widsnoy's template

1. 数论

1.1. 取模还原分数

1.2. 解不定方程

1.3. 中国剩余定理

1.4. 卢卡斯定理

1.5. exBSGS

1.6. 二次剩余

1.7. Miller-Rabin

1.8. Pollard-rho

1.9. 莫比乌斯反演 (两种形式)

1.10. 除法分块 (上下取整)

1.11. Min25 筛

1.12. 区间筛

1.13. 数论卷积

2. 动态规划

2.1. 缺 1 背包

3. 图论

3.1. 找环

```cpp
const int N = 5e5 + 5;
int n, m, col[N], pre[N], pre_edg[N];
vector<pii> G[N];
vector<vector<int>> resp, rese;
//point
void get_cyc(int u, int v) {
    if (!resp.empty()) return;
    vector<int> cyc;
    cyc.push_back(v);
    while (true) {
        v = pre[v];
        if (v == 0) break;
        cyc.push_back(v);
        if (v == u) break;
    }
    reverse(cyc.begin(), cyc.end());
    resp.push_back(cyc);
}
// edge
void get_cyc(int u, int v, int id) {
    if (!rese.empty()) return;
    vector<int> cyc;
    cyc.push_back(id);
    while (true) {
        if (pre[v] == 0) break;
        cyc.push_back(pre_edg[v]);
        v = pre[v];
        if (v == u) break;
    }
    reverse(cyc.begin(), cyc.end());
    rese.push_back(cyc);
}
void dfs(int u, int edg) {
    col[u] = 1;
    for (auto [v, id] : G[u]) if (id != edg) {
        if (col[v] == 1) {
            get_cyc(v, u);
            get_cyc(v, u, id);
        } else if (col[v] == 0) {
            pre[v] = u;
            pre_edg[v] = id;
            dfs(v, id);
        }
    }
    col[u] = 2;
}
void MAIN() {
    cin >> n >> m;
    for (int i = 1; i <= m; i++) {
        int u, v; cin >> u >> v;
        // G[u].push_back({v, i});
        // G[v].push_back({u, i});
    }
    for (int i = 1; i <= n; i++) if (!col[i])
dfs(i, -1);
}
```

3.2. SPFA 乱搞

```cpp
mt19937_64
rng(chrono::steady_clock::now().time_since_epoch().count());

const int mod = 998244353;
const int N = 5e5 + 5;
const ll inf = 1e17;
int n, m, s, t, q[N], ql, qr;
```

```
int vis[N], fr[N];
ll dis[N];
vector<pii> G[N];
void MAIN() {
    cin >> n >> m >> s >> t;
    for (int i = 1; i <= m; i++) {
        int u, v, w;
        cin >> u >> v >> w;
        G[u].push_back({v, w});
    }
    for (int i = 0; i <= n; i++) dis[i] = inf;
    dis[s] = 0; q[qr] = s; vis[s] = 1;
    while (ql <= qr) {
        if (rng() % (qr - ql + 1) == 0) sort(q
+ ql, q + qr + 1, [](int x, int y) {
            return dis[x] < dis[y];
        });
        int u = q[ql++];
        vis[u] = 0;
        for (auto [v, w] : G[u]) {
            if (dis[u] + w < dis[v]) {
                dis[v] = dis[u] + w;
                fr[v] = u;
                if (!vis[v]) {
                    if (ql > 0) q[--ql] = v;
                    else q[++qr] = v;
                    vis[v] = 1;
                }
            }
        }
    }
    if (dis[t] == inf) {
        cout << "-1\n";
        return;
    }
    cout << dis[t] << ' ';
    vector<pii> stk;
    while (t != s) {
        stk.push_back({fr[t], t});
        t = fr[t];
    }
    reverse(stk.begin(), stk.end());
    cout << stk.size() << '\n';
    for (auto [u, v] : stk) cout << u << ' ' <<
v << '\n';
}
```

3.3. 差分约束

3.4. 竞赛图

3.5. 有向图强连通分量

3.5.1. Tarjan

3.5.2. Kosaraju

3.6. 连通分量

3.6.1. 割点

3.6.2. 桥

3.6.3. 点双

3.6.4. 边双

3.7. 二分图匹配