

# The event of zero and one

## ACM Template



目录

1.1 Manacher . . . . . 3

1 String 3

# 1 String

## 1.1 Manacher

```

302f #include<bits/stdc++.h>
421c using namespace std;
571f const int MAX = 2e5+10000;
99d0 char s[MAX];
81d4 struct Manacher{
9ccd     int lc[MAX];
04f3     char ch[MAX];
d7af     int N;
053c Manacher(char *s){init(s);manacher();}
44ca     /* s 1 bas */
e798     void init(char *s){
0de8         int n = strlen(s+1);
ad19         ch[n*2 +1] = '#';
ce0d         ch[0] = '@';
46cd         ch[n*2 +2] = '\0';
0c3f         for (int i=n;i>=1;i--){
6beb             ch[i*2] = s[i];ch[i*2 -1] = '#';
95cf         }
5991         N = 2* n +1;
95cf     }
     void manacher(){
         lc[1]=1; int k=1;
         for (int i=2;i<=N;i++){
             int p = k+lc[k]-1;
             if (i<=p){
                 lc[i]=min(lc[2*k-i],p-i+1);
             }else{ lc[i]=1; }
             while (ch[i+lc[i]]==ch[i-lc[i]])lc[i]++;
             if (i+lc[i]>k+lc[k])k=i;
         }
     }
     void debug(){
         puts(ch);
         for (int i=1;i<=N;i++){
             printf("lc[%d]=%d\n",i,lc[i]);
         }
     }
};
int main(){

```

```

scanf("%s",s+1);
Manacher manacher(s);
manacher.debug();
return 0;
}

```

a275  
382e  
9c07  
7021  
95cf

## 2 Math

### 2.1 Linear sieve

```

302f #include<bits/stdc++.h>
421c using namespace std;
68e4 const int maxn = 1e7+10;
4085 typedef long long ll;
727f bool used[maxn];
efe5 int mu[maxn];
7c8f vector<int> prime;
c882 ll f[maxn];
a0b1 int low[maxn];
22c5 void sieve(int size){
427e     //f:multiplicative function;
7d97     assert(size < maxn);
7f5a     mu[1] = 1;
c6b9     6c5fff[1] = 1;
40bd     for (int i=2;i<=size;i++){
efb1         256b         if (!used[i]){
1024             7957             prime.push_back(i);
7171             5e04             mu[i] = -1;
427e             24a1             //f:TODO
c21b             87d6             low[i] = i;
95cf             aa80         }
eb1a             2b9a         for (int j = 0;j < prime.size();j++){
d3c2             95cf             ll nxt = 1ll * i * prime[j];
b561             95cf             if (nxt > size)break;
6b89             56dd             used[nxt] = 1;
073a             b492             if (i % prime[j]){
b9b8                 cd0f                 low[nxt] = prime[j];
66f9                 0d62                 mu[nxt] = -mu[i];
427e             95cf                 //f: mod or not?
7225             95cf                 f[nxt] = f[i] * f[prime[j]];
8e2e             329b             }else{
734b                 3117                 low[nxt] = prime[j] * low[i];

```

```

8ec3      mu[nxt] = 0;
b401      if (low[nxt] != nxt){
427e          //mod or not?
4d18          f[nxt] = 1ll * f[low[nxt]] * f[nxt/low[nxt]];
8e2e      }else{
427e          // i = prime[j] ^ k
427e          //f:TODO
95cf      }
6173      break;

```

```

95cf      }
95cf      }
95cf      }
95cf      }
3117 int main(){
ff91     sieve(1e7);
7021     return 0;
95cf }

```