# 第九周直播课（上午）作业讲评

洛谷网校

基础-提高衔接计划

2024-09

disangan233

洛谷

www.luogu.com.cn

# 【第九周】简单题

- T509457
- 买 $T$ 次卡牌，每次买了 $n$ 张
- 每张卡都有一个数字编号，不同次买的卡牌数量不一定相同
- 求每次购买的卡牌中，是否买到了编号为 $0$ 的卡牌

# 【第九周】简单题

- T509457
- 买 $T$ 次卡牌，每次买了 $n$ 张
- 每张卡都有一个数字编号，不同次买的卡牌数量不一定相同
- 求每次购买的卡牌中，是否买到了编号为 $0$ 的卡牌
- 本题考察对多测输入方面的理解与掌握
- 本题的大致思路：首先读入一个测试组数 $T$。循环 $T$ 次，每次读入一个 $n$ 后循环 $n$ 次，读入 $n$ 个数字
- 如果这 $n$ 个数字中有 $0$，则输出 yes，否则输出 no 即可

# 【第九周】简单题

- 一种常见的错误代码为：

```
for (int i = 1, x; i <= n; ++i) {
    cin >> x;
    if (x == 0) {
        puts("yes");
        ok = true;
        break; // 不能 break
    }
}
if (ok == false)
    puts("no");
```

- 读入 `2 3 1 0 1 3 1 0 0` 时，第二组 $n$ 会被读入为 $1$ 而不是 $3$

# 【第九周】简单题

```
while (T--) {
    cin >> n;
    bool ok = false;
    for (int i = 1, x; i <= n; ++i) {
        cin >> x;
        if (x == 0)
            ok = true;
    }
    if (ok == true)
        puts("yes");
    else
        puts("no");
}
```

# 【第九周】砍树

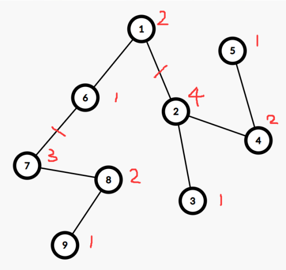
- T509453
- 有一棵 $n$ 个节点的树，可以在树上删除 $k$ 条边
- 求一种删除方案，使删除后的的图中，所有连通块含有节点数的最小值最大。输出这个最大值
- $\sum n \le 10^5$，$1 \le k < n$

# 【第九周】砍树

- 「最小值最大」是很经典的二分信号
- 如果删边后，最小值为可以为 $P$，那么也可以为 $P-1$，答案满足单调性
- 假设当前要判断 $x$ 是否满足答案，可以贪心地在树上删边
- 若当前节点的子树大小 $\geq x$（子树内可能已删除了一些边），我们可以把父边删掉，连通块数加一，$siz_x$ 清零，继续向上合并子树

# 【第九周】砍树

- 若求完根节点后，删除的边数 $\geq k+1$，则合法
- 时间复杂度 $O(n \log n)$
- $k+1$ 是因为删 $k$ 条边是自下向上，最后一个含根连通块的大小也应该 $\geq x$
- 以 $x=3, k=2$ 为例，此时 $x=3$ 不合法



# 【第九周】砍树

```
void dfs(int u, int fa) {
    siz[u] = 1;
    for (int v : e[u])
        if (v ^ fa) {
            dfs(v, u);
            siz[u] += siz[v];
        }
    if (siz[u] >= mid)
        siz[u] = 0, cnt++;
}
l = 1, r = n;
while (l <= r) {
    cnt = 0;
    dfs(1, 0);
    if (cnt >= k + 1)
        ans = mid, l = mid + 1;
    else
        r = mid - 1;
}
```

# 【第九周】最大和

- T509454
- 有一个 $n$ 个整数组成的数组 $a$，进行 $k$ 次操作
- 在一次操作中，选择数组 $a$ 的任意连续子数组（可能为空），并在数组的任意位置插入了该子数组的和
- 求经过 $k$ 次操作后，数组所有数的和的最大值，输出的结果对 $10^9+7$ 取模
- $n, k \le 10^6$，$|a_i| \le 10^9$

# 【第九周】最大和

- 如果只有一次操作，应该选取数组的最大子段和
- 最大子段和的 dp：$f_i = \max(f_{i-1}, 0) + a_i$
- $k$ 次操作呢？

# 【第九周】最大和

- 如果只有一次操作，应该选取数组的最大子段和
- 最大子段和的 dp：$f_i = \max(f_{i-1}, 0) + a_i$
- $k$ 次操作呢？
- 每次把和插入到最大子段和后，下一次最大子段和翻倍

$$\{3, -5\} \to \{3, 3, -5\} \to \{3, 3, 6, -5\} \to \{3, 3, 6, 12, -5\}$$

- 设最大子段和为 $mx$，$k$ 次操作会让数组和增加 $mx \times (2^k - 1)$
- 设原数组和为 $sum$，那么答案为 $sum - mx + mx \times 2^k$
- 注意 $sum, mx$ 都是 `long long`，要先取模再乘 `int`

# 【第九周】最大和

```
scanf("%d%d", &n, &k);
for (int i = 1; i <= n; i++)
    scanf("%d", &a[i]);
for (int i = 1; i <= n; i++) {
    f[i] = max(0ll, f[i - 1]) + a[i];
    sum += a[i];
}
for (int i = 1; i <= n; i++)
    mx = max(mx, f[i]);
pw[0] = 1;
for (int i = 1; i <= k; i++)
    pw[i] = 2ll * pw[i - 1] % p;
ans = ((sum - mx) % p + p) % p;
ans = (ans + mx % p * pw[k] % p) % p;
printf("%d\n", ans);
```

# 【第九周】分身

- T509455
- 有 $n$ 个分身，第 $i$ 个分身的战力为 $a_i$
- 初始 $n$ 个分身都在场，它们会进行 $n-1$ 次战斗。每次战斗选取两个在场的分身 $i,j$ 对战，可以获得 $a_i \operatorname{xor} a_j$ 的分数
- 每次战斗结束后，其中一个分身会落败退场。指定每次战斗的分身编号及胜负，使得总分最大，输出最大总分
- $n \leq 2000$，$1 \leq a_i \leq 10^9$

# 【第九周】分身

- 最小生成树模板题
- 对于一棵生成树，指定每个点负于其父节点，便可得到边权之和的总分
- 等价于求完全图的一棵最大生成树，使用 Kruskal 算法
- 时间复杂度 $O(n^2 \log n)$

# 【第九周】分身

```
for (int i = 1; i <= n; i++)
    for (int j = i + 1; j <= n; j++)
        add(i, j, c[i] ^ c[j]);
sort(a + 1, a + cnt + 1, cmp);
for (int i = 1; i <= n; i++)
    f[i] = i;
for (int i = 1; i <= cnt; i++) {
    int u = find(a[i].u);
    int v = find(a[i].v);
    if (u == v)
        continue;
    f[u] = v;
    ans += a[i].w;
    if ((++d) == n - 1)
        break;
}
```

# 【第九周】字符串

- T509456
- 给定字符串 $S$，可以改变其中不超过 $k$ 个字符
- 定义一个字符串的美丽度为，对所有相邻的字符对 $(x, y)$，在总和中累加 $c(x, y)$ 后的值
- 求修改后的字符串可能的最高美丽度
- 对于所有数据，$1 \le k \le |S| \le 100$，$1 \le n \le 26^2$，$|c(x, y)| \le 10^3$，所有字符均为小写英文字母

# 【第九周】字符串

- $|S| \le 100$，是很经典的 $n$ 选 $k$ 最优化问题，考虑 dp
- 令 $f_{i,j,k}$ 表示前 $i$ 个字符中修改了 $j$ 次，当前位为 $k$ 的最大美丽度
- 每一次枚举相邻的两位转移，共 $26^2$ 种情况

$$f_{i+1,j,k} = \max_{0 \le p < 26} \{ f_{i,j-[s_{i+1} \ne k],p} + c_{p,k} \}$$

- 注意不一定改变完 $k$ 次最优，答案为

$$\max_{0 \le i \le k} \max_{0 \le j < 26} f_{n,i,j}$$

- 时间复杂度 $O(|S| \times k \times |\Sigma|^2)$

# 【第九周】字符串

```
void Max(int &x, int y) {
    if (x < y)
        x = y;
}
cin >> s;
n = s.length();
for (int i = 0; i < n; i++)
    s[i] -= 'a';
cin >> k >> m;
for (int i = 1; i <= m; i++) {
    char f, t;
    int b;
    cin >> f >> t >> b;
    d[f - 'a'][t - 'a'] = b;
}
```

# 【第九周】字符串

```
memset(f, -63, sizeof(f));
for (int i = 0; i < 26; i++)
    f[0][s[0] != i][i] = 0;
for (int i = 0; i < n - 1; i++)
for (int j = 0; j <= k; j++)
for (int p = 0; p < 26; p++)
if (f[i][j][p] > -1e9)
for (int q = 0; q < 26; q++)
    Max(f[i+1][j+(s[i+1]!=q)][q],f[i][j][p]+d[p][q]);
int ans = -1e9;
for (int i = 0; i <= k; i++)
    for (int c = 0; c < 26; c++)
        Max(ans, f[n - 1][i][c]);
cout << ans << endl;
```

# 买礼物

- P1194
- 需要买 $B$ 个东西，每个初始价格均为 $A$ 元
- 买了第 $i$ 个东西，再买第 $j$ 个时，只需要 $K_{i,j}$ 元
- 求最小总花费
- $B \leq 500$，$A, K_{i,j} \leq 1000$

# 买礼物

- 买了第 $i$ 个再买第 $j$ 个相当于遍历 $i \to j$ 边
- 直接建完全图跑最小生成树？
- 每个东西需要 $A$ 的初始价格来启动后续的连边
- 所以需要新建一个 $0$ 号节点，代表以初始价格购买，向每一个节点连一条边权为 $A$ 的边
- 再求 MST 即可

# 买礼物

```
for (int i = 1; i <= n; i++)
    for (int j = i + 1; j <= n; j++)
        add(i, j, k[i][j]);
for (int i = 1; i <= n; i++)
    add(0, i, A);
sort(a + 1, a + cnt + 1);
for (int i = 0; i <= n; i++)
    f[i] = i;
for (int i = 1; i <= cnt; i++) {
    int u = find(a[i].u);
    int v = find(a[i].v);
    if (u == v)
        continue;
    f[u] = v;
    ans += a[i].w;
    if ((++d) == n)
        break;
}
```

# 会议

- P1395
- 给定一棵 $n$ 个点的树，求一个点使得所有点到该点距离之和最小
- $n \leq 5 \times 10^4$

# 会议

- P1395
- 给定一棵 $n$ 个点的树，求一个点使得所有点到该点距离之和最小
- $n \leq 5 \times 10^4$
- 树的重心：找到一个点，删去该点后，所有的子树中最大的子树节点数最少
- 题目所求是树的重心的另一个性质
- 反证，若最优解不是树的重心，将答案向重心方向移动一步，距离之和一定变小，与最优解矛盾，所以最优解一定是树的重心

# 会议

- 树的重心如果不唯一，则至多有两个，且这两个重心相邻
- 以树的重心为根时，所有子树的大小都不超过整棵树大小的一半
- 在 DFS 中计算每个子树的大小，记录「向下」的子树的最大大小，利用总点数 - 当前子树（这里的子树指有根树的子树）的大小得到「向上」的子树的大小，然后就可以依据定义找到重心了
- 时间复杂度 $O(n)$

# 会议

```cpp
void dfs(int u, int fa) {
    siz[u] = 1;
    w[u] = 0;
    for (int v : e[u]) {
        if (v != fa) {
            dfs(v, u);
            siz[u] += siz[v];
            w[u] = max(w[u], siz[v]);
        }
    }
    w[u] = max(w[u], n - siz[u]);
    if (w[u] <= n / 2) {  // 依照树的重心的定义统计
        centroid[centroid[0] != 0] = cur;
    }
}
```

# [USACO3.1] 最短网络 Agri-Net

- P1546
- FJ 已经给他的农场安排了一条高速的网络线路，他想把这条线路共享给其他农场。为了用最小的消费，他想铺设最短的光纤去连接所有的农场
- 你将得到一份各农场之间连接费用的列表，你必须找出能连接所有农场并所用光纤最短的方案
- $n \leq 100$

# [USACO3.1] 最短网络 Agri-Net

- P1546
- FJ 已经给他的农场安排了一条高速的网络线路，他想把这条线路共享给其他农场。为了用最小的消费，他想铺设最短的光纤去连接所有的农场
- 你将得到一份各农场之间连接费用的列表，你必须找出能连接所有农场并所用光纤最短的方案
- $n \leq 100$
- 最小生成树模板

# 炸铁路

- P1656
- 给定一个 $n$ 个点 $m$ 条边的无向图，求无向图的所有割边
- 割边（桥）：无向图中如果删去这条边就会使图不连通的边
- $n \leq 150$，$m \leq 5000$

# 炸铁路

- P1656
- 给定一个 $n$ 个点 $m$ 条边的无向图，求无向图的所有割边
- 割边（桥）：无向图中如果删去这条边就会使图不连通的边
- $n \leq 150$，$m \leq 5000$
- 正解是 tarjan 算法，可以自行了解，时间复杂度 $O(n+m)$
- 如何用我们学过的知识通过此题呢？

# 炸铁路

- P1656
- 给定一个 $n$ 个点 $m$ 条边的无向图，求无向图的所有割边
- 割边（桥）：无向图中如果删去这条边就会使图不连通的边
- $n \leq 150$，$m \leq 5000$
- 正解是 tarjan 算法，可以自行了解，时间复杂度 $O(n+m)$
- 如何用我们学过的知识通过此题呢？
- 枚举每条边作为删除边，判断新图是否连通，如果不连通则是割边
- 可以用并查集或者直接 dfs 的方法来判断
- 时间复杂度 $O(m^2)$

# 炸铁路

```
void check(int t)  //删除第 t 条边图是否不连通
{
    for(int i=1;i<=n;i++)
        fa[i]=i;
    for(int i=1;i<=m;i++)
        if(i!=t&&find(e[i].u)!=find(e[i].v))
            fa[find(e[i].u)]=find(e[i].v);
    for(int u=find(1),i=2;i<=n;i++)
    if(find(i)!=u)
    {
        a[++tot]=e[t];
        break;
    }
}
```

# [CCC 2023 S2] Symmetric Mountains

- P9325
- 给定一个长度为 $n$ 的数列 $a$，定义一个区间的不对称性为，在区间中位于对称位置的若干个点差的绝对值的总和
- 对于 $1 \le i \le n$，求出长度为 $i$ 的所有区间最小的不对称性
- $n \le 5000$，$a_i \le 10^5$

# [CCC 2023 S2] Symmetric Mountains

- P9325
- 给定一个长度为 $n$ 的数列 $a$，定义一个区间的不对称性为，在区间中位于对称位置的若干个点差的绝对值的总和
- 对于 $1 \le i \le n$，求出长度为 $i$ 的所有区间最小的不对称性
- $n \le 5000$，$a_i \le 10^5$
- 首先有 $O(n^3)$ 的暴力做法
- 枚举每一个长度为 $i$ 的区间，枚举出起点 $k$，从 $k$ 遍历到 $k+i-1$ 求不对称性
- $k+j$ 的对称位置是 $k+i-1-j$

# [CCC 2023 S2] Symmetric Mountains

- 暴力做法的复杂度瓶颈在于每次都要遍历整个区间
- 如何更高效地求出区间的不对称性？
- 对于区间 $[i, j]$，假设已经求出了其不对称性
- 对于区间 $[i-1, j+1]$，它的中点和 $[i-1, j+1]$ 相同，只需要加上 $|a_{i-1} - a_{j+1}|$ 就可以得到其不对称性
- 所以可以由长度为 $i$ 的区间推出长度为 $i+2$ 的同中点区间
- 那么就可以区间 dp 了

# [CCC 2023 S2] Symmetric Mountains

- 令 $f_{i,j}$ 为区间 $[i,j]$ 的不对称性，有

$$f_{i-1,j+1}=f_{i,j}+|a_{i-1}-a_{j+1}|$$

- 时间复杂度 $O(n^2)$

```
for(int i=0;i<n;i++){
    mn=1e9;
    for(int j=1;i+j<=n;j++){
        int l=i+j;
        f[j][l]=f[j+1][l-1]+abs(a[j]-a[l]);
        mn=min(mn,f[j][l]);
    }
    cout<<mn<<' ';
}
```

# 狗哥采矿

- P2380
- 给定 $n \times m$ 的矩阵，第 $i$ 行 $j$ 列有 $a_{i,j}$ 种矿石 A，$b_{i,j}$ 种矿石 B
- 矿石 A 只能向左运输，矿石 B 只能向上运输，传输到左边界的矿石 A 会被采集，传输到上边界的矿石 B 会被采集
- 给每个位置设置向左或者向上的传送带，传输过程中不能转弯，只能走直路
- 求最多能采到的矿
- $n, m \leq 500$

# 狗哥采矿

- 由于传输只能是直线，如果一个位置要采集矿石 A，它左边的所有位置都只能采集矿石 A，矿石 B 同理
- 令 $f_{i,j}$ 为 $(1,1)$ 到 $(i,j)$ 能采集的最多矿石，我们只关心 $(i,j)$ 传送带的方向，有转移

$$f_{i,j} = \max\left\{ f_{i-1,j} + \sum_{k=1}^{j} a_{i,k}, \quad f_{i,j-1} + \sum_{k=1}^{i} b_{k,j} \right\}$$

- 维护每行 $a_{i,j}$ 的前缀和、每列 $b_{i,j}$ 的前缀和，便可 $O(1)$ 转移
- 时间复杂度 $O(nm)$

# 狗哥采矿

```
for(int i=1; i<=n; i++) {
    for(int j=1; j<=m; j++) {
        s[i][j]=s[i][j-1]+a[i][j];
    }
}
for(int i=1; i<=n; i++) {
    for(int j=1; j<=m; j++) {
        t[i][j]=t[i-1][j]+b[i][j];
    }
}
for(int i=1; i<=n; i++) {
    for(int j=1; j<=m; j++) {
        f[i][j]=max(f[i][j-1]+t[i][j],f[i-1][j]+s[i][j]);
    }
}
```

# [USACO08JAN] Cow Contest S

- P2419
- 给定长为 $n$ 的值未知的数列 $a$，有 $m$ 对关系，第 $i$ 对关系 $(u_i, v_i)$ 表示 $a_{u_i} > a_{v_i}$
- 求有多少个数的排名确定
- $n \leq 100$

# [USACO08JAN] Cow Contest S

- P2419
- 给定长为 $n$ 的值未知的数列 $a$，有 $m$ 对关系，第 $i$ 对关系 $(u_i, v_i)$ 表示 $a_{u_i} > a_{v_i}$
- 求有多少个数的排名确定
- $n \le 100$
- $a_u > a_v$ 可以看成是一条 $u \to v$ 的有向边
- 对于点 $x$，比它小的数是它能到达的数，比它大的数是能到达它的数
- 只要两种数加起来的和是 $n-1$，这个数的排名就确定了
- 可以用 floyd 或者 dfs 来判断连通性

# [USACO08JAN] Cow Contest S

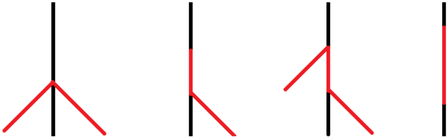
```
for(int i=1;i<=m;i++) {
    int x,y;
    scanf("%d%d",&x,&y);
    f[x][y]=1;
}
for(int k=1;k<=n;k++)
    for(int i=1;i<=n;i++)
        for(int j=1;j<=n;j++)
            f[i][j]|=(f[i][k]&f[k][j]);
for(int i=1;i<=n;i++) {
    cnt=0;
    for(int j=1;j<=n;j++)
        if(f[i][j]==1||f[j][i]==1) cnt++;
    if(cnt==n-1) ans++;
}
```

# 仓鼠找 sugar

- P3398
- 给定一棵 $n$ 的点的树，$q$ 次询问，每次询问点 $a$ 到点 $b$ 的路径是否和点 $c$ 到点 $d$ 的路径相交
- $n, q \leq 10^5$

# 仓鼠找 sugar

- P3398
- 给定一棵 $n$ 的点的树，$q$ 次询问，每次询问点 $a$ 到点 $b$ 的路径是否和点 $c$ 到点 $d$ 的路径相交
- $n, q \leq 10^5$
- 如果两个路径相交，必有一个路径的 lca 在另一条路径上



# 仓鼠找 sugar

- 判断点 $w$ 在点 $u$ 到点 $v$ 的路径上

$$\mathrm{dis}(u, w) + \mathrm{dis}(w, v) = \mathrm{dis}(u, v)$$

- 使用倍增求 lca 即可，维护树上 $\mathrm{dis}$

$$\mathrm{dis}(u, v) = dep_u + dep_v - 2 \times dep_{lca(u,v)}$$

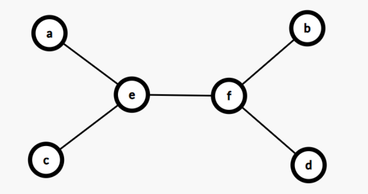
# 仓鼠找 sugar

- 当然，还有更简单的判断路径 $ab, cd$ 相交的办法

$$\mathrm{dis}(a,b)+\mathrm{dis}(c,d)\geq \mathrm{dis}(a,c)+\mathrm{dis}(b,d)$$

- 证明：相交时，

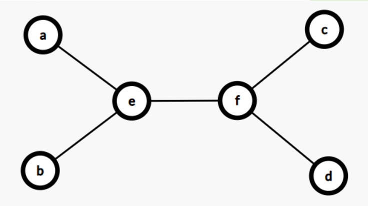
$$\mathrm{dis}(a,b)+\mathrm{dis}(c,d)-\mathrm{dis}(a,c)-\mathrm{dis}(b,d)=2\mathrm{dis}(e,f)\geq 0$$



# 仓鼠找 sugar

- 无交时,

$$\mathrm{dis}(a,b)+\mathrm{dis}(c,d)-\mathrm{dis}(a,c)-\mathrm{dis}(b,d)=-2\mathrm{dis}(e,f)<0$$




- 得证

# 仓鼠找 sugar

```
int lca(int x,int y) {
    if(d[x]>d[y]) swap(x,y);
    for(int i=t;i>=0;i--)
        if(d[f[y][i]]>=d[x])
            y=f[y][i];
    if(x==y) return x;
    for(int i=t;i>=0;i--)
        if(f[x][i]!=f[y][i])
            x=f[x][i],y=f[y][i];
    return f[x][0];
}
int dis(int a,int b) {return d[a]+d[b]-2*d[lca(a,b)];}
for(int i=1;i<=m;i++) {
    scanf("%d%d%d%d",&x,&y,&u,&v);
    if(dis(x,y)+dis(u,v)>=dis(x,u)+dis(y,v))
        printf("Y\n");
    else
        printf("N\n");
}
```

# 走廊泼水节

- P10928
- 给定一棵 $n$ 个节点的树，要求增加若干条边，把这棵树扩充为完全图，并满足图的唯一最小生成树仍然是这棵树
- 求增加的边的权值总和最小是多少
- $n \leq 6000$

# 走廊泼水节

- 考虑 Kruskal 是怎么求最小生成树的
- 枚举到边 $\{u, v, w\}$ 时，并查集合并 $u, v$ 两个连通块，答案累加上 $w$
- 要满足完全图的最小生成树仍是这棵树，那么 $u, v$ 两个连通块只能由 $w$ 边权来合并
- 并查集时维护连通块大小 $siz$，此时把连通块 $u, v$ 合并成完全图需要连 $siz_u \times siz_v$ 条边
- 这些边的边权都不能 $\le w$，所以连的新边的总边权是 $(siz_u \times siz_v - 1) \times (w + 1)$，然后合并连通块大小
- 每合并一次得到更大的一个完全图，求出 MST 后就得到了需要加边的总边权

# 走廊泼水节

```
int find(int x) {
    return fa[x] == x ? x : fa[x] = find(fa[x]);
}
void merge(int u, int v, int w) {
    u = find(u), v = find(v);
    if(siz[u] < siz[v]) // 按秩合并
        swap(u, v);
    ans += (siz[u] * siz[v] - 1) * (w + 1);
    fa[v] = u;
    siz[u] += siz[v];
}
```