泡泡猿 ACM 模板

Rand0w & REXWIND & Dallby

2021 年 10 月 3 日

目录

1 头文件

1.1 头文件 (Rand0w)

```cpp
#include <bits/stdc++.h>
//#include <bits/extc++.h>
//using namespace __gnu_pbds;
//using namespace __gnu_cxx;
using namespace std;
#pragma optimize(2)
//#pragma GCC optimize("Ofast,no-stack-protector")
//#pragma GCC target("sse,sse2,sse3,ssse3,sse4,popcnt,abm,mmx,avx,avx2,tune=native")
#define rbset(T) tree<T,null_type,less<T>,rb_tree_tag,tree_order_statistics_node_update>
const int inf = 0x7FFFFFFF;
typedef long long ll;
typedef double db;
typedef long double ld;
template<class T>inline void MAX(T &x,T y){if(y>x)x=y;}
template<class T>inline void MIN(T &x,T y){if(y<x)x=y;}
namespace FastIO
{
char buf[1 << 21], buf2[1 << 21], a[20], *p1 = buf, *p2 = buf, hh = '\n';
int p, p3 = -1;
void read() {}
void print() {}
inline int getc()
{
return p1 == p2 && (p2 = (p1 = buf) + fread(buf, 1, 1 << 21, stdin), p1 == p2) ? EOF : *p1++;
}
inline void flush()
{
fwrite(buf2, 1, p3 + 1, stdout), p3 = -1;
}
template <typename T, typename... T2>
inline void read(T &x, T2 &... oth)
{
int f = 0;x = 0;char ch = getc();
while (!isdigit(ch)){if (ch == '-')f = 1;ch = getc();}
while (isdigit(ch)){x = x * 10 + ch - 48;ch = getc();}
x = f ? -x : x;read(oth...);
}
template <typename T, typename... T2>
inline void print(T x, T2... oth)
{
if (p3 > 1 << 20)flush();
if (x < 0)buf2[++p3] = 45, x = -x;
do{a[++p] = x % 10 + 48;}while (x /= 10);
do{buf2[++p3] = a[p];}while (--p);
buf2[++p3] = hh;
print(oth...);
}
} // namespace FastIO
#define read FastIO::read
#define print FastIO::print
#define flush FastIO::flush
#define spt fixed<<setprecision
#define endll '\n'
#define mul(a,b,mod) (__int128)(a)*(b)%(mod)
#define pii(a,b) pair<a,b>
#define pow powmod
#define X first
#define Y second
#define lowbit(x) (x&-x)
#define MP make_pair
#define pb push_back
#define pt putchar
#define yx_queue priority_queue
#define lson(pos) (pos<<1)
#define rson(pos) (pos<<1|1)
#define y1 code_by_Rand0w
#define yn A_muban_for_ACM
#define j1 it_is just_an_eastegg
#define lr hope_you_will_be_happy_to_see_this
#define int long long
#define rep(i, a, n) for (register int i = a; i <= n; ++i)
#define per(i, a, n) for (register int i = n; i >= a; --i)
const ll llinf = 4223372036854775851;
const ll mod = (0 ? 1000000007 : 998244353);
ll pow(ll a,ll b,ll md=mod) {ll res=1;a%=md; assert(b>=0); for(;b;b>>=1){if(b&1)res=mul(res,a,md);a=mul(a,a,md);}return res;}
const ll mod2 = 999998639;
const int m1 = 998244353;
const int m2 = 1000001011;
const int pr=233;
const double eps = 1e-7;
const int maxm= 1;
const int maxn = 510000;
void work()
{

}
signed main()
{
   #ifndef ONLINE_JUDGE
   //freopen("in.txt","r",stdin);
    //freopen("out.txt","w",stdout);
#endif
    //std::ios::sync_with_stdio(false);
    //cin.tie(NULL);
    int t = 1;
    //cin>>t;
    for(int i=1;i<=t;i++){
        //cout<<"Case #"<<i<<":"<<endll;
        work();
    }
    return 0;
}
```

1.2 头文件 (REXWind)

```
#include<iostream>
#include<cstring>
#include<cstdio>
#include<algorithm>
#include<vector>
#include<map>
#include<queue>
#include<cmath>
using namespace std;

template<class T>inline void read(T &x){x=0;char o,f=1;while(o=getchar(),o<48)if(o==45)f=-f;do x=(x<<3)+(x<<1)+(o^48);while(o=getchar(),o>47);x*=f;}
int cansel_sync=(ios::sync_with_stdio(0),cin.tie(0),0);
#define ll long long
#define ull unsigned long long
#define rep(i,a,b) for(int i=(a);i<=(b);i++)
#define repb(i,a,b) for(int i=(a);i>=b;i--)
#define mkp make_pair
#define ft first
#define sd second
#define log(x) (31-__builtin_clz(x))
#define INF 0x3f3f3f3f
typedef pair<int,int> pii;
typedef pair<ll,ll> pll;
ll gcd(ll a,ll b){ while(b^=a^=b^=a%=b); return a; }
//#define INF 0x7fffffff

void solve(){

}

int main(){
   int z;
   cin>>z;
   while(z--) solve();
}
```

1.3 头文件 (Dallby)

```
#include<bits/stdc++.h>
cout<<"hello<<endl;
```

2 数论

2.1 欧拉筛

$O(n)$ 筛素数

```
int primes[maxn+5],tail;
bool is_prime[maxn+5];
void euler(){
  is_prime[1] = 1;
  for (int i = 2; i < maxn; i++)
  {
    if (!is_prime[i])
    primes[++tail]=i;
    for (int j = 0; j < primes.size() && i * primes[j] < maxn; j++)
    {
      is_prime[i * primes[j]] = 1;
      if ((i % primes[j]) == 0)
        break;
    }
  }
}
```

2.2 Exgcd

求出 $ax + by = gcd(a, b)$ 的一组可行解 $O(logn)$

```
void Exgcd(ll a,ll b,ll &d,ll &x,ll &y){
   if(!b){d=a;x=1;y=0;}
   else{Exgcd(b,a%b,d,y,x);y-=x*(a/b);}
}
```

2.3 Excrt 扩展中国剩余定理

求解同余方程组 $\begin{cases} x \ \% \ b_1 \equiv a_1 \\ x \ \% \ b_2 \equiv a_2 \\ \vdots \\ x \ \% \ b_n \equiv a_n \end{cases}$

```
int excrt(int a[],int b[],int n){
   int lc=1;
   for(int i=1;i<=n;i++)
      lc=lcm(lc,a[i]);
   for(int i=1;i<n;i++){
      int p,q,g;
      g=exgcd(a[i],a[i+1],p,q);
      int k=(b[i+1]-b[i])/g;
      q=-q;p*=k;q*=k;
      b[i+1]=a[i]*p%lc+b[i];
      b[i+1]%=lc;
      a[i+1]=lcm(a[i],a[i+1]);
   }
   return (b[n]%lc+lc)%lc;
}
```

2.4 线性求逆元

```
void init(int p){
   inv[1] = 1;
   for(int i=2;i<=n;i++){
      inv[i] = (ll)(p-p/i)*inv[p%i]%p;
   }
}
```

2.5 组合数

预处理阶乘，并通过逆元实现相除

```
ll jc[MAXN];
ll qpow(ll d,ll c){//快速幂
   ll res = 1;
   while(c){
      if(c&1) res=res*d%med;
      d=d*d%med;c>>=1;
   }return res;
}
inline ll niyuan(ll x){return qpow(x,med-2);}
```

```cpp
void initjc(){//初始化阶乘
    jc[0] = 1;
    rep(i,1,MAXN-1) jc[i] = jc[i-1]*i%med;
}
inline int C(int n,int m){//n是下面的
    if(n<m) return 0;
    return jc[n]*niyuan(jc[n-m])%med*niyuan(jc[m])%med
        ;
}
int main(){
    initjc();
    int n,m;
    while(cin>>n>>m) cout<<C(n,m)<<endl;
}
```

2.6 矩阵快速幂

```cpp
struct Matrix{
    ll a[MAXN][MAXN];

    Matrix(ll x=0){//感觉是特别好的初始化,从hjt那里学(抄
        )来的
        for(int i=0;i<n;i++){
            for(int j=0;j<n;j++){
                a[i][j]=x*(i==j);//这句特简洁
            }
        }
    }

    Matrix operator *(const Matrix &b)const{//通过重载
        运算符实现矩阵乘法
        Matrix res(0);
        for(int i=0;i<n;i++){
            for(int j=0;j<n;j++){
                for(int k=0;k<n;k++){
                    ll &ma = res.a[i][j];
                    ma = (ma+a[i][k]*b.a[k][j])%mod;
                }
            }
        }
        return res;
    }
};

Matrix qpow(Matrix d,ll m){//底数和幂次数
    Matrix res(1);//构造E单位矩阵
    while(m){
        if(m&1){
            m--;//其实这句是可以不要的
            res=res*d;
        }
        d=d*d;
        m>>=1;
    }
    return res;
}
```

2.7 高斯消元

```
待补充
```

2.8 三点求圆心

```cpp
struct point{
    double x;
    double y;
};

point cal(point a,point b,point c){
    double x1 = a.x;double y1 = a.y;
    double x2 = b.x;double y2 = b.y;
    double x3 = c.x; double y3 = c.y;
    double a1 = 2*(x2-x1); double a2 = 2*(x3-x2);
    double b1 = 2*(y2-y1); double b2 = 2*(y3-y2);
    double c1 = x2*x2 + y2*y2 - x1*x1 - y1*y1;
    double c2 = x3*x3 + y3*y3 - x2*x2 - y2*y2;
    double rx = (c1*b2-c2*b1)/(a1*b2-a2*b1);
    double ry = (c2*a1-c1*a2)/(a1*b2-a2*b1);
    return point{rx,ry};
}
```

2.9 欧拉降幂

$$a^b \equiv \begin{cases} a^{b\%\phi(p)}, & \gcd(a,p)=1 \\ a^b, & \gcd(a,p)\neq 1, b<\phi(p) \\ a^{b\%\phi(p)+\phi(p)}, & \gcd(a,p)\neq 1, b\geq\phi(p) \end{cases} \pmod p$$

3 字符串

3.1 FFT 解决字符串匹配问题

可以用来解决含有通配符的字符串匹配问题定义匹配函数

$$(x,y)=(A_x-B_x)^2$$

如果两个字符相同，则满足 $C(x,y)=0$
定义模式串和文本串 x 位置对齐时候的完全匹配函数为

$$P(x)=\sum C(i,x+i)$$

模式串在位置 x 上匹配时,$p(x)=0$
通过将模式串 reverse 后卷积，可以快速处理每个位置 x 上的完全匹配函数 $P(x)$ 同理，如果包含通配符，则设通配符的值为 0，可以构造损失函数

$$C(x,y)=(A_x-B_x)^2\cdot A_x\cdot B_x=A_x^3B_x+A_xB_x^3-2A_x^2B_x^2$$

通过三次 FFT 即可求得每个位置上的 P(x)

3.2 后缀数组 SA+LCP

LCP(i,j) 后缀 i 和后缀 j 的最长公共前缀

```cpp
int n,m;
string s;
int rk[MAXN],sa[MAXN],c[MAXN],rk2[MAXN];
//sa[i]存排名i的原始编号 rk[i]存编号i的排名 第二关键字
    rk2
inline void get_SA(){
    rep(i,1,n) ++c[rk[i]=s[i]];//基数排序
    rep(i,2,m) c[i] += c[i-1];
    //c做前缀和, 可以知道每个关键字的排名最低在哪里
    repb(i,n,1) sa[c[rk[i]]--] = i;//记录每个排名的原编
        号

    for(int w=1;w<=n;w<<=1){//倍增
```

```
        int num = 0;
        rep(i,n-w+1,n) rk2[++num] = i;//没有第二关键字的排在前面
        rep(i,1,n) if(sa[i]>w) rk2[++num] = sa[i]-w;
        //编号sa[i]大于w的才能作为编号sa[i]-w的第二关键字
        rep(i,1,m) c[i] = 0;
        rep(i,1,n) ++c[rk[i]];
        rep(i,2,m) c[i]+=c[i-1];
        repb(i,n,1) sa[c[rk[rk2[i]]]--]=rk2[i],rk2[i]=0;
        //同一个桶中按照第二关键字排序
        swap(rk,rk2);
        //这时候的rk2时这次排序用到的上一轮的rk,要计算出新的rk给下一轮排序

        rk[sa[1]]=1,num=1;
        rep(i,2,n)
            rk[sa[i]] = (rk2[sa[i]]==rk2[sa[i-1]]&&rk2[sa[i]+w]==rk2[sa[i-1]+w])?num:++num;
        //下一次排名的第一关键字,相同的两个元素排名也相同
        if(num==n) break;//rk都唯一时, 排序结束
        m=num;
    }
}
int height[MAXN];
inline void get_height(){
    int k = 0,j;
    rep(i,1,n) rk[sa[i]] = i;
    rep(i,1,n){
        if(rk[i]==1) continue;//第一名往前没有前缀
        if(k) k--;//h[i]>=h[i-1]-1 即height[rk[i]]>=height[rk[i-1]]-1
        j = sa[rk[i]-1];//找排在rk[i]前面的
        while(j+k<=n&&i+k<=n&&s[i+k]==s[j+k]) ++k;//逐字符比较
        //因为每次k只会-1, 故++k最多只会加2n次
        height[rk[i]] = k;
    }
}
inline void solve(){
    cin>>s;
    s = ' '+s;
    n = s.size()-1,m = 122;//m为字符个数'z'=122
    get_SA();
    rep(i,1,n) cout<<sa[i]<<' ';
    cout<<endl;
}
```

4 STL 等小技巧

4.1 集合 set

还可以通过 lower_bound 和 upper_bound 返回迭代器来找前驱，后继

```
//并交集
vector<int> ANS;
set_union(s1.begin(),s1.end(),s2.begin(),s2.end(),inserter(ANS,ANS.begin()));//set_intersection()

//通过迭代器遍历集合
set<char>::iterator iter = temp1.begin();
while (iter!=temp1.end()){
    cout<<*iter;
    iter++;
}
```

4.2 快读快写 (短)

```
template<class T>inline void read(T &x){x=0;char o,f=1;while(o=getchar(),o<48)if(o==45)f=-f;do x=(x<<3)+(x<<1)+(o^48);while(o=getchar(),o>47);x*=f;}
template<class T>
void wt(T x){//快写
  if(x < 0) putchar('-'), x = -x;
  if(x >= 10) wt(x / 10);
  putchar('0' + x % 10);
}
```

4.3 GCD(压行)

```
ll gcd(ll a,ll b){ while(b^=a^=b^=a%=b); return a; }
```

4.4 计时

```
inline double run_time(){
    return 1.0*clock()/CLOCKS_PER_SEC;
}
```