Leetcode 208前缀树

class Trie {

private class TrieNode{//每个节点最多有26个不同的小写字母

private boolean isEnd;

private TrieNode[] next;

public TrieNode(){

isEnd=false;

next=new TrieNode[26];

}

}

private TrieNode root;

/\*\* Initialize your data structure here. \*/

public Trie() {

root=new TrieNode();

}

/\*\* Inserts a word into the trie. \*/

public void insert(String word) {

TrieNode cur=root;

for(int i=0,len=word.length(),ch;i<len;i++){

ch=word.charAt(i)-'a';

if(cur.next[ch]==null)

cur.next[ch]=new TrieNode();

cur=cur.next[ch];

}

cur.isEnd=true;//加上标记，表示为一个单词

}

/\*\* Returns if the word is in the trie. \*/

public boolean search(String word) {

TrieNode cur=root;

for(int i=0,len=word.length(),ch;i<len;i++){

ch=word.charAt(i)-'a';

if(cur.next[ch]==null){

return false;

}

cur=cur.next[ch];

}

return cur.isEnd;

}

/\*\* Returns if there is any word in the trie that starts with the given prefix. \*/

public boolean startsWith(String prefix) {

TrieNode cur=root;

for(int i=0,len=prefix.length(),ch;i<len;i++){

ch=prefix.charAt(i)-'a';

if(cur.next[ch]==null){

return false;

}

cur=cur.next[ch];

}

return true;//直接返回true

}

}

/\*\*

\* Your Trie object will be instantiated and called as such:

\* Trie obj = new Trie();

\* obj.insert(word);

\* boolean param\_2 = obj.search(word);

\* boolean param\_3 = obj.startsWith(prefix);

\*/

Leetcode 547 省份数量

class Solution {

public int findCircleNum(int[][] isConnected) {

//使用DFS进行搜索

boolean[] visited =new boolean[isConnected.length];

int ret=0;

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

if(!visited[i]){

dfs(isConnected,visited,i);

ret++;

}

}

return ret;

}

private void dfs(int[][] m,boolean[] visited,int i){

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

if(m[i][j]==1 && !visited[j]){

visited[j]=true;

dfs(m,visited,j);

}

}

}

}