图论

人员

叶乐山、郭明瑞、齐中磊、张皓宁、曹承贤、陈瀚霄、范家畅、赵广宇、于跃 到课, 李政毅 线上

作业

https://vjudge.net/contest/745738, 课上讲了 A B C 这几道题, 课后作业是 D E F G H

课堂表现

今天的几道题目主要涉及图论的内容, 主要是 dijkstra、分层图、kruskal 等内容

课上有几位同学明显对于图论的模板这一部分内容掌握不熟, 课下要多花功夫把模板掌握熟练。

课堂内容

P1948 [USACO08JAN] Telephone Lines S

二分 + dijkstra

二分一个 mid, 看 check(mid) 能否完成任务

方法: 把 <=mid 的边权视为 0, 把 >mid 的边权视为 1, 看 1 到 n 之间的最短路有没有 <=k

(边权 01 的最短路, 也可以用 deque 来维护, 时间复杂度会比 dijkstra 的堆维护少一个 log)

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

using namespace std;

const int maxn = 1e3 + 5;
struct node {
  int to, value;
  bool operator > (const node& p) const { return value > p.value; }
};
vector<node> vec[maxn];
int n, p, k;
int dis[maxn];
bool st[maxn];

bool check(int mid) {
  memset(dis, 0x3f, sizeof(dis)); memset(st, false, sizeof(st));
  priority_queue<node, vector<node>, greater<node>> q; q.push({1,0}); dis[1] = 0;
  while (!q.empty()) {
    node u = q.top(); q.pop();
    int u_id = u.to, u_dis = u.value;
    if (st[u_id]) continue;
    st[u_id] = true;
```

```
    for (node it : vec[u_id]) {
      int to = it.to, value = (it.value<=mid ? 0 : 1);
      if (dis[to]==-1 || u_dis+value<dis[to]) {
        dis[to] = u_dis+value; q.push({to, dis[to]});
      }
    }
  }

  return dis[n] <= k;
}

int main()
{
  cin >> n >> p >> k;
  while (p -- ) {
    int u, v, w; cin >> u >> v >> w;
    vec[u].push_back({v,w}), vec[v].push_back({u,w});
  }

  int l = 0, r = 1e6;
  while (l <= r) {
    int mid = (l + r) / 2;
    if (check(mid)) r = mid-1;
    else l = mid+1;
  }

  if (l == 1e6+1) cout << -1 << endl;
  else cout << l << endl;
  return 0;
}
```

P4568 [JLOI2011] 飞行路线

分层图 + dijkstra求最短路

把整张图分成 k+1 层, 每层之间按照原本的边和边权进行建图, 第 i 层与第 i+1 层之间, 边权按照 0 来建图

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

using namespace std;

const int maxn = 2e5 + 5;
const int inf = 0x3f3f3f3f;
struct node {
  int to, value;
  bool operator > (const node& p) const { return value > p.value; }
};
vector<node> vec[maxn];
int n, m, k;
int getId(int u, int c) { return u*(k+1) + c; }
int dis[maxn];
```

```
bool st[maxn];

void dijkstra(int id) {
  memset(dis, 0x3f, sizeof(dis));
  priority_queue<node, vector<node>, greater<node>>q; q.push({id,0}); dis[id] = 0;
  while (!q.empty()) {
    node u = q.top(); q.pop();
    int u_id = u.to, u_dis = u.value;
    if (st[u_id]) continue;
    st[u_id] = true;

    for (node it : vec[u_id]) {
      if (u_dis + it.value < dis[it.to]) {
        dis[it.to] = u_dis + it.value; q.push({it.to, dis[it.to]});
      }
    }
  }
}

int main()
{
  cin >> n >> m >> k;
  int st, ed; cin >> st >> ed;
  while (m -- ) {
    int a, b, c; cin >> a >> b >> c;
    for (int i = 1; i <= k+1; ++i) {
      int a1 = getId(a,i), a2 = getId(a,i+1), b1 = getId(b,i), b2 = getId(b,i+1);
      vec[a1].push_back({b1,c}), vec[b1].push_back({a1,c});
      if (i < k+1) vec[a1].push_back({b2,0}), vec[b1].push_back({a2,0});
    }
  }

  dijkstra(getId(st,1));

  int res = inf;
  for (int i = 1; i <= k+1; ++i) res = min(res, dis[getId(ed,i)]);
  cout << res << endl;
  return 0;
}
```

P10928 走廊泼水节

最小生成树 拓展, 并查集维护每个联通块大小

把边权按照从小到大排序, 然后遍历每条边

对于第 i 条边来说, 如果其相连的两个点和边权分别是 a[i].u, a[i].v 和 a[i].w

那么相连后需要增加的边值为: (f(a[i].u) * f(a[i].v) - 1) * (a[i].w - 1)

其中, f(i) 代表点 i 所在的联通块的大小

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

using namespace std;

const int maxn = 6000 + 5;
struct node {
  int u, v, w;
} a[maxn];
bool cmp(node p, node q) { return p.w < q.w; }

int f[maxn], sz[maxn];
int fFind(int x) {
  if (f[x] != x) f[x] = fFind(f[x]);
  return f[x];
}

void solve() {
  int n; cin >> n;
  for (int i = 1; i <= n; ++i) f[i] = i, sz[i] = 1;
  for (int i = 1; i <= n-1; ++i) cin >> a[i].u >> a[i].v >> a[i].w;

  int res = 0;
  sort(a+1, a+n, cmp);
  for (int i = 1; i <= n-1; ++i) {
    int u = a[i].u, v = a[i].v, w = a[i].w;
    int fu = fFind(u), fv = fFind(v);
    int su = sz[fu], sv = sz[fv];
    res += (su*sv - 1) * (w+1);
    f[fu] = fv, sz[fv] += sz[fu];
  }

  cout << res << endl;
}

int main()
{
  int T; cin >> T;
  while (T -- ) solve();
  return 0;
}
```