第一题：数色块

```
#include<bits/stdc++.h>
using namespace std;
int main(){
    int n;
    cin >> n;
    string s;
    cin >> s;
    s = " " + s;
    char last = s[1];
    int ans = 1;
    for(int i = 2; i <= n; i ++ ){
        if(s[i] != last){
            ans ++;
            last = s[i];
        }
    }
    cout << ans;
    return 0;
}
```

第二题：扔硬币游戏 I

```
#pragma GCC optimize(3,"Ofast","inline")
#include <bits/stdc++.h>

#define ios_close std::ios::sync_with_stdio(false), std::cin.tie(nullptr),
std::cout.tie(nullptr)

using ll = long long;
using ull = unsigned long long;
using i128 = __int128;
#define Pi  acos(-1.0);
#define PINF 0x3f3f3f3f
#define NINF -0x3f3f3f3f

const int MOD = 1e9 + 7;

void solve(){
    int n;
    std::cin >> n;
    //表示 第 i 个数字是 k的方案数
    std::vector<std::vector<int>> dp(1000010, std::vector<int>(2, 0));
    dp[1][0] = 1;
    dp[2][0] = 2;
    dp[1][1] = 1;
    dp[2][1] = 2;
    for(int i = 3; i <= n; i ++ ){
        dp[i][1] = (dp[i - 1][0] + dp[i - 2][0]) % MOD;
        dp[i][0] = (dp[i - 1][1] + dp[i - 2][1]) % MOD;
```

```
    }
    std::cout << (dp[n][1] + dp[n][0]) % MOD;
}

int main(){
#if 0
    ios_close;
#endif

#if 0
    freopen(".in", "r", stdin);
    freopen(".out", "w", stdout);
#endif
    int T = 1;
#if 0
    std::cin >> T;
#endif
    while(T -- ){
        solve();
    }
    return 0;
}
```

第三题：扔硬币游戏 II

```
#pragma GCC optimize(3,"Ofast","inline")
#include <bits/stdc++.h>

#define ios_close std::ios::sync_with_stdio(false), std::cin.tie(nullptr),
std::cout.tie(nullptr)

using ll = long long;
using ull = unsigned long long;
using i128 = __int128;
#define Pi  acos(-1.0);
#define PINF 0x3f3f3f3f
#define NINF -0x3f3f3f3f

void solve(){
    int n, k;
    scanf("%d %d", &n, &k);
    std::vector<int> a(1000010);
    std::vector<int> pre(1000010, 0);
    for(int i = 1; i <= n; i ++ ){
        scanf("%1d", &a[i]);
        pre[i] = pre[i - 1] + a[i];
    }
    int min = INT_MAX;
    for(int i = 1; i + k - 1 <= n; i ++ ){
        int l = i, r = i + k - 1;
        min = std::min(min, pre[r] - pre[l - 1]);
    }
    std::cout << min;
}
```

```
int main(){
#if 0
    ios_close;
#endif

#if 0
    freopen(".in", "r", stdin);
    freopen(".out", "w", stdout);
#endif
    int T = 1;
#if 0
    std::cin >> T;
#endif
    while(T -- ){
        solve();
    }
    return 0;
}
```

第四题：涂路径

```
#include<bits/stdc++.h>

std::vector<std::vector<char>> map(1010, std::vector<char>(1010));
std::vector<int> dx{0, -1, 0, 1};
std::vector<int> dy{-1, 0, 1, 0};
std::vector<std::vector<bool>> vis(1010, std::vector<bool>(1010, false));
struct Node {
    int x, y, step;
    bool operator< (const Node a) const {
        return step > a.step;
    }
};
int n, m;
int sx, sy, ex, ey;

int bfs(){
    std::priority_queue<Node> q;
    q.push({sx, sy, 0});
    while(!q.empty()){
        auto t = q.top();
        q.pop();
        if(t.x < 1 || t.x > n || t.y < 1 || t.y > m || vis[t.x][t.y] || map[t.x]
[t.y] == '#'){
            continue;
        }
        //std::cout << t.x << " " << t.y << " " << t.step << '\n';
        vis[t.x][t.y] = true;
        if(map[t.x][t.y] == 'E'){
            return t.step;
        }
        for(int i = 0; i < 4; i ++ ){
            int x = t.x + dx[i];
            int y = t.y + dy[i];
            if(map[x][y] == '@'){
```

```
                q.push({x, y, t.step + 3});
            }
            if(map[x][y] == 'E' || map[x][y] == '.'){
                q.push({x, y, t.step + 1});
            }
        }
    }
    return -1;
}

int main(){
    std::cin >> n >> m;
    for(int i = 1; i <= n; i ++ ){
        for(int j = 1; j <= m; j ++ ){
            std::cin >> map[i][j];
            if(map[i][j] == 'S'){
                sx = i;
                sy = j;
            }
            if(map[i][j] == 'E'){
                ex = i;
                ey = j;
            }
        }
    }
    int ans = bfs();
    if(ans == -1){
        puts("The End!!!");
    } else {
        std::cout << ans;
    }
    return 0;
}
```

第五题：团伙

```
#include <bits/stdc++.h>

using namespace std;

int fa[10010];

void init(int n){
    for(int i = 1; i <= 2 * n; i ++ ){
        fa[i] = i;
    }
}

int find(int x){
    return (fa[x] == x) ? x : (fa[x] = find(fa[x]));
}

void merge(int x, int y){
    int fx = find(x);
    int fy = find(y);
    if(fx != fy){
        fa[fy] = fx;
```

```
    }
}

int main(){
    int n, m;
    cin >> n >> m;
    int ans = 0;
    init(n);
    while(m -- ){
        int a, x, y;
        cin >> a >> x >> y;
        if(a){
            merge(x, n + y);
            merge(y, n + x);
        } else {
            merge(x, y);
        }
    }
    for(int i = 1; i <= n; i ++ ){
        if(fa[i] == i){
            ans ++;
        }
    }
    cout << ans;
    return 0;
}
```