# string

## kmp

const int MAXN = 1e6 + 10;  
string str;  
string p;  
int n;  
int m;  
  
int nxt[MAXN];  
  
void make()  
{  
 // nxt(标号从[1,m]，m是匹配串长度)数组意义：nxt[i]是取出从1到i的前缀，对于这个前缀s，存在一个它的子串t，并且t != s，同时t既是s的前缀又是s的后缀，nxt[i] = max(|t|)。  
 nxt[0] = -1;  
 nxt[1] = 0;  
 for (int i = 0,k = -1;i < m;)  
 {  
 if (k == -1 || p[i] == p[k]) nxt[++ i] = ++ k;  
 else k = nxt[k];  
 }  
}  
  
int kmp()  
{  
 int ok = 0;  
 make();  
   
 //int f = 0;  
 for (int i = 0,j = 0;i < n;)  
 {  
 if (str[i] == p[j]) {  
 j ++;  
 i ++;  
 if (j == m) {  
 //printf("Found: %lu\n",i - m); // 找到一个p串在str串中的下标  
 ok ++;  
 //f = 1;  
 j = nxt[j];  
 }  
 }  
 else {  
 if (j == 0) i ++;  
 j = nxt[j];  
 }  
   
 }  
 //if (!f) printf("Not Found!\n");  
 return ok;  
}  
  
int main()  
{  
 cin >> str >> p;  
 n = (int) str.size();  
 m = (int) p.size();  
 printf("%d\n",kmp());  
 return 0;  
}

## AC automation

// AC自动机  
// 给定n个模式串s\_i和一个文本串t，求有多少个不同的模式串在文本串里出现过。两个模式串不同当且仅当他们编号不同。  
  
const int MAXN = 5e5 + 10;  
  
struct AC\_Auto {  
 int next[MAXN][26],fail[MAXN],end[MAXN];  
 int root,cnt;  
  
 inline int newNode() {  
 for (int i = 0;i < 26;i ++) next[cnt][i] = -1;  
 end[cnt ++] = 0;  
 return cnt - 1;  
 }  
 void init() {  
 cnt = 0;  
 root = newNode();  
 }  
  
 void insert(char str[]) {  
 int len = (int) strlen(str);  
 int now = root;  
 for (int i = 0;i < len;i++) {  
 if (next[now][str[i] - 'a'] == -1) next[now][str[i] - 'a'] = newNode();  
 now = next[now][str[i] - 'a'];  
 }  
 end[now] ++;  
 }  
  
 void build() {  
 queue<int> q;  
 fail[root] = root;  
 for(int i = 0;i < 26;i++)  
 {  
 if (next[root][i] == -1) next[root][i] = root;  
 else  
 {  
 fail[next[root][i]] = root;  
 q.push(next[root][i]);  
 }  
 }  
 while (!q.empty()) {  
 int now = q.front();  
 q.pop();  
 for (int i = 0;i < 26;i++)  
 {  
 if (next[now][i] == -1) next[now][i] = next[fail[now]][i];  
 else  
 {  
 fail[next[now][i]] = next[fail[now]][i];  
 q.push(next[now][i]);  
 }  
 }  
 }  
 }  
  
 int query(char \*s) {  
 int len = (int) strlen(s);  
 int now = 0,ans = 0;  
 for (int i = 0;i < len;i ++){  
 now = next[now][s[i] - 'a'];  
 for (int t = now;t && ~end[t];t = fail[t]) {  
 ans += end[t];  
 end[t] = -1;  
 }  
 }  
 return ans;  
 }  
  
 void debug() {  
 for (int i = 0;i < cnt;i ++) {  
 printf("id = %3d,fail = %3d,end = %3d,chi = [",i,fail[i],end[i]);  
 for (int j = 0;j < 26;j ++) printf("%2d",next[i][j]);  
 printf("]\n");  
 }  
 }  
};  
  
AC\_Auto ac;  
char str[MAXN << 1];  
  
int main()  
{  
 int n;  
 \_\_T {  
 scanf("%d",&n);  
 ac.init();  
 rep(i,1,n) {  
 scanf("%s",str);  
 ac.insert(str);  
 }  
 ac.build();  
 scanf("%s",str);  
 printf("%d\n",ac.query(str));  
 }  
 return 0;  
}