

原创 字符串模板

2019-10-17 10:42:46 _Y-_Y_ 阅读数 3 文章标签: ACM

[编辑](#)

版权声明：本文为博主原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接和本声明。

本文链接：https://blog.csdn.net/weixin_44410512/article/details/102601139

KMP

```
1  /*
2  https://www.luogu.org/problem/P3375
3  P3375
4  【模板】KMP 字符串匹配
5  */
6  #include<iostream>
7  #include<cstdio>
8  #include<queue>
9  #include<stack>
10 #include<cstring>
11 using namespace std;
12 #define maxn 1000005
13 char s[maxn];
14 char p[maxn];
15 int ans[maxn];
16 int next_[maxn];
17 int cnt;
18 int lens, lenp;
19 void build(){
20     next_[0]=-1;
21     int k=-1;
22     for(int i=1; i<lenp; i++){
23         while(k!=-1&&p[k+1]!=p[i]) k=next_[k];
24         if(p[k+1]==p[i]) k++;
25         next_[i]=k;
26     }
27 }
28 void kmp(){
29     cnt=0;
30     lenp=strlen(p);
31     lens=strlen(s);
32     build();
33     int j=-1;
34     for(int i=0; i<lens; i++){
35         while(j!=-1&&p[j+1]!=s[i]) j=next_[j];
36         if(p[j+1]==s[i]) j++;
37         if(j==lenp-1){
38             ans[cnt++]=i-lenp+2;
39             j=next_[j];
40         }
41     }
42 }
43 int main(){
44     scanf("%s%s", s, p);
45     kmp();
46     for(int i=0; i<cnt; i++){
47         printf("%d\n", ans[i]);
48     }
49     for(int i=0; i<lenp; i++){
50         printf("%d ", next_[i]+1);
51     }
52     return 0;
53 }
54
```

AC 自动机

IT 程序员转型学什么？

[关闭](#)

```

1  /*
2  https://www.luogu.org/problem/P3796
3  P3796
4  【模板】AC自动机（加强版）
5  */
6  #include <queue>
7  #include <cstdlib>
8  #include <cmath>
9  #include <cstdio>
10 #include <string>
11 #include <cstring>
12 #include <iostream>
13 #include <algorithm>
14 #include <map>
15 using namespace std;
16 char a[160][100];
17 const int maxn = 1*1e6+9;
18 int trie[maxn][26]; // 字典树
19 int fail[maxn]; // 失败时的回溯指针
20 int cnt = 0;
21 int ans[160];
22 int mp[maxn];
23 int anss;
24 void insertWords(char s[],int id){
25     int root = 0;
26     int len=strlen(s);
27     for(int i=0;i<len;i++){
28         int next = s[i] - 'a';
29         if(!trie[root][next])
30             trie[root][next] = ++cnt;
31         root = trie[root][next];
32     }
33     mp[root]=id;
34 }
35 void getFail(){
36     queue <int>q;
37     for(int i=0;i<26;i++){
38         if(trie[0][i]){
39             fail[trie[0][i]] = 0;
40             q.push(trie[0][i]);
41         }
42     }
43     while(!q.empty()){
44         int now = q.front();
45         q.pop();
46         for(int i=0;i<26;i++){
47             if(trie[now][i]){
48                 fail[trie[now][i]] = trie[fail[now]][i];
49                 q.push(trie[now][i]);
50             }
51             else{
52                 trie[now][i] = trie[fail[now]][i];
53             }
54         }
55     }
56 }
57 void query(char s[]){
58     int now = 0;
59     int len=strlen(s);
60     for(int i=0;i<len;i++){
61         now = trie[now][s[i]-'a'];
62         for(int j=now;j=fail[j]){
63             ans[mp[j]] ++;
64         }
65     }
66 }
67 char s[maxn];
68 int main() {
69     int n;
70     while(~scanf("%d", &n)&& n){

```

```

71     memset(ans,0,sizeof ans);
72     memset(fail,0,sizeof fail);
73     memset(trie,0,sizeof trie);
74     memset(mp,0,sizeof mp);
75     cnt=0;
76     anss=0;
77     for(int i=1;i<=n;i++){
78         scanf("%s", a[i]);
79         insertWords(a[i],i);
80     }
81     getFail();
82     scanf("%s", s);
83     query(s);
84     for(int i=1;i<=n;i++){
85         if(ans[i]>anss) anss=ans[i];
86     }
87     printf("%d\n", anss);
88     for(int i=1;i<=n;i++){
89         if(ans[i]==anss){
90             printf("%s\n", a[i]);
91         }
92     }
93 }
94 return 0;
95 }
96

```

后缀自动机

```

1  /*
2  Glass Beads
3  POJ - 1509
4  https://cn.vjudge.net/problem/POJ-1509#author=hzo12017_gyz
5  题意： 给一个字符串S，每次可以将它的第一个字符移到后面，求这样能得到的字典序最小的字符串。 如BBAAB，最小的就是AABBB
6  解法： 后缀自动机
7  */
8  #include <iostream>
9  #include <algorithm>
10 #include <cstring>
11 #define maxn 100000
12 using namespace std;
13 struct node{
14     int len,fail;
15     int next[26];
16     void init(){
17         fail=-1;
18         len=0;
19         memset(next,-1,sizeof next);
20     }
21 }a[maxn];
22 int top,last;
23 char s[maxn];
24 void init(){
25     top=last=0;
26     a[top++].init();
27 }
28 int newnode(){
29     a[top].init();
30     return top++;
31 }
32 void add(int c){
33     int end=newnode();
34     int now=last;
35     a[end].len=a[now].len+1;
36     for(;now!=-1&&a[now].next[c]==-1; now=a[now].fail){
37         a[now].next[c]=end;
38     }
39     if(now==-1) a[end].fail=0;
40     else{
41         int next=a[now].next[c];

```

```

42     if(a[now].len+1==a[next].len) a[end].fail=next;
43     else{
44         int np = newnode();
45         a[np]=a[next];
46         a[np].len=a[now].len+1;
47         a[end].fail=a[next].fail = np;
48         for(;now!=-1&&a[now].next[c]==next;now=a[now].fail){
49             a[now].next[c] = np;
50         }
51     }
52 }
53 last = end;
54 }
55 int main(){
56     int T;
57     cin>>T;
58     while(T--){
59         init();
60         cin>>s;
61         int len=strlen(s);
62         for(int i=0;i<len*2;i++){
63             add(s[i%len]-'a');
64         }
65         int ans=0;
66         for(int i=0;i<len;i++){
67             for(int j=0;j<26;j++){
68                 if(a[ans].next[j]!=-1){
69                     ans=a[ans].next[j];
70                     break;
71                 }
72             }
73         }
74         cout<<a[ans].len-len+1<<endl;
75     }
76 }
77

```

回文自动机

```

1  /*
2  回文串
3  HYSBZ - 3676
4  https://cn.vjudge.net/problem/HYSBZ-3676
5  题意：给你一个字符串 s，求回文子串长度 * 该回文串子出现次数的最大值。
6  解法：回文树+拓扑序
7  */
8  #include<bits/stdc++.h>
9  using namespace std;
10 #define ll long long
11 const int N = 3e5+100;
12 struct node
13 {
14     int len, cnt;//该回文串的长度，该回文串出现的次数
15     int next[26];//只有26个小写字母
16     int fail;//指向等于最最长后缀回文的前缀点
17 }a[N];
18 int top, last;
19 char s[N];
20 void Init_Tr()//初始化很重要
21 {
22     top = 1, last = 0;
23     a[0].len = 0, a[1].len = -1;
24     a[0].fail = 1;
25 }
26 int i;//减少传参可以优化很大的时间复杂度
27 int get_id(int now)
28 {
29     while(s[i] != s[i-a[now].len-1]) now = a[now].fail;//判断是否满足回文
30     return now;
31 }

```

```

32 void Insert()
33 {
34     int len = strlen(s+1);
35     for(i = 1; i <= len; i++) {
36         int t = s[i]-'a';
37         int id = get_id(last);
38         if(!a[id].next[t]) {
39             a[++top].len = a[id].len + 2; // 每次前后各增加一个点
40             a[top].fail = a[get_id(a[id].fail)].next[t];
41             a[id].next[t] = top;
42         }
43         last = a[id].next[t]; //
44         a[last].cnt++;
45     }
46 }
47 ll solve()
48 {
49     ll ans = 0;
50     for(int i = top; i >= 2; i--) { // 从后往前遍历相当于拓扑序
51         a[a[i].fail].cnt += a[i].cnt;
52         ans = max(ans, 1LL*a[i].cnt*a[i].len);
53     }
54     return ans;
55 }
56 int main()
57 {
58     scanf("%s", s+1);
59     Init_Tr();
60     Insert();
61     printf("%lld\n", solve());
62     return 0;
63 }
64
65

```

马拉车

```

1  /*
2  最长回文
3  HDU - 3068
4  https://cn.vjudge.net/problem/HDU-3068
5  题面: 最长回文长度
6  解法: 马拉车算法
7  */
8  #include <bits/stdc++.h>
9  using namespace std;
10 #define maxn 300005
11 int radius[maxn];
12 char s[maxn];
13 char s_new[maxn];
14 int init(){
15     int len=strlen(s);
16     s_new[0]='$';
17     s_new[1]='#';
18     int j=2;
19     for(int i=0;i<len;i++){
20         s_new[j++]=s[i];
21         s_new[j++]='#';
22     }
23     s_new[j]='\0';
24     return j;
25 }
26 int Manacher(){
27     int len=init();
28     int ans=-1;
29     int id;
30     int mx=0;
31     for(int i=1;i<len;i++){
32         if(i<mx) radius[i]=min(radius[2 * id - i], mx - i);
33         else radius[i]=1;

```

```
34     while(s_new[i-radius[i]]==s_new[i+radius[i]]){
35         radius[i]++;
36     }
37     if(mx<i+radius[i]){
38         id=i;
39         mx=i+radius[i];
40     }
41     ans=max(ans,radius[i]-1);
42 }
43 return ans;
44 }
45 int main(){
46     while(~scanf("%s", s)){
47         printf("%d\n", Manacher());
48     }
49 }
50
51
```

文章最后发布于: 2019-10-17 10:45:36

有 0 个人打赏

©2019 CSDN 皮肤主题: 大白 设计师: CSDN官方博客

IT
程序员
转型
学什么？

关闭