比赛链接

题解

A. Island Travels

题解

bfs 求一下连通块，再 bfs 求一下最短路，最后一个状压 dp 就好了。

比赛的时候把最后的 dp 当成 TSP 问题了，但其实可以走重复的路，不然会漏解。

于是补了个 Floyd 算法，就 AC 了。

```
#include <iostream>
#include <cstdio>
#include <cstdlib>
#include <algorithm>
#include <string>
#include <sstream>
#include <cstring>
#include <cctype>
#include <cmath>
#include <vector>
#include <set>
#include <map>
#include <stack>
#include <queue>
#include <ctime>
#include <cassert>
#define _for(i,a,b) for(int i=(a);i<(b);++i)
#define _rep(i,a,b) for(int i=(a);i<=(b);++i)
#define mem(a,b) memset(a,b,sizeof(a))
using namespace std;
typedef long long LL;
inline int read_int(){
    int t=0;bool sign=false;char c=getchar();
    while(!isdigit(c)){sign|=c=='-';c=getchar();}
    while(isdigit(c)){t=(t<<1)+(t<<3)+(c&15);c=getchar();}
    return sign?-t:t;
}
inline LL read_LL(){
    LL t=0;bool sign=false;char c=getchar();
    while(!isdigit(c)){sign|=c=='-';c=getchar();}
    while(isdigit(c)){t=(t<<1)+(t<<3)+(c&15);c=getchar();}
    return sign?-t:t;
}
```

```
inline char get_char(){
    char c=getchar();
    while(c==' '||c=='\n'||c=='\r')c=getchar();
    return c;
}
inline void write(LL x){
    register char c[21],len=0;
    if(!x)return putchar('0'),void();
    if(x<0)x=-x,putchar('-');
    while(x)c[++len]=x%10,x/=10;
    while(len)putchar(c[len--]+48);
}
inline void space(LL x){write(x),putchar(' ');}
inline void enter(LL x){write(x),putchar('\n');}
const int MAXN=55,dir[][2]={{1,0},{-1,0},{0,1},{0,-1}};
typedef pair<int,int> pii;
int
color[MAXN][MAXN],n,m,tot_c,vis[MAXN][MAXN],dist[MAXN][MAXN],dis[MAXN][MAXN
];
vector<pii> gcc[MAXN];
char a[MAXN][MAXN];
bool in(int x,int y){
    return x>=0&&x<n&&y>=0&&y<m;
}
void bfs(pii s,int c){
    queue<pii> q;
    q.push(s);
    color[s.first][s.second]=c;
    gcc[c].push_back(s);
    while(!q.empty()){
        pii u=q.front();q.pop();
        _for(i,0,4){
            int x=u.first+dir[i][0],y=u.second+dir[i][1];
            if(in(x,y)&&a[x][y]=='X'&&color[x][y]==-1){
                q.push(make_pair(x,y));
                color[x][y]=c;
                gcc[c].push_back(make_pair(x,y));
            }
        }
    }
}
void bfs_2(int c){
    queue<pii> q;
    _for(i,0,gcc[c].size()){
        vis[gcc[c][i].first][gcc[c][i].second]=c;
        dist[gcc[c][i].first][gcc[c][i].second]=0;
        q.push(gcc[c][i]);
    }
    while(!q.empty()){
```

```
        pii u=q.front();q.pop();
        _for(i,0,4){
            int x=u.first+dir[i][0],y=u.second+dir[i][1];
            if(in(x,y)&&a[x][y]!='.'&&vis[x][y]!=c){
                vis[x][y]=c;
                dist[x][y]=dist[u.first][u.second]+1;
                if(color[x][y]!=-1)
                dis[c][color[x][y]]=min(dis[c][color[x][y]],dist[x][y]-1);
                else
                q.push(make_pair(x,y));
            }
        }
    }
}
void bf(){
    _for(i,0,tot_c)
    dis[i][i]=0;
    _for(i,0,tot_c)
    _for(j,0,tot_c)
    _for(k,0,tot_c)
    dis[i][j]=min(dis[i][j],dis[i][k]+dis[j][k]);
}
const int MAXS=1<<15;
int dp[15][MAXS];
int main()
{
    n=read_int(),m=read_int();
    _for(i,0,n)
    gets(a[i]);
    mem(color,-1);mem(vis,-1);
    _for(i,0,n)
    _for(j,0,m){
        if(a[i][j]=='X'&&color[i][j]==-1)
        bfs(make_pair(i,j),tot_c++);
    }
    mem(dis,127/3);
    _for(i,0,tot_c)
    bfs_2(i);
    int MS=(1<<tot_c)-1;
    mem(dp,127/3);
    _for(i,0,tot_c)
    dp[i][MS^(1<<i)]=0;
    for(int s=MS;s>=0;s--){
        _for(i,0,tot_c){
            if(s&(1<<i))
            continue;
            _for(j,0,tot_c){
                if(i!=j)
                dp[i][s]=min(dp[i][s],dp[j][s|(1<<i)]+dis[i][j]);
            }
        }
```

```
    }
    int ans=1e9;
    _for(i,0,tot_c)
    ans=min(ans,dp[i][0]);
    enter(ans);
    return 0;
}
```

B. Cow Lineup

题解

直接取尺法就好了，但比赛的时候写麻烦了，还写了个线段树维护答案，其实不需要。

```
#include <iostream>
#include <cstdio>
#include <cstdlib>
#include <algorithm>
#include <string>
#include <sstream>
#include <cstring>
#include <cctype>
#include <cmath>
#include <vector>
#include <set>
#include <map>
#include <stack>
#include <queue>
#include <ctime>
#include <cassert>
#define _for(i,a,b) for(int i=(a);i<(b);++i)
#define _rep(i,a,b) for(int i=(a);i<=(b);++i)
#define mem(a,b) memset(a,b,sizeof(a))
using namespace std;
typedef long long LL;
inline int read_int(){
    int t=0;bool sign=false;char c=getchar();
    while(!isdigit(c)){sign|=c=='-';c=getchar();}
    while(isdigit(c)){t=(t<<1)+(t<<3)+(c&15);c=getchar();}
    return sign?-t:t;
}
inline LL read_LL(){
    LL t=0;bool sign=false;char c=getchar();
    while(!isdigit(c)){sign|=c=='-';c=getchar();}
    while(isdigit(c)){t=(t<<1)+(t<<3)+(c&15);c=getchar();}
    return sign?-t:t;
```

```
}
inline char get_char(){
    char c=getchar();
    while(c==' '||c=='\n'||c=='\r')c=getchar();
    return c;
}
inline void write(LL x){
    register char c[21],len=0;
    if(!x)return putchar('0'),void();
    if(x<0)x=-x,putchar('-');
    while(x)c[++len]=x%10,x/=10;
    while(len)putchar(c[len--]+48);
}
inline void space(LL x){write(x),putchar(' ');}
inline void enter(LL x){write(x),putchar('\n');}
const int MAXN=1e5+5;
int a[MAXN],b[MAXN],cnt[MAXN];
struct Tree{
    int lef[MAXN<<2],rig[MAXN<<2],w[MAXN<<2];
    int (*merge)(int,int);
    void Push_up(int k){w[k]=merge(w[k<<1],w[k<<1|1]);}
    void build(int k,int L,int R){
        lef[k]=L,rig[k]=R,w[k]=w[0];
        if(L==R)
        return;
        int M=L+R>>1;
        build(k<<1,L,M);
        build(k<<1|1,M+1,R);
    }
    void build(int n,int w_0){
        w[0]=w_0;
        build(1,1,n);
    }
    void update(int k,int p,int v){
        if(lef[k]==rig[k])
        return w[k]=v,void();
        int mid=lef[k]+rig[k]>>1;
        if(p<=mid)
        update(k<<1,p,v);
        else
        update(k<<1|1,p,v);
        Push_up(k);
    }
    void update_2(int k,int p,int v){
        if(lef[k]==rig[k])
        return w[k]+=v,void();
        int mid=lef[k]+rig[k]>>1;
        if(p<=mid)
        update_2(k<<1,p,v);
        else
        update_2(k<<1|1,p,v);
```

```
        Push_up(k);
    }
    int query(int k,int L,int R){
        if(L>R)
        return w[0];
        if(L<=lef[k]&&rig[k]<=R)
        return w[k];
        int mid=lef[k]+rig[k]>>1;
        if(R<=mid)
        return query(k<<1,L,R);
        else if(L>mid)
        return query(k<<1|1,L,R);
        return merge(query(k<<1,L,R),query(k<<1|1,L,R));
    }
}tree;
int Max(int a,int b){
    return a>b?a:b;
}
int main()
{
    int n=read_int(),k=read_int();
    _for(i,0,n)
    a[i]=read_int();
    memcpy(b,a,sizeof(a));
    sort(b,b+n);
    int m=unique(b,b+n)-b;
    _for(i,0,n)
    a[i]=lower_bound(b,b+m,a[i])-b;
    int lef=0,rig=0,tot=0,ans=0;
    tree.merge=Max;
    tree.build(m,0);
    while(lef<n){
        while(rig<n&&(tot<=(k+1))){
            cnt[a[rig]]++;
            if(cnt[a[rig]]==1)
            tot++;
            tree.update_2(1,a[rig]+1,1);
            rig++;
        }
        if(tot>(k+1)){
            rig--;
            cnt[a[rig]]--;
            tot--;
            tree.update_2(1,a[rig]+1,-1);
        }
        ans=max(ans,tree.query(1,1,m));
        cnt[a[lef]]--;
        if(cnt[a[lef]]==0)
        tot--;
```

```
        tree.update_2(1,a[lef]+1,-1);
        lef++;
    }
    enter(ans);
    return 0;
}
```

```
#include<cstdio>
#include<map>
int n , k ;

std :: map < int  , int > cnt ;

const int N = 1e5 + 10 ;
int a[N] ;
inline int max(int x , int y) {
    return x > y ? x : y ;
}
signed main() {
    scanf("%d %d" , & n , & k) ;
    for(register int i = 1 ; i <= n ; i ++) scanf("%d" , & a[i]) ;
    int l = 1 , r  = 0 ;
    int kind = 0 ;
    int ans = 0 ;
    while( r <= n ) {
        if(++ cnt [ a[++ r]] == 1) kind ++ ;
        while( kind == k + 2 ) {
            if(-- cnt[ a[l ++]] == 0) kind -- ;
        }
        ans = max (ans , cnt[a[r]]) ;
    }
    printf("%d\n" , ans) ;
    return 0 ;
}
```

I. Running Away From the Barn

题解

这题思路还挺多的。有跑树上差分 $+$ 倍增的；还有按 depth 降序询问，然后权值线段树/树状数组维护答案的；还有左偏树的。

比赛时没想到调整询问顺序，直接写了个主席树 $+$ 权值线段树，感觉又写麻烦了。

```
#include <iostream>
#include <cstdio>
#include <cstdlib>
```

```
#include <algorithm>
#include <string>
#include <sstream>
#include <cstring>
#include <cctype>
#include <cmath>
#include <vector>
#include <set>
#include <map>
#include <stack>
#include <queue>
#include <ctime>
#include <cassert>
#define _for(i,a,b) for(int i=(a);i<(b);++i)
#define _rep(i,a,b) for(int i=(a);i<=(b);++i)
#define mem(a,b) memset(a,b,sizeof(a))
using namespace std;
typedef long long LL;
inline int read_int(){
    int t=0;bool sign=false;char c=getchar();
    while(!isdigit(c)){sign|=c=='-';c=getchar();}
    while(isdigit(c)){t=(t<<1)+(t<<3)+(c&15);c=getchar();}
    return sign?-t:t;
}
inline LL read_LL(){
    LL t=0;bool sign=false;char c=getchar();
    while(!isdigit(c)){sign|=c=='-';c=getchar();}
    while(isdigit(c)){t=(t<<1)+(t<<3)+(c&15);c=getchar();}
    return sign?-t:t;
}
inline char get_char(){
    char c=getchar();
    while(c==' '||c=='\n'||c=='\r')c=getchar();
    return c;
}
inline void write(LL x){
    register char c[21],len=0;
    if(!x)return putchar('0'),void();
    if(x<0)x=-x,putchar('-');
    while(x)c[++len]=x%10,x/=10;
    while(len)putchar(c[len--]+48);
}
inline void space(LL x){write(x),putchar(' ');}
inline void enter(LL x){write(x),putchar('\n');}
const int MAXN=2e5+5;
struct Node{
    int lef,rig,val;
};
Node node[MAXN*40];
```

```
int cnt,root[MAXN];
int nodecopy(int k){
    node[++cnt]=node[k];
    return cnt;
}
void build(int &k,int lef,int rig){
    k=++cnt;
    int mid=lef+rig>>1;
    if(lef==rig)
    return;
    build(node[k].lef,lef,mid);
    build(node[k].rig,mid+1,rig);
}
void update(int &k,int p,int lef,int rig,int pos){
    k=nodecopy(p);
    node[k].val++;
    if(lef==rig)
    return;
    int mid=lef+rig>>1;
    if(mid<pos)
    update(node[k].rig,node[p].rig,mid+1,rig,pos);
    else
    update(node[k].lef,node[p].lef,lef,mid,pos);
}
int query(int k1,int k2,int lef,int rig,int L,int R){
    if(L<=lef&&rig<=R)
    return node[k1].val-node[k2].val;
    int mid=lef+rig>>1;
    if(mid>=R)
    return query(node[k1].lef,node[k2].lef,lef,mid,L,R);
    else if(mid<L)
    return query(node[k1].rig,node[k2].rig,mid+1,rig,L,R);
    else
    return
query(node[k1].lef,node[k2].lef,lef,mid,L,R)+query(node[k1].rig,node[k2].ri
g,mid+1,rig,L,R);
}
int head[MAXN],edge_cnt;
struct Edge{
    int to,next;
    LL w;
}edge[MAXN];
void Insert(int u,int v,LL w){
    edge[++edge_cnt].next=head[u];
    edge[edge_cnt].to=v;
    edge[edge_cnt].w=w;
    head[u]=edge_cnt;
}
LL dep[MAXN],dw[MAXN],b[MAXN];
int dfs_t,lf[MAXN],rf[MAXN];
void dfs(int u,LL d){
```

```
    lf[u]=++dfs_t;dep[u]=d;dw[dfs_t]=d;
    for(int i=head[u];i;i=edge[i].next){
        int v=edge[i].to;
        dfs(v,d+edge[i].w);
    }
    rf[u]=dfs_t;
}
int main()
{
    int n=read_int(),u;
    LL L=read_LL(),w;
    _rep(i,2,n){
        u=read_int(),w=read_LL();
        Insert(u,i,w);
    }
    dfs(1,0);
    memcpy(b,dep,sizeof(dep));
    sort(b+1,b+n+1);
    int m=unique(b+1,b+n+1)-b;
    build(root[0],1,m);
    _rep(i,1,n)
    update(root[i],root[i-1],1,m,lower_bound(b+1,b+m,dw[i])-b);
    _rep(i,1,n)
enter(query(root[rf[i]],root[lf[i]-1],1,m,1,upper_bound(b+1,b+m,dep[i]+L)-
b-1));
    return 0;
}
```

```
#include<bits/stdc++.h>
using namespace std;
const int N=200005;
int n,id[N],r[N],now,ans[N];
pair<long long,int> a[N];
struct BIT
{
    int tree[N];
    void modify(int x,int val)
    {
        for(;x<=n;x+=x&-x)
            tree[x]+=val;
    }
    int query(int x)
    {
        int ans=0;
        for(;x;x-=x&-x)
            ans+=tree[x];
        return ans;
    }
```

```
}T;
//树状数组模板
vector<pair<int,long long>> mat[N];
void dfs(int k)
{
    id[k]=++now;
  //dfs序
    for(auto e:mat[k])
    {
        a[e.first]=make_pair(a[k].first+e.second,e.first);
        dfs(e.first);
    }
    r[k]=now;
  //结束时间戳
}
int main()
{
    long long l;
    cin>>n>>l;
    for(int i=2;i<=n;i++)
    {
        int p;
        long long w;
        cin>>p>>w;
        mat[p].push_back(make_pair(i,w));
    }
    a[1]=make_pair(0ll,1);
    dfs(1);
    sort(a+1,a+n+1);
    int j=n;
    for(int i=n;i;i--)
    {
        for(;a[j].first-a[i].first>l;j--)
            T.modify(id[a[j].second],-1);
      //删除超过距离超过L的点
        T.modify(id[a[i].second],1);
      //插入当前点
        ans[a[i].second]=T.query(r[a[i].second])-
T.query(id[a[i].second]-1);
      //统计子树答案
    }
    for(int i=1;i<=n;i++)
        cout<<ans[i]<<endl;
    return 0;
}
```