AC自动机 模板 （Keywords Search hdu 2222)

<http://acm.hdu.edu.cn/showproblem.php?pid=2222>

#include <iostream>  
#include <cstring>  
#include <cstdio>  
using namespace std;  
const int MAXN = 1000001; //模式串的最大长度MAXN - 1  
const int MAXM = 51; //单词最大长度为MAXM - 1  
const int KEYSIZE = 26; //26个小写字母  
struct Node {  
      Node \*fail;  //失败指针  
      Node \*next[KEYSIZE]; //儿子结点个数  
      int count; //单词个数  
      Node() {  
            fail = NULL;  
            count = 0;  
            memset(next, 0, sizeof(next));  
      }  
     ~Node() {  
          delete next;  
      }  
}\*q[MAXN / 2];  
void insert(char \*str, Node \*root)  
{  
      Node \*p = root;  
      int i = 0;  
      while(str[i]) {  
           int index = str[i] - 'a';  
           if(p -> next[index] == NULL)  
                  p -> next[index] = new Node();  
           p = p -> next[index];  
           i ++;  
      }  
      p -> count ++; //在单词的最后一个结点count + 1,代表一个单词  
}  
void build\_ac\_automation(Node \*root)  
{  
      root -> fail = NULL;  
      int head, tail;  
      head = tail = 0;  
      q[tail ++] = root;  
      while(head < tail) {  
            Node \*temp = q[head ++];  
            for(int i = 0; i < KEYSIZE; i ++) {  
                if(temp -> next[i] != NULL) {  
                     if(temp == root) {  
                          temp -> next[i] -> fail = root;  
                     }else {  
                          Node \*p = temp -> fail;  
                          while(p != NULL) {  
                               if(p -> next[i] != NULL) {  
                                         temp -> next[i] -> fail = p -> next[i];  
                                    break;  
                               }  
                               p = p -> fail;  
                          }  
                          if(p == NULL)  
                               temp -> next[i] -> fail = root;  
                     }  
                     q[tail ++] = temp -> next[i];  
                }  
           }  
      }  
}  
int query(char \*str, Node \*root)  
{  
      int i = 0, cnt = 0;  
      Node \*p = root;  
      while(str[i]) {  
           int index = str[i] - 'a';  
           while(p -> next[index] == NULL && p != root) p = p -> fail;  
           p = p -> next[index];  
           p = (p == NULL) ? root : p;  
           Node \*temp = p;  
           while(temp != root && temp -> count != -1) {  
                 cnt += temp -> count;  
                 temp -> count = -1;  
                 temp = temp -> fail;  
           }  
           i ++;  
      }  
      return cnt;  
}  
int main()  
{  
      int n, cas;  
      Node \*root;  
      char keyword[MAXM]; //单词  
      char str[MAXN]; //模式串  
      scanf("%d", &cas);  
      while(cas --) {  
            scanf("%d", &n);  
            getchar();  
            root = new Node();  
            while(n --) {  
                 scanf("%s", keyword);  
                 insert(keyword, root);  
            }  
            build\_ac\_automation(root);  
            scanf("%s", str);  
            printf("%d\n", query(str, root));  
      }  
      return(0);  
}