

递归全排列

人员

柳力玮、韩昱辰、温郝冬、田心一、高健桓、蒋叔璋、刘宸熙、赵书梵、苑钊、李瑞涵 到课, 初锦阳 线上

上周作业检查

上周作业链接: <https://cppoj.kids123code.com/contest/1343>

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

王向东老师周日三点半C++递归入门

刷新

#	用户名	姓名	编程分	时间	A	B	C	D	E	F	G
1	liuliwei	柳力玮	560	1171	100	100	100	100	100	20	40
2	hanyuchen	韩昱辰	560	1174	100	100	100	100	100	20	40
3	gaojianhuan	高健桓	560	1175	100	100	100	100	100	20	40
4	tianxinyi	田心一	560	1177	100	100	100	100	100	20	40
5	wenhaodong	温郝冬	505	45	100	100	100	100	85	20	
6	yuanzhao	苑钊	500	1173	100	100	100	100	100		
7	liuchenxi	刘宸熙	500	1174	100	100	100	100	100		
8	jiangshuzhang	蒋叔璋	500	1176	100	100	100	100	100		
9	liruihan	李瑞涵	500	1177	100	100	100	100	100		
10	jibohan	纪博涵	485	44	100	100	100	100	85		
11	zhaoshufan	赵书梵	400	46	100	100	100	100			
12	chujinyang	初锦阳	385	35	100	100	100		85		

本周作业

<https://cppoj.kids123code.com/contest/1480> (课上讲了 A ~ E 题, 课后作业是 F 题必做, G 题选做)

课堂表现

今天的第一题稍微复杂一些, 同学们课上第一遍做都普遍做的不好, 课下需要再好好做一做这个题。

课堂内容

Coverage

考虑每个集合选或不选的所有情况, 最后把选的集合挑出来判断是否符合题目要求。

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

using namespace std;

const int maxn = 10 + 5;
set<int> s[maxn];
int n, m, res = 0;
bool st[maxn];
```

```
void dfs(int u) {
    if (u == m+1) {
        // 判断选出来的集合，能否凑出 1~n
        set<int> bingji;
        for (int i = 1; i <= m; i++) {
            if (st[i]) { // 选第 i 个集合，把第 i 个集合里的数放到并集里
                for (int j : s[i]) bingji.insert(j);
            }
        }
        bool flag = true;
        for (int i = 1; i <= n; i++) { // 判断并集里面是否 1~n 都有
            if (!bingji.count(i)) flag = false;
        }
        if (flag) res++; // 如果 1~n 都有，就 res++
        return;
    }

    st[u] = false;
    dfs(u+1);
    st[u] = true;
    dfs(u+1);
}

int main()
{
    cin >> n >> m;
    for (int i = 1; i <= m; i++) {
        int len; cin >> len;
        for (int j = 1; j <= len; j++) {
            int x; cin >> x;
            s[i].insert(x);
        }
    }
    dfs(1);
    cout << res << endl;
    return 0;
}
```

全排列问题

从第 1 层递归处理到第 n 层，每层考虑填 1~n 之间的数，填过的数后面不能用了，所以要打上标记，递归结束后记得把标记解除

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

using namespace std;

int a[15];
bool st[15];
int n;
```

```
void dfs(int u) {
    if (u == n+1) {
        for (int i = 1; i <= n; i++) printf("%5d", a[i]);
        cout << endl;
        return;
    }
    for (int i = 1; i <= n; i++) {
        if (st[i]) continue;
        a[u] = i, st[i] = true;
        dfs(u+1);
        st[i] = false;
    }
}

int main()
{
    cin >> n;
    dfs(1);
    return 0;
}
```

枚举排列（递归实现排列型枚举）

跟上个题目基本一致, 只是不再是 n 层了, 递归到第 k 层截止

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

using namespace std;

int a[15];
bool st[15];
int n, k;

void dfs(int u) {
    if (u == k+1) {
        for (int i = 1; i <= k; i++) printf("%d ", a[i]);
        cout << endl;
        return;
    }
    for (int i = 1; i <= n; i++) {
        if (st[i]) continue;
        a[u] = i, st[i] = true;
        dfs(u+1);
        st[i] = false;
    }
}

int main()
{
    cin >> n >> k;
```

```
    dfs(1);  
    return 0;  
}
```

增进感情

2^n 枚举所有状态, 考虑每件事情做或者不做, 最后维护一个最小的差值

```
#include <bits/stdc++.h>  
  
using namespace std;  
  
const int maxn = 30 + 5;  
int a[maxn], b[maxn];  
int n, v, res = 1e9;  
  
void dfs(int u, int sum1, int sum2) {  
    if (u == n+1) {  
        if (sum1+sum2 >= v) {  
            res = min(res, abs(sum1-sum2));  
        }  
        return;  
    }  
    dfs(u+1, sum1, sum2);  
    dfs(u+1, sum1+a[u], sum2+b[u]);  
}  
  
int main()  
{  
    cin >> n >> v;  
    for (int i = 1; i <= n; i++) cin >> a[i] >> b[i];  
    dfs(1, 0, 0);  
    if (res == 1e9) cout << -1 << endl;  
    else cout << res << endl;  
    return 0;  
}
```

回形方阵

先递归构建图形, 最后输出

```
#include <bits/stdc++.h>  
  
using namespace std;  
  
const int maxn = 200 + 5;  
int w[maxn][maxn];  
  
void dfs(int x1, int y1, int x2, int y2, int v) {
```

```
if (v == 0) {
    w[x1][y1] = v; return;
}

for (int i = x1; i <= x2; ++i) w[i][y1] = w[i][y2] = v;
for (int i = y1; i <= y2; ++i) w[x1][i] = w[x2][i] = v;
dfs(x1+1,y1+1,x2-1,y2-1,v-1);
}

int main()
{
    int n; cin >> n;
    dfs(1,1,2*n+1,2*n+1,n);
    for (int i = 1; i <= 2*n+1; ++i) {
        for (int j = 1; j <= 2*n+1; ++j) {
            printf("%3d", w[i][j]);
        }
        cout << endl;
    }
    return 0;
}
```