

折半搜索

人员

阮文璋、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/2021>

🏠 比赛概况

📋 题目列表

📋 选择题列表

📋 提交记录

★ 实时榜单

★ 选择题排行榜

王向东老师周日一点半C++综合练习

🔄 刷新

#	用户名	姓名	编程分	时间	A	B	C	D	E
1	ruanwenzhang	阮文璋	400	5006	100	100	100	100	
2	wangchengzhou	王承周	317	2244	100	100	100	17	
3	chujinxuan	褚锦轩	300	1302	100	100	100		
4	xuruiqian	许睿谦	300	1982	100	100	100		
5	wangyibo	王毅博	273	1077	100	100	73		

本周作业

<https://cppoj.kids123code.com/contest/2171> (课上讲了 A ~ C 题, 课后作业是 D 题)

课堂表现

今天讲了 折半搜索 这个算法, 同学们课上整体听课吸收的都很不错。

课堂内容

Go Stone Puzzle (上周作业)

跟 string 变化相关的 bfs

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

using namespace std;

map<string, int> mp;

int main()
{
    int n; string _st, _ed; cin >> n >> _st >> _ed;
    _st += " ", _ed += " ";
    queue<string> q; q.push(_st); mp[_st] = 0;
    while (!q.empty()) {
        string u = q.front(); q.pop();
        if (u == _ed) { cout << mp[u] << endl; return 0; }
    }
```

```

int pos = u.find(" ");
for (int i = 0; i <= n; ++i) {
    if (u[i]==' ' || u[i+1]==' ') continue;
    string t = u;
    swap(t[i], t[pos]), swap(t[i+1], t[pos+1]);
    if (!mp.count(t)) q.push(t), mp[t] = mp[u]+1;
}
}
cout << -1 << endl;
return 0;
}

```

[CEOI 2015] 世界冰球锦标赛 (Day2)

折半搜索模板题, 直接 2^{40} 会超时, 所以对前一部分进行 2^{20} 的搜索, 对后一部分也进行 2^{20} 的搜索, 然后把两次的结果记录下来最终求答案

```

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

using namespace std;

const int maxn = 2e6 + 5;
int w[maxn];
int v1[maxn], v2[maxn];
int cnt1 = 0, cnt2 = 0;

void dfs(int u, int ed, int sum, int v[], int& cnt) {
    if (u == ed+1) {
        ++cnt; v[cnt] = sum;
        return;
    }

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

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

    int mid = n / 2;
    dfs(1, mid, 0, v1, cnt1); dfs(mid+1, n, 0, v2, cnt2);
    sort(v1+1, v1+cnt1+1); sort(v2+1, v2+cnt2+1);

    int res = 0;
    for (int i = 1; i <= cnt1; ++i) {
        int pos = upper_bound(v2+1, v2+cnt2+1, m-v1[i]) - v2 - 1;
        res += pos;
    }
}

```

```

    }
    cout << res << endl;
    return 0;
}

```

XOR on Grid Path

从左上角往右下做一次搜索, 再从右下角往左上做一次搜索, 把搜到的结果用 map 存下来, 最后统计方案数即可

```

#include <bits/stdc++.h>

using namespace std;

typedef long long LL;
const int maxn = 20 + 5;
int w[maxn][maxn];
map<int, int> mp1[maxn][maxn], mp2[maxn][maxn];
int n;

void dfs1(int x, int y, int value) {
    if (x+y == n+1) { mp1[x][y][value^w[x][y]]++; return; }

    dfs1(x+1, y, value^w[x][y]);
    dfs1(x, y+1, value^w[x][y]);
}

void dfs2(int x, int y, int value) {
    if (x+y == n+2) { mp2[x][y][value^w[x][y]]++; return; }

    dfs2(x-1, y, value^w[x][y]);
    dfs2(x, y-1, value^w[x][y]);
}

int main()
{
    cin >> n;
    for (int i = 1; i <= n; ++i) {
        for (int j = 1; j <= n; ++j) cin >> w[i][j];
    }

    dfs1(1, 1, 0); dfs2(n, n, 0);

    LL res = 0;
    for (int x = 1, y = n; x <= n; ++x, --y) {
        for (auto it : mp1[x][y]) {
            int val = it.first, cnt = it.second;
            res += (LL)cnt * mp2[x+1][y][val];
            res += (LL)cnt * mp2[x][y+1][val];
        }
    }
    cout << res << endl;
}

```

```
    return 0;
}
```

[POI 2000] 公共串

二分 + hash

二分一个长度 mid, 在每个字符串中, 记录每个长度为 mid 的子串的 hash 值, 最后看是否有一个 hash 值在所有字符串中都有

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

using namespace std;

typedef unsigned long long ULL;
const int N = 5 + 5, M = 2000 + 5;
const int P = 131;
char s[N][M];
int len[N];
ULL h[N][M], p[M];
set<ULL> st[N];

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

bool check(int n, int mid) {
    for (int i = 1; i <= n; ++i) {
        st[i].clear();
        for (int l = 1, r = mid; r <= len[i]; ++l, ++r) st[i].insert(get_hash(i,l,r));
    }

    for (ULL i : st[1]) {
        bool flag = true;
        for (int j = 2; j <= n; ++j) {
            if (!st[j].count(i)) flag = false;
        }
        if (flag) return true;
    }
    return false;
}

int main()
{
    p[0] = 1;
    for (int i = 1; i < M; ++i) p[i] = p[i-1] * P;

    int n, l = 1, r = 1e9; cin >> n;
    for (int i = 1; i <= n; ++i) {
        cin >> (s[i]+1);
        len[i] = strlen(s[i]+1); r = min(r, len[i]);

        h[i][0] = 1;
    }
}
```

```
    for (int j = 1; j <= len[i]; ++j) h[i][j] = h[i][j-1]*P + s[i][j];
}

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