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

KMP

```
/*
https://www.luogu.org/problem/P3375
P3375
【模板】KMP字符串匹配
*/
#include<iostream>
#include<cstdio>
#include<queue>
#include<stack>
#include<cstring>
using namespace std;
#define maxn 1000005
char s[maxn];
char p[maxn];
int ans[maxn];
int next_[maxn];
int cnt;
int lens,lenp;
void build(){
  next_[0]=-1;
  int k=-1;
  for(int i=1;i<lenp;i++){
    while(k!=-1&&p[k+1]!=p[i]) k=next_[k];
    if(p[k+1]==p[i]) k++;
    next_[i]=k;
  }
}
void kmp(){
  cnt=0;
  lenp=strlen(p);
  lens=strlen(s);
  build();
  int j=-1;
  for(int i=0;i<lens;i++){
    while(j!=-1&&p[j+1]!=s[i]) j=next_[j];
    if(p[j+1]==s[i]) j++;
    if(j==lenp-1){
      ans[cnt++]=i-lenp+2;
      j=next_[j];
    }
  }
}
int main(){
  scanf("%s%s", s, p);
  kmp();
  for(int i=0;i<cnt;i++){
    printf("%d\n", ans[i]);
  }
  for(int i=0;i<lenp;i++){
    printf("%d ", next_[i]+1);
  }
  return 0;
}

```

AC 自动机

```cpp
/*
https://www.luogu.org/problem/P3796
P3796
【模板】AC自动机（加强版）
*/
#include <queue>
#include <cstdlib>
#include <cmath>
#include <cstdio>
#include <string>
#include <cstring>
#include <iostream>
#include <algorithm>
#include <map>
using namespace std;
char a[160][100];
const int maxn =  1*1e6+9;
int trie[maxn][26]; //字典树
int fail[maxn];     //失败时的回溯指针
int cnt = 0;
int ans[160];
int mp[maxn];
int anss;
void insertWords(char s[],int id){
    int root = 0;
    int len=strlen(s);
    for(int i=0;i<len;i++){
        int next = s[i] - 'a';
        if(!trie[root][next])
            trie[root][next] = ++cnt;
        root = trie[root][next];
    }
    mp[root]=id;
}
void getFail(){
    queue <int>q;
    for(int i=0;i<26;i++){
        if(trie[0][i]){
            fail[trie[0][i]] = 0;
            q.push(trie[0][i]);
        }
    }
    while(!q.empty()){
        int now = q.front();
        q.pop();
        for(int i=0;i<26;i++){
            if(trie[now][i]){
                fail[trie[now][i]] = trie[fail[now]][i];
                q.push(trie[now][i]);
            }
            else{
                trie[now][i] = trie[fail[now]][i];
            }
        }
    }
}
void query(char s[]){
    int now = 0;
    int len=strlen(s);
    for(int i=0;i<len;i++){
        now = trie[now][s[i]-'a'];
        for(int j=now;j;j=fail[j]){
            ans[mp[j]] ++;
        }
    }
}
char s[maxn];
int main() {
    int n;
    while(~scanf("%d", &n)&& n){
```

```
        memset(ans,0,sizeof ans);
        memset(fail,0,sizeof fail);
        memset(trie,0,sizeof trie);
        memset(mp,0,sizeof mp);
        cnt=0;
        anss=0;
        for(int i=1;i<=n;i++){
            scanf("%s", a[i]);
            insertWords(a[i],i);
        }
        getFail();
        scanf("%s", s);
        query(s);
        for(int i=1;i<=n;i++){
          if(ans[i]>anss) anss=ans[i];
        }
        printf("%d\n", anss);
        for(int i=1;i<=n;i++){
          if(ans[i]==anss){
            printf("%s\n", a[i]);
          }
        }
    }
    return 0;
}

```


后缀自动机

```
/*
Glass Beads
POJ - 1509
https://cn.vjudge.net/problem/POJ-1509#author=hzoi2017_gyz
题意：给一个字符串S，每次可以将它的第一个字符移到最后面，求这样能得到的字典序最小的字符串。 如BBAAB，最小的就是AABBB
解法：后缀自动机
*/
#include <iostream>
#include <algorithm>
#include <cstring>
#define maxn 100000
using namespace std;
struct node{
  int len,fail;
  int next[26];
  void init(){
    fail=-1;
    len=0;
    memset(next,-1,sizeof next);
  }
}a[maxn];
int top,last;
char s[maxn];
void init(){
  top=last=0;
  a[top++].init();
}
int newnode(){
  a[top].init();
  return top++;
}
void add(int c){
  int end=newnode();
  int now=last;
  a[end].len=a[last].len+1;
  for(;now!=-1&&a[now].next[c]==-1; now=a[now].fail){
    a[now].next[c]=end;
  }
  if(now==-1) a[end].fail=0;
  else{
    int next=a[now].next[c];
```

```
    if(a[now].len+1==a[next].len) a[end].fail=next;
    else{
      int np = newnode();
      a[np]=a[next];
      a[np].len=a[now].len+1;
      a[end].fail=a[next].fail = np;
      for(;now!=-1&&a[now].next[c]==next;now=a[now].fail){
        a[now].next[c] = np;
      }
    }
  }
  last = end;
}
int main(){
  int T;
  cin>>T;
  while(T--){
    init();
    cin>>s;
    int len=strlen(s);
    for(int i=0;i<len*2;i++){
      add(s[i%len]-'a');
    }
    int ans=0;
    for(int i=0;i<len;i++){
      for(int j=0;j<26;j++){
        if(a[ans].next[j]!=-1){
          ans=a[ans].next[j];
          break;
        }
      }
    }
    cout<<a[ans].len-len+1<<endl;
  }
}

```


回文自动机

```
/*
回文串
HYSBZ - 3676
https://cn.vjudge.net/problem/HYSBZ-3676
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。
解法：回文树+拓扑序
*/
#include<bits/stdc++.h>
using namespace std;
#define ll long long
const int N = 3e5+100;
struct node
{
    int len, cnt;//该回文串的长度，该回文串出现的次数
    int next[26];//只有26个小写字母
    int fail;//指向等于最长后缀回文的前缀点
}a[N];
int top, last;
char s[N];
void Init_Tr()//初始化很重要
{
    top = 1, last = 0;
    a[0].len = 0, a[1].len = -1;
    a[0].fail = 1;
}
int i;//减少传参可以优化很大的时间复杂度
int get_id(int now)
{
    while(s[i] != s[i-a[now].len-1]) now = a[now].fail;//判断是否满足回文
    return now;
}
```

```
void Insert()
{
    int len = strlen(s+1);
    for(i = 1; i <= len; i++) {
        int t = s[i]-'a';
        int id = get_id(last);
        if(!a[id].next[t]) {
            a[++top].len = a[id].len + 2;//每次前后各增加一个点
            a[top].fail = a[get_id(a[id].fail)].next[t];
            a[id].next[t] = top;
        }
        last = a[id].next[t];//
        a[last].cnt++;
    }
}
ll solve()
{
    ll ans = 0;
    for(int i = top; i >= 2; i--) {//从后往前遍历相当于拓扑序
        a[a[i].fail].cnt += a[i].cnt;
        ans = max(ans, 1LL*a[i].cnt*a[i].len);
    }
    return ans;
}
int main()
{
    scanf("%s", s+1);
    Init_Tr();
    Insert();
    printf("%lld\n", solve());
    return 0;
}


```


马拉车

```
/*
最长回文
HDU - 3068
https://cn.vjudge.net/problem/HDU-3068
题面：最长回文长度
解法：马拉车算法
*/
#include <bits/stdc++.h>
using namespace std;
#define maxn 300005
int radius[maxn];
char s[maxn];
char s_new[maxn];
int init(){
  int len=strlen(s);
  s_new[0]='$';
  s_new[1]='#';
  int j=2;
  for(int i=0;i<len;i++){
    s_new[j++]=s[i];
    s_new[j++]='#';
  }
  s_new[j]='\0';
  return j;
}
int Manacher(){
  int len=init();
  int ans=-1;
  int id;
  int mx=0;
  for(int i=1;i<len;i++){
    if(i<mx) radius[i]=min(radius[2 * id - i], mx - i);
    else radius[i]=1;
```

```
    while(s_new[i-radius[i]]==s_new[i+radius[i]]){
      radius[i]++;
    }
    if(mx<i+radius[i]){
      id=i;
      mx=i+radius[i];
    }
    ans=max(ans,radius[i]-1);
  }
  return ans;
}
int main(){
  while(~scanf("%s", s)){
    printf("%d\n", Manacher());
  }
}
```

文章最后发布于: 2019-10-17 10:45:36