

# 综合练习

## 人员

王毅博、褚锦轩、王承周、许睿谦 到课

## 上周作业检查

上周作业链接: <https://cppoj.kids123code.com/contest/1342>

#	用户名	姓名	编程分	时间	A	B	C	D
1	wangchengzhou	王承周	300	1505	100	100	100	
2	ruanwenzhang	阮文璋	300	3009	100	100	100	
3	dongyuhan	董昱含	200	1233	100	100		
4	xuruiqian	许睿谦	200	1767	100	100	0	
5	chujinxuan	褚锦轩	140	1968	100	40		
6	wangyibo	王毅博	100	1656	100			

## 本周作业

<https://cppoj.kids123code.com/contest/1479> (课上讲了 A ~ C 题, 课后作业是 D 题)

## 课堂表现

今天的第二题思路并不复杂, 就是代码稍微复杂一点, 大部分同学课上没写完, 课下需要好好补一下这道题。

## 课堂内容

### 平铺图案 (上周作业)

二维前缀和直接维护即可

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

using namespace std;

const int maxn = 1000 + 5;
char s[maxn][maxn];
int p[maxn][maxn];

int calc(int n, int x, int y) {
    int c1 = x / n, c2 = y / n;
    int sx = x % n, sy = y % n;
    int res = c1 * c2 * p[n][n] + c1 * p[n][sy] + c2 * p[sx][n] + p[sx][sy];
}
```

```

    return res;
}

signed main()
{
    int n, m; cin >> n >> m;
    for (int i = 1; i <= n; ++i) cin >> (s[i]+1);
    for (int i = 1; i <= n; ++i) {
        for (int j = 1; j <= n; ++j) {
            p[i][j] = p[i-1][j] + p[i][j-1] - p[i-1][j-1] + (s[i][j]=='B');
        }
    }
    while (m -- ) {
        int a, b, c, d; cin >> a >> b >> c >> d;
        ++a, ++b, ++c, ++d;
        // cout << "----- ";
        cout << calc(n,c,d) - calc(n,a-1,d) - calc(n,c,b-1) + calc(n,a-1,b-1) << endl;
    }
    return 0;
}

```

## [USACO06DEC] Milk Patterns G

二分一个 mid, 判断是否有 长度为mid 的子数组重复了  $\geq K$  次

判断可以用 hash 维护, 把每个长度为 mid 的子数组的 hash值 求出来, 看是否有一个 hash值 重复出现了  $\geq K$  次

```

#include <bits/stdc++.h>

using namespace std;

typedef unsigned ULL;
const int maxn = 20000 + 5;
const int P = 1e9+7;
int w[maxn];
ULL p[maxn], h[maxn];

ULL get_hash(int l, int r) { return h[r] - h[l-1]*p[r-l+1]; }

bool check(int n, int K, int mid) {
    map<ULL, int> mp;
    for (int i = 1, j = mid; j <= n; ++i, ++j) mp[get_hash(i,j)]++;
    for (auto it : mp) {
        if (it.second >= K) return true;
    }
    return false;
}

int main()

```

```

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

    p[0] = h[0] = 1;
    for (int i = 1; i <= n; ++i) {
        p[i] = p[i-1] * P, h[i] = h[i-1] * P + w[i];
    }

    int l = 1, r = n;
    while (l <= r) {
        int mid = (l + r) / 2;
        if (check(n, K, mid)) l = mid+1;
        else r = mid-1;
    }
    cout << r << endl;
    return 0;
}

```

## Road Blocked

考虑删边操作是比较复杂的, 因为删完边后点与点之间的最短路变化可能变化很大

因此, 可以把整个问题反过来思考, 考虑加边操作

每次加完一条边后, 最短路的更新只可能通过这条边进行更新, 所以最短路的更新是  $O(n^2)$  级别的

题目保证, 最多添加 300 条边, 那么这个题就解决了

```

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

using namespace std;

const int N = 300 + 5, M = 2e5 + 5;
const int inf = 0x3f3f3f3f3f3f3f3f;
struct Edge {
    int a, b, c;
} w[N*N];
bool st[N*N];
int f[N][N];

struct node {
    int op, id, x, y;
} q[M];

signed main()
{
    int n, m, Q; cin >> n >> m >> Q;
    for (int i = 1; i <= m; ++i) cin >> w[i].a >> w[i].b >> w[i].c, st[i] = true;
    for (int i = 1; i <= Q; ++i) {
        cin >> q[i].op;
    }
}

```

```

if (q[i].op == 1) cin >> q[i].id, st[q[i].id] = false;
else cin >> q[i].x >> q[i].y;
}

memset(f, 0x3f, sizeof(f));
for (int i = 1; i <= n; ++i) f[i][i] = 0;
for (int i = 1; i <= m; ++i) {
    int a = w[i].a, b = w[i].b, c = w[i].c;
    if (st[i]) f[a][b] = f[b][a] = c;
}

for (int k = 1; k <= n; ++k) {
    for (int i = 1; i <= n; ++i) {
        for (int j = 1; j <= n; ++j) f[i][j] = min(f[i][j], f[i][k]+f[k][j]);
    }
}

vector<int> ans;
for (int i = Q; i >= 1; --i) {
    int op = q[i].op, id = q[i].id, x = q[i].x, y = q[i].y;
    if (op == 1) {
        int a = w[id].a, b = w[id].b, c = w[id].c;
        f[a][b] = min(f[a][b], c);
        for (int i = 1; i <= n; ++i) {
            for (int j = 1; j <= n; ++j) {
                int value = min(f[i][a]+f[b][j], f[i][b]+f[a][j]) + f[a][b];
                f[i][j] = min(f[i][j], value);
            }
        }
    } else {
        int dis = (f[x][y]==inf ? -1 : f[x][y]);
        ans.push_back(dis);
    }
}

reverse(ans.begin(), ans.end());
for (int i : ans) cout << i << endl;
return 0;
}

```

## [蓝桥杯 2025 省 Java B] 数组翻转

找到每个数出现的最长的连续两段的长度

```

#include <bits/stdc++.h>

using namespace std;

typedef long long LL;
const int maxn = 1e6 + 5;
int w[maxn];

```

```
vector<int> vec[maxn];

int main()
{
    int n; cin >> n;
    for (int i = 1; i <= n; ++i) cin >> w[i];
    int cnt = 1;
    for (int i = 2; i <= n; ++i) {
        if (w[i] == w[i-1]) ++cnt;
        else vec[w[i-1]].push_back(cnt), cnt = 1;
    }
    vec[w[n]].push_back(cnt);

    LL res = 0;
    for (int i = 1; i < maxn; ++i) {
        sort(vec[i].begin(), vec[i].end()), reverse(vec[i].begin(), vec[i].end());
        int len = 0;
        if ((int)vec[i].size() >= 1) len += vec[i][0];
        if ((int)vec[i].size() >= 2) len += vec[i][1];
        res = max(res, (LL)len * i);
    }
    cout << res << endl;
    return 0;
}
```