//space complexity:O(N)