普及模拟赛题解

变换 (change)

不难发现，转换的过程只和 a 和 b 的二进制位有关，且不同二进制位之间无关。我们可以将 a 和 b 转化为二进制表示，每一位分别判断，如果这位不同，答案 $+1$。

更快的方法

可以发现对于每一位，如果 a,b 的这位相同，异或值为 0，如果不同，异或值为 1。所以答案为 a 异或 b 的二进制 1 的个数。

```
#include<bits/stdc++.h>
using namespace std;

int main(){
    freopen("change.in", "r", stdin);
    freopen("change.out", "w", stdout);

    long long a, b;

    scanf("%lld%lld", &a, &b);
    printf("%d\n", __builtin_popcountll(a ^ b));

    return 0;
}
```

`__builtin_popcount()` 函数可以在 $O(1)$ 的复杂的计算一个数二进制 1 的个数而 `__builtin_popcountll()` 是其的 `long long` 版本。

打地鼠 (mouse)

简单贪心，每次找到位置最靠前的未被消灭的地鼠 i，对 $i\sim i+k-1$ 的地鼠进行打击。

```
#include<bits/stdc++.h>
using namespace std;
const int N=2e5+10;
int n,k,ans;
char s[N];
int main(){
    freopen("mouse.in","r",stdin);
    freopen("mouse.out","w",stdout);
    scanf("%d%d%s",&n,&k,s+1);ans=0;
    for(int i=1;i<=n;++i)
        if(s[i]=='1') ++ans,i+=k-1;
    printf("%d\n",ans);
    return 0;
}
```

删除 (delete)

我们考虑删图的逆过程，每次加入一个点，就将和它相连的的已经加入的点的连通块并成一个新的连通块。连通块及其的权值可以用并查集维护。

```cpp
#include<bits/stdc++.h>
using namespace std;

const int N = 1e5 + 5;
typedef long long LL;

int v[N], id[N];
LL sz[N];
int fa[N];
bool st[N];
vector<int> e[N];
LL ans[N];

int fi(int x){
    if(fa[x] == x)return x;
    fa[x] = fi(fa[x]);
    return fa[x];
}

int main(){
    freopen("delete.in", "r", stdin);
    freopen("delete.out", "w", stdout);

    int n, m;
    scanf("%d%d", &n, &m);

    for(int i = 1; i <= n; i ++){
        scanf("%d", &v[i]);
        sz[i] = v[i];
        fa[i] = i;
    }

    for(int i = 1; i <= m; i ++){
        int a, b;
        scanf("%d%d", &a, &b);
        e[a].push_back(b);
        e[b].push_back(a);
    }

    for(int i = 1; i <= n; i ++){
        scanf("%d", &id[i]);
    }

    LL res = 0;
    for(int i = n; i >= 1; i --){
        int x = id[i];

        for(int u : e[x]){
            if(!st[u])continue;

            if(fi(u) != fi(x)){
                sz[fi(x)] += sz[fi(u)];
                fa[fi(u)] = fi(x);
            }
```

```
        }

        st[x] = 1;
        res = max(res, sz[fi(x)]);
        ans[i - 1] = res;
    }

    for(int i = 1; i <= n; i ++)printf("%lld ", ans[i]);
    puts("");

    return 0;
}
```

刮彩票 (lottery)

简单模拟发现，操作相当于可以把一个形如 `AAA.....AB` 的转化为 `BCC.....CC`，或将形如 `BAA.....AA` 的转化为 `CC.....CCB`。

等价于将一个 `A` 连续段与一个与其相邻的 `B` 删去。

故我们可以进行 dp，对于每个初始是 `B` 的位置 i，$f_{i,1/0}$ 为在 i 的前缀字符串中，i 位置有/没有被前面的 `A` 连续段占用。

$$f_{i,0} = max(f_{la,0} + i - la - 1, f_{la,1})$$

$$f_{i,1} = max(f_{la,0} + i - la - 1, f_{la,1} + i - la - 1)$$

其中 la 为上一个 `B` 的位置。

最终答案为 $max(f_{las,0} + n - las, f_{las,1})$，$las$ 为最后一个 `B` 的位置。

```
#include<bits/stdc++.h>
using namespace std;
const int N=2e5+10;
int n,f[N][2];
char s[N];
int main(){
    freopen("lottery.in","r",stdin);
    freopen("lottery.out","w",stdout);
    scanf("%d%s",&n,s+1);
    int la=0;
    f[0][0]=-n;
    for(int i=1;i<=n;++i)
        if(s[i]=='B'){
            f[i][0]=max(f[la][0]+i-la-1,f[la][1]);
            f[i][1]=max(f[la][0],f[la][1])+i-la-1;
            la=i;
        }
    printf("%d\n",max(f[la][0]+n-la,f[la][1]));
    return 0;
}
```