

# 综合练习

## 人员

刘奕辰、杨俊彦、袁晨峻、隋天乙 到课, 李锦澍 线上

## 上周作业检查

上周作业链接: <https://cppoj.kids123code.com/contest/2169>

The screenshot shows a competition results page with a table of scores. The table has columns for rank (#), username, name, programming score (编程分), time (时间), and four specific problem scores (A, B, C, D). The participants and their scores are:

| # | 用户名         | 姓名  | 编程分 | 时间    | A   | B   | C   | D   |
|---|-------------|-----|-----|-------|-----|-----|-----|-----|
| 1 | yangjunyan  | 杨俊彦 | 400 | 6231  | 100 | 100 | 100 | 100 |
| 2 | liuyichen   | 刘奕辰 | 400 | 6647  | 100 | 100 | 100 | 100 |
| 3 | chenxinmiao | 陈欣妙 | 400 | 10314 | 100 | 100 | 100 | 100 |
| 4 | lijinshu    | 李锦澍 | 200 | 5668  | 100 | 100 |     |     |

## 本周作业

<https://cppoj.kids123code.com/contest/2238> (课上讲了 A ~ B 题, 课后作业是 C 题必做, D 题选做)

## 课堂表现

今天课上的 A B 题整体会复杂一些, 用到了之前学的 逆元、组合数、树状数组 等内容, 同学们课上没写完的题课下要补完。

## 课堂内容

### 送礼物 (上周作业)

折半搜索, 对左部分做一次搜索, 右部分做一次搜索, 最后综合统计答案即可

```
#include <bits/stdc++.h>
#define int long long

using namespace std;

const int maxn = 1e7 + 5;
int w[maxn];
vector<int> v1, v2;

void dfs(int u, int ed, int sum, vector<int>& vec) {
    if (u == ed+1) {
        vec.push_back(sum); return;
    }
}
```

```

dfs(u+1, ed, sum, vec);
dfs(u+1, ed, sum+w[u], vec);
}

signed main()
{
    int m, n; cin >> m >> n;
    for (int i = 1; i <= n; ++i) cin >> w[i];

    int mid = n / 2;
    dfs(1, mid, 0, v1); dfs(mid+1, n, 0, v2);
    sort(v1.begin(), v1.end()), sort(v2.begin(), v2.end());

    int res = 0;
    for (int i : v1) {
        int pos = upper_bound(v2.begin(), v2.end(), m-i) - v2.begin() - 1;
        if (pos >= 0) res = max(res, i + v2[pos]);
    }
    cout << res << endl;
    return 0;
}

```

## Red and Blue Graph

一开始可以先默认所有点都是蓝色的, 然后要求选择 K 个点, 把这 K 个点变红

如果一个点的度数是奇数, 把这个点变红之后, 连接不同颜色的边的数量会加一个奇数, 偶数则反之

因此, 题意可以转化为, 选 K 个点, 要求这 K 个点的总度数为偶数, 问有多少方案

```

#include <bits/stdc++.h>

using namespace std;

typedef long long LL;
const int maxn = 2e5 + 5;
const int mod = 998244353;
int deg[maxn];
int fac[maxn], inv_fac[maxn];

int qmod(int a, int b) {
    int res = 1;
    while (b) {
        if (b&1) res = (LL)res*a % mod;
        a = (LL)a*a % mod;
        b >>= 1;
    }
    return res;
}

int C(int n, int m) { return (LL)fac[n]*inv_fac[n-m]%mod*inv_fac[m]%mod; }

```

```

int main()
{
    fac[0] = inv_fac[0] = 1;
    for (int i = 1; i < maxn; ++i) {
        fac[i] = (LL)fac[i-1]*i % mod;
        inv_fac[i] = qmod(fac[i], mod-2);
    }

    int n, m, K; cin >> n >> m >> K;
    while (m -- ) {
        int a, b; cin >> a >> b; ++deg[a], ++deg[b];
    }

    int odd = 0, even = 0;
    for (int i = 1; i <= n; ++i) {
        if (deg[i]&1) ++odd;
        else ++even;
    }

    int res = 0;
    for (int i = 0; i <= odd; i += 2) {
        int j = K - i;
        if (j > even || j < 0) continue;
        int nums = (LL)C(odd,i) * C(even,j) % mod;
        res = (res + nums) % mod;
    }
    cout << res << endl;
    return 0;
}

```

## Double Chance

当把  $a[i]$  考虑上时, 此时需要快速知道  $a[1] \sim a[i-1]$  中有多少比  $a[i]$  大的和有多少比  $a[i]$  小的以及他们的和, 这个可以用两个树状数组来维护

```

#include <bits/stdc++.h>
#define int long long

using namespace std;

const int maxn = 2e5 + 5;
const int mod = 998244353;
int tr_1[maxn], tr_2[maxn];

int qmod(int a, int b) {
    int res = 1;
    while (b) {
        if (b&1) res = res*a % mod;
        a = a*a % mod;
        b >>= 1;
    }
}

```

```
    }
    return res;
}
int inv(int x) { return qmod(x, mod-2); }

int lowbit(int x) { return x&(-x); }
void update(int tr[], int x, int k) {
    while (x < maxn) { tr[x] += k; x += lowbit(x); }
}
int query(int tr[], int x) {
    int res = 0;
    while (x) { res += tr[x]; x -= lowbit(x); }
    return res;
}

signed main()
{
    int n, res = 0, sum = 0; cin >> n;
    for (int i = 1; i <= n; ++i) {
        int x; cin >> x;

        int cnt = query(tr_1, x);
        int value = sum - query(tr_2, x);
        res += 2*cnt*x + 2*value + x; res %= mod;

        cout << res * inv(i*i%mod) % mod << endl;
        update(tr_1, x, 1); update(tr_2, x, x); sum += x;
    }
    return 0;
}
```