二分图匹配&KM 题解

A - Girls and Boys (HDU-1068)

题意

N个人每个人有几个浪漫关系，求一个最大的集合，使集合中没存在两两有浪漫关系

思路

首先求出最大匹配 (结果要除2)，这道题其实是求最大独立集，最大独立集 = N - 最大匹配，因为至少要把最大匹配的所有边都去掉才能保证集合内的点不存在两两相关

```
#include <bits/stdc++.h>
const int maxn = 1e3 + 5;
const int inf = 0x3f3f3f3f;
using namespace std;
int girl[maxn], used[maxn];
vector<int> g[maxn];
int dfs(int u) {
    for (int i = 0; i < (int)g[u].size(); ++i) {
        int v = g[u][i];
        if (used[v]) continue;
        used[v] = 1;
        if (girl[v] == 0 || dfs(girl[v])) {
            girl[v] = u;
            return true;
        }
    }
    return false;
}
int main() {
```

```
    ios::sync_with_stdio(false);
    cin.tie(0), cout.tie(0);
    int n;
    while (scanf("%d", &n) != EOF) {
        for (int i = 0; i <= n; ++i) {
            g[i].clear();
        }
        memset(girl, 0, sizeof(girl));
        int u, v, m;
        for (int j = 0; j < n; ++j) {
            scanf("%d: (%d)", &u, &m);
            for (int i = 0; i < m; ++i) {
                scanf("%d", &v);
                g[u].push_back(v);
            }
        }
        int ans = 0;
        for (int i = 0; i < n; ++i) {
            memset(used, 0, sizeof(used));
            int t = dfs(i);
            ans += t;
        }
        ans = n - ans / 2;
        printf("%d\n", ans);
    }
    return 0;
}
```

B - Machine Schedule (HDU-1150)

题意

一共有AB两台机器，每个机器有多个工作模式。K个任务每个任务可以在A的某个模式、B的某个模式中工作，每次切换模式需要花费1。求最少的花费。

思路

最小点覆盖 = 最大匹配

A和B的某个模式对应一个任务，将AB建边。然后求最大匹配。

```
#include <bits/stdc++.h>
const int maxn = 1e3 + 5;
const int inf = 0x3f3f3f3f;
using namespace std;
int used[maxn], girl[maxn];
vector<int> g[maxn];
void init() {
    for (int i = 0; i < maxn; ++i) g[i].clear();
    memset(girl, 0, sizeof(girl));
}
int dfs(int u) {
    for (int i = 0; i < (int)g[u].size(); ++i) {
        int v = g[u][i];
        if (used[v]) continue;
        used[v] = 1;
```

```
        if (girl[v] == 0 || dfs(girl[v])) {
            girl[v] = u;
            return 1;
        }
    }
    return 0;
}
int main() {
    ios::sync_with_stdio(false);
    cin.tie(0), cout.tie(0);
    int n, m, k;
    while (cin >> n >> m >> k , n) {
        init();
        for (int i = 0; i < k; ++i) {
            int a, b, c;
            cin >> a >> b >> c;
            if (!b || !c) continue;
            g[b].push_back(c);
        }
        int ans = 0;
        for (int i = 0; i < n; ++i) {
            memset(used, 0, sizeof(used));
            ans += dfs(i);
        }
        cout << ans << endl;
    }

    return 0;
}
```

C - Air Raid (HDU - 1151)

题意

一些伞兵降落之后可以选择访问它所能到达的交叉路口，求最小的伞兵数量可以访问每个交叉路口，同时每个伞兵的经过的加岔路口不能重合(题目好像没有明显的说明)

思路

最小路径覆盖 = 点数 - 最大匹配

```
#include <bits/stdc++.h>
const int maxn = 1e3 + 5;
const int inf = 0x3f3f3f3f;
using namespace std;
int used[maxn], girl[maxn];
vector<int> g[maxn];
void init() {
    for (int i = 0; i < maxn; ++i) g[i].clear();
    memset(girl, 0, sizeof(girl));
}
int dfs(int u) {
    for (int i = 0; i < (int)g[u].size(); ++i) {
        int v = g[u][i];
        if (used[v]) continue;
```

```
        used[v] = 1;
        if (girl[v] == 0 || dfs(girl[v])) {
            girl[v] = u;
            return 1;
        }
    }
    return 0;
}
int main() {
    ios::sync_with_stdio(false);
    cin.tie(0), cout.tie(0);
    int T;
    cin >> T;
    while (T--) {
        init();
        int n, m;
        cin >> n >> m;
        for (int i = 0; i < m; ++i) {
            int u, v;
            cin >> u >> v;
            g[u].push_back(v);
        }
        int ans = 0;
        for (int i = 1; i <= n; ++i) {
            memset(used, 0, sizeof(used));
            ans += dfs(i);
        }
        ans = n - ans;
        cout << ans << endl;
    }
    return 0;
}
```

D - Ollivanders: Makers of Fine Wands since 382 BC.(HDU - 1179)

题意

N个巫师，M个魔杖，求最大匹配

思路

板子题

```
#include <bits/stdc++.h>
const int maxn = 1e3 + 5;
const int inf = 0x3f3f3f3f;
using namespace std;
int used[maxn], girl[maxn];
vector<int> g[maxn];
void init() {
    for (int i = 0; i < maxn; ++i) g[i].clear();
    memset(girl, 0, sizeof(girl));
}
int dfs(int u) {
    for (int i = 0; i < (int)g[u].size(); ++i) {
        int v = g[u][i];
```

```
        if (used[v]) continue;
        used[v] = 1;
        if (girl[v] == 0 || dfs(girl[v])) {
            girl[v] = u;
            return 1;
        }
    }
    return 0;
}
int main() {
    ios::sync_with_stdio(false);
    cin.tie(0), cout.tie(0);
    int n, m;
    while (cin >> n >> m) {
        init();
        for (int i = 0; i < m; ++i) {
            int a, b;
            cin >> a;
            while (a--) {
                cin >> b;
                g[i+1].push_back(b);
            }
        }
        int ans = 0;
        for (int i = 1; i <= m; ++i) {
            memset(used, 0, sizeof(used));
            ans += dfs(i);
        }
        cout << ans << endl;
    }
    return 0;
}
```

E - 棋盘游戏 (HDU-1281)

题意

在一个N*M的矩阵上，规定一些位置可以放车，求最多放几个车相互不冲突并求那些车是important

思路

每行每列只能放一个车，最多放车的数量就是行和列的最大匹配。

枚举删除每个点，如果最大匹配减少则为important

```
#include <bits/stdc++.h>
const int maxn = 1e3 + 5;
const int inf = 0x3f3f3f3f;
using namespace std;
int used[maxn], girl[maxn];
vector<int> g[maxn];
int R, C;
void init() {
    R = C = -1;
    for (int i = 0; i < maxn; ++i) g[i].clear();
    memset(girl, 0, sizeof(girl));
}
```

```
int dfs(int u) {
    for (int i = 0; i < (int)g[u].size(); ++i) {
        int v = g[u][i];
        if (u == R && v == C) continue;
        if (used[v]) continue;
        used[v] = 1;
        if (girl[v] == 0 || dfs(girl[v])) {
            girl[v] = u;
            return 1;
        }
    }
    return 0;
}
int main() {
    ios::sync_with_stdio(false);
    cin.tie(0), cout.tie(0);
    int n, m, k, Case = 0;
    while (cin >> n >> m >> k) {
        init();
        vector<pair<int,int>> vec;
        for (int i = 0; i < k; ++i) {
            int r, c;
            cin >> r >> c;
            vec.push_back(make_pair(r, c));
            g[r].push_back(c);
        }
        int all = 0, master = 0;
        for (int i = 1; i <= n; ++i) {
            memset(used, 0, sizeof(used));
            all += dfs(i);
        }
        for (auto it : vec) {
            R = it.first;
            C = it.second;
            int cnt = 0;
            memset(girl, 0, sizeof(girl));
            for (int i = 1; i <= n; ++i) {
                memset(used, 0, sizeof(used));
                cnt += dfs(i);
            }
            if (cnt < all) master++;
        }
        cout << "Board " << ++Case << " have " << master << " important blanks
for " << all << " chessmen.\n";
    }
    return 0;
}
```

F - 50 years, 50 colors (HDU-1498)

题意

n*n的矩阵每个位置有一种颜色的气球，你可以每次选择一行(列)并把这行(列)相同的气球删去。问K次操作那些气球不能被删除？

思路

最小点覆盖 = 最大匹配

枚举每一种颜色，对于一种颜色的气球，根据行列坐标建图转化为最小点覆盖问题，如果点数大于K则不能完全删除

```
#include <bits/stdc++.h>
#define mem(a, b) memset(a, b, sizeof(a))
const int maxn = 1e2 + 5;
using namespace std;
int n, m;
int used[maxn], girl[maxn];
int g[maxn][maxn];
int dfs(int u, int c) {
    for (int i = 1; i <= n; ++i) {
        int v = g[u][i];
        if (v != c || used[i]) continue;
        used[i] = 1;
        if (girl[i] == 0 || dfs(girl[i], c)) {
            girl[i] = u;
            return 1;
        }
    }
    return 0;
}
int main () {
    ios::sync_with_stdio(false);
    cin.tie(0), cout.tie(0);
    while (cin >> n >> m, n&&m) {
        set<int> st;
        for (int i = 1; i <= n; ++i) {
            for (int j = 1; j <= n; ++j) {
                cin >> g[i][j];
                st.insert(g[i][j]);
            }
        }
        vector<int> ans;
        set<int>::iterator it = st.begin();
        while (it != st.end()){
            int cnt = 0;
            mem(girl, 0);
            for (int j = 1; j <= n; ++j) {
                mem(used, 0);
                cnt += dfs(j, *it);
            }
            if (cnt > m) ans.push_back(*it);
            ++it;
        }
        sort(ans.begin(), ans.end());
        if (ans.empty()) {
            cout << -1;
        }else {
            for (int i = 0; i < (int)ans.size(); ++i) {
                if (i) cout << " ";
                cout << ans[i];
            }
        }
        cout << endl;
```

```
    }
    return 0;
}
```

G - 奔小康赚大钱 (HDU - 2255)

思路

板子题

```
#include <bits/stdc++.h>
const int maxn = 5e2 + 5;
const int inf = 0x3f3f3f3f;
using namespace std;
int usex[maxn], usey[maxn], topx[maxn], topy[maxn], slack[maxn];
int girl[maxn];
int g[maxn][maxn];
int n;
int dfs(int x) {
    usex[x] = 1;
    for (int i = 1; i <= n; ++i) {
        if (usey[i]) continue;
        int tmp = topx[x] + topy[i] - g[x][i];
        if (tmp != 0) {
            slack[i] = min(tmp, slack[i]);
        }else {
            usey[i] = 1;
            if (girl[i] == -1 || dfs(girl[i])) {
                girl[i] = x;
                return 1;
            }
        }
    }
    return 0;
}
int km() {
    memset(girl, -1, sizeof(girl));
    memset(topx, 0, sizeof(topx));
    memset(topy, 0, sizeof(topy));
    for (int i = 1; i <= n; ++i) {
        for (int j = 1; j <= n; ++j) {
            topx[i] = max(topx[i], g[i][j]);
        }
    }
    for (int i = 1; i <= n; ++i) {
        memset(slack, inf, sizeof(slack));
        while (1) {
            memset(usex, 0, sizeof(usex));
            memset(usey, 0, sizeof(usey));
            if (dfs(i)) break;
            int tmp = inf;
            for (int j = 1; j <= n; ++j) {
                if (usey[j]) continue;
                tmp = min(tmp, slack[j]);
            }
            if (tmp == inf) return -1;
            for (int j = 1; j <= n; ++j) {
```

```
                if (usex[j]) topx[j] -= tmp;
            }
            for (int j = 1; j <= n; ++j) {
                if (usey[j]) topy[j] += tmp;
                else slack[j] -= tmp;
            }
        }
    }
    int ans = 0;
    for (int i = 1; i <= n; ++i) {
        if (girl[i] != -1) ans += g[girl[i]][i];
    }
    return ans;
}
int main () {
    ios::sync_with_stdio(false);
    cin.tie(0), cout.tie(0);
    while (cin >> n) {
        for (int i = 1; i <= n; ++i) {
            for (int j = 1; j <= n; ++j) {
                cin >> g[i][j];
            }
        }
        cout << km() << endl;
    }
    return 0;
}
```

H - Interesting Housing Problem (HDU - 2426)

题意

给N个学生分配宿舍，学生对一些宿舍进行了评分，作为校方你想让每个学生都有房间住并且总体的满意度最高，不能将学生分到他好感度为负的宿舍

思路

KM模板题，N，M的大小不同最后需要统计匹配的人数，如果最大匹配不等于学生人数就无解

```
#include <bits/stdc++.h>
const int maxn = 5e2 + 5;
const int inf = 0x3f3f3f3f;
using namespace std;
int usex[maxn], usey[maxn], topx[maxn], topy[maxn], slack[maxn];
int girl[maxn];
int g[maxn][maxn];
int n, m, e;
int dfs(int x) {
    usex[x] = 1;
    for (int i = 0; i < m; ++i) {
        if (usey[i]) continue;
        int tmp = topx[x] + topy[i] - g[x][i];
        if (tmp != 0) {
            slack[i] = min(tmp, slack[i]);
        }else {
            usey[i] = 1;
            if (girl[i] == -1 || dfs(girl[i])) {
```

```
                girl[i] = x;
                return 1;
            }
        }
    }
    return 0;
}
int km() {
    memset(girl, -1, sizeof(girl));
    memset(topx, 0, sizeof(topx));
    memset(topy, 0, sizeof(topy));
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < m; ++j) {
            topx[i] = max(topx[i], g[i][j]);
        }
    }
    for (int i = 0; i < n; ++i) {
        memset(slack, inf, sizeof(slack));
        while (1) {
            memset(usex, 0, sizeof(usex));
            memset(usey, 0, sizeof(usey));
            if (dfs(i)) break;
            int tmp = inf;
            for (int j = 0; j < m; ++j) {
                if (usey[j]) continue;
                tmp = min(tmp, slack[j]);
            }
            if (tmp == inf) return -1;
            for (int j = 0; j < n; ++j) {
                if (usex[j]) topx[j] -= tmp;
            }
            for (int j = 0; j < m; ++j) {
                if (usey[j]) topy[j] += tmp;
                else slack[j] -= tmp;
            }
        }
    }
    int ans = 0;
    for (int i = 0; i < m; ++i) {
        if (girl[i] != -1) ans += g[girl[i]][i];
    }
    return ans;
}
int main () {
    ios::sync_with_stdio(false);
    cin.tie(0), cout.tie(0);
    int Case = 0;
    while (cin >> n >> m >> e) {
        cout << "Case " << ++Case << ": ";
        fill(g[0], g[0]+maxn*maxn, -inf);
        int u, v, c;
        for (int i = 0; i < e; ++i) {
            cin >> u >> v >> c;
            if (c < 0) continue;
            g[u][v] = c;
        }
        cout << km()<< endl;
    }
```

```
    return 0;
}
```

I - One fight one (HDU - 2813)

题意

吕布的士兵和曹操的对打，你要选取合理的对阵使吕布军队受伤最小

存负边跑KM

思路

```
#include <bits/stdc++.h>
const int maxn = 2e2 + 5;
const int inf = 0x3f3f3f3f;
using namespace std;
int n, m, k;
int usex[maxn], usey[maxn], topx[maxn], topy[maxn], slack[maxn];
int girl[maxn], g[maxn][maxn];
int dfs(int x) {
    usex[x] = 1;
    for (int i = 1; i <= m; ++i) {
        if (usey[i]) continue;
        int tmp = topx[x] + topy[i] - g[x][i];
        if (tmp != 0) {
            slack[i] = min(tmp, slack[i]);
        }else {
            usey[i] = 1;
            if (girl[i] == -1 || dfs(girl[i])) {
                girl[i] = x;
                return 1;
            }
        }
    }
    return 0;
}
int km() {
    memset(girl, -1, sizeof(girl));
    memset(topy, 0, sizeof(topy));
    for (int i = 0; i <= n; ++i) topx[i] = -inf;
    for (int i = 1; i <= n; ++i) {
        for (int j = 1; j <= m; ++j) {
            topx[i] = max(topx[i], g[i][j]);
        }
    }
    for (int i = 1; i <= n; ++i) {
        memset(slack, inf, sizeof(slack));
        while (1) {
            memset(usex, 0, sizeof(usex));
            memset(usey, 0, sizeof(usey));
            if (dfs(i)) break;
            int tmp = inf;
            for (int j = 1; j <= m; ++j) {
                if (usey[j]) continue;
                tmp = min(tmp, slack[j]);
            }
```

```
            if (tmp == inf) return -1;
            for (int j = 1; j <= n; ++j) {
                if (usex[j]) topx[j] -= tmp;
            }
            for (int j = 1; j <= m; ++j) {
                if (usey[j]) topy[j] += tmp;
                else slack[j] -= tmp;
            }
        }
    }
    int ans = 0;
    for (int i = 1; i <= m; ++i) {
        if (girl[i] != -1) ans += g[girl[i]][i];
    }
    return -ans;
}
int main () {
    ios::sync_with_stdio(false);
    cin.tie(0), cout.tie(0);
    while(cin >> n >> m >> k) {
        fill(g[0], g[0]+maxn*maxn, -inf);
        map<string, int> Lv, Cao;
        int L = 1, R = 1;
        for (int i = 0; i < k; ++i) {
            string s1, s2;
            int t;
            cin >> s1 >> s2 >> t;
            if (Lv[s1] == 0)  Lv[s1] = L++;
            if (Cao[s2] == 0) Cao[s2] = R++;
            g[Lv[s1]][Cao[s2]] = -t;
        }
        cout << km() << endl;
    }
    return 0;
}
```