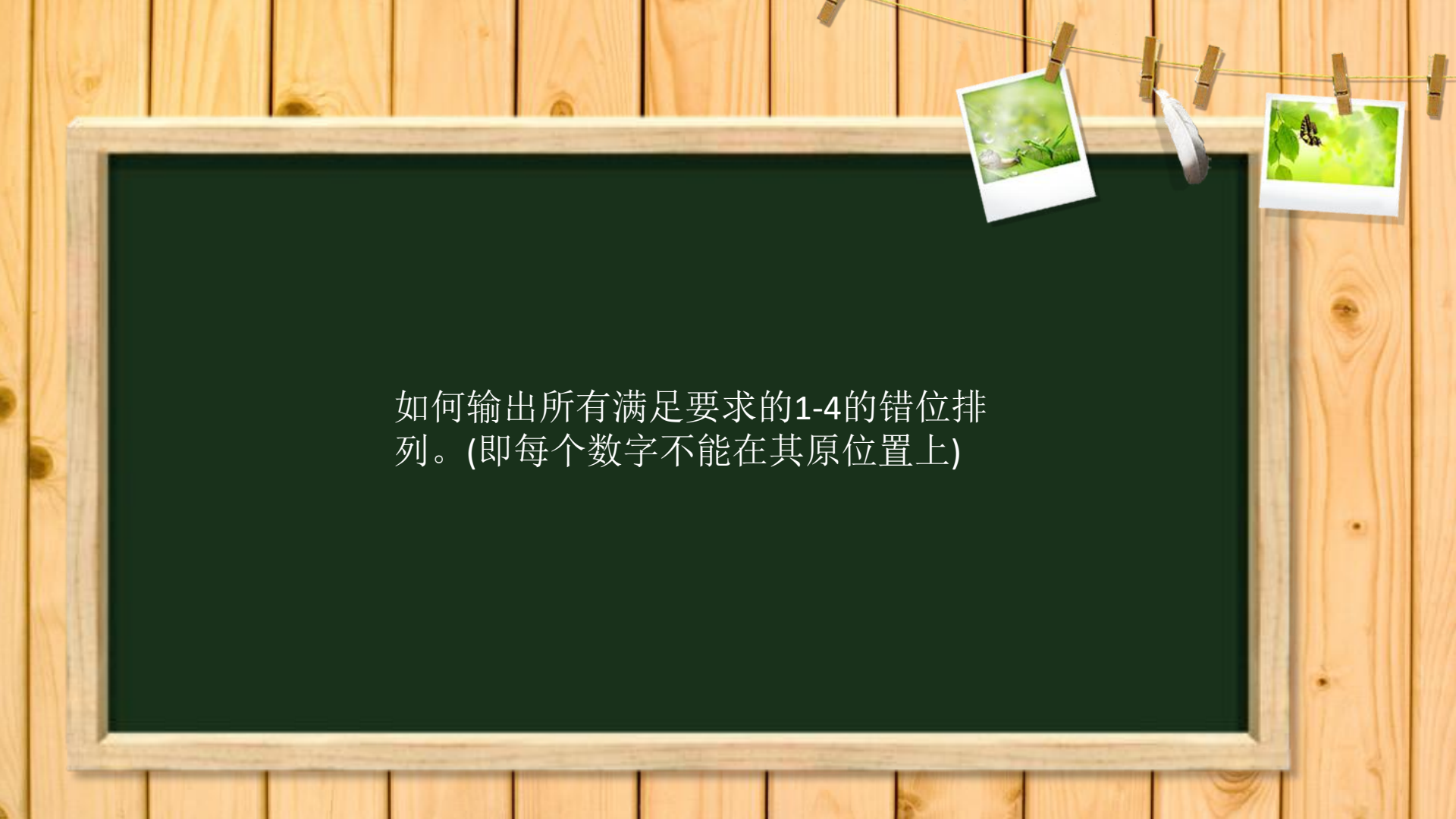


2016暑期算法集训

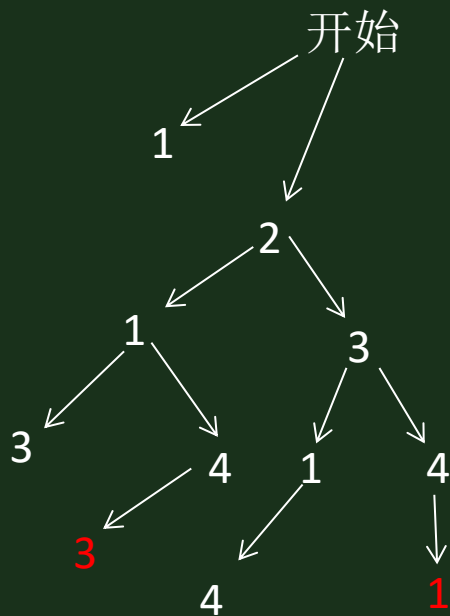
第3日

搜索算法1

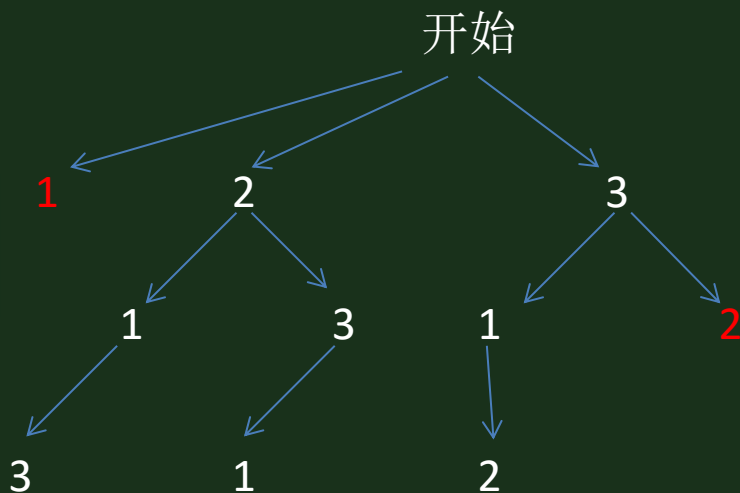




如何输出所有满足要求的1-4的错位排列。(即每个数字不能在其原位置上)



```
1#include<iostream>
2using namespace std;
3int n,deep=0,arr[105]={0},vis[105]={0},nex[105]={0};
4int main()
5{
6    cin>>n;
7    while (deep!=-1)
8    {
9        if (deep==n)
10       {
11           for (int i=1;i<=deep;i++)
12               cout<<arr[i]<<' ';
13           cout<<endl;
14           vis[arr[deep]]=0,deep--;
15           continue;
16       }
17       nex[deep]++;
18       if (nex[deep]==n+1)
19       {
20           vis[arr[deep]]=0,deep--;
21           continue;
22       }
23       if (vis[nex[deep]]==0 && deep+1!=nex[deep])
24       {
25           nex[deep+1]=0,vis[nex[deep]]=1;
26           arr[deep+1]=nex[deep],deep++;
27       }
28     }
29 }
```



```
1#include<iostream>
2#include<queue>
3using namespace std;
4queue<string> q[105];
5int n;
6int main()
7{
8    cin>>n;
9    q[0].push("");
10   for (int i=0;i<n;i++)
11   {
12       while (!q[i].empty())
13       {
14           string now=q[i].front();q[i].pop();
15           for (char c='1';c<='0'+n;c++)
16           {
17               int flag=0;
18               for (int j=0;j<now.size();j++)
19               if (now[j]==c)
20                   flag=1;
21               if (flag==1 || i+1==c-'0')
22                   continue;
23               q[i+1].push(now+c);
24           }
25       }
26   }
27   while (!q[n].empty()) {cout<<q[n].front()<<
28}
```


不要鄙视我，
我也觉得土。



深度优先搜索

深度优先搜索常用写法

```
1 void dfs(当前状态)
2 {
3     if (搜到结果)
4     {
5         更新答案 or 输出答案;
6         return;
7     }
8     for (当前状态的每一个可能性)
9     {
10        算出下一个状态;
11        if (下一个状态是合法的)
12            dfs(下一个状态);
13    }
14    return;
15 }
```

错位排列问题

```
1void dfs(string now) //当前状态
2{
3    if (now.size()==n) //搜到结果
4    {
5        cout<<now<<endl; //更新答案 or 输出答案;
6        return;
7    }
8    for (char c='1';c<='0'+n;c++) //枚举当前状态的每一个可能性
9    {
10        string nex=now+c; //算出下一个状态;
11        int flag=0; //判断下一个状态是否合法
12        for (int i=0;i<nex.size()-1;i++)
13            if (nex[i]==nex[nex.size()-1])
14                flag=1;
15        if (nex.size()==nex[nex.size()-1])
16            flag=1;
17        if (flag==0) //下一个状态是合法的
18            dfs(nex);
19    }
20    return;
21}
```


深度优先搜索的时间复杂度

深搜时间复杂度=状态数 \times 求得新状态的代价

求得新状态的代价=扩展的可能性 \times 得到新状态的复杂度

例如错位排序:

状态数:约 $(n-1)!$

扩展的可能性: n `for (char c='1';c<='0'+n;c++)`

得到新状态的复杂度: n

```
string nex=now+c; //算出下一个状态;
int flag=0; //判断下一个状态是否合法
for (int i=0;i<nex.size()-1;i++)
    if (nex[i]==nex[nex.size()-1])
        flag=1;
```

错位排列的若干优化

状态数:每种情况都是答案，很难优化。

扩展的可能性:利用链表进行枚举，一旦某个数字被用了，直接从链表中删掉它，这样每次只需要枚举没用过的点即可。

得到新状态的复杂度:从记录 n 位改为记录 1 位，利用`vis`数组优化判断是否合法的速度。

本题中实际上链表优化和`vis`数组都能起到这样的效果，任取其一即可。

学习深搜有什么用？

1.加深对递归的理解（后续treedp，高级数据结构都用到递归）

2.解搜索题。

3.学习图论必须掌握的知识之一。

4.打表（神器）

5.数学题小数据找规律（神器）

6.乱搞骗过某些神题（神器）



谈起乱搞，同志们需要掌握更高级的技巧

- 1.卡时（很重要）
- 2.随机化（重要）
- 3.选择搜索顺序（很重要）
- 4.剪枝（很重要）
- 5.IDA*（很重要）
- 6.大胆去写（非常重要）
- 7.代码能力超群（非常非常非常重要）



给一个数独局面，怎么解？



给你一个连连看的局面，可以把所有的格子都消除吗？能输出YES，不能输出NO。



给一个 $n*m$ 的棋盘，现在有 k 种颜色的涂料，每种涂料可以使用 a_i 次。求能否找出一种方案，给所有的格子染色，保证相邻的格子之间颜色不同。如果存在给出任意一种解决方案；否则输出NO
 $n, m \leq 5, k \leq 25, \sum a_i = n*m$



给定 n 个人， m 对朋友关系，朋友关系可以选择online聊天，也可以选择offline，对于每个人，他选择的online和offline的朋友相同（比如拥有 x 个online，就得拥有 x 个offline），问全部人满足有多少种情况？只要一个选择不同，视为不同情况。



共有 n 个物品，每个物品的重量是 w_i ，价格是 c_i ，问：恰好用 K 元，最多能买到多重的物品。

$n \leq 30, c_i, w_i \leq 10^9$



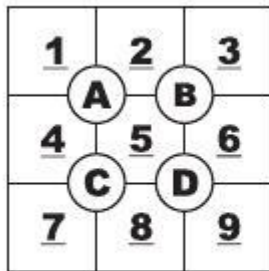


Figure (a)

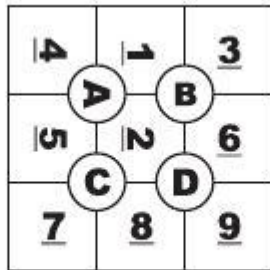


Figure (b)

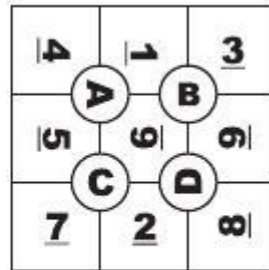


Figure (c)

上述游戏，最少几步达到目标状态





上
正
餐
咯

$n*m$ 的格子中，最多能放多少个棋子，使得他们不出现 k 子连？

$n, m, k \leq 7$



有 N 个糖果， M 个小孩 ($1 \leq N$, $M \leq 13$), 给定一个like矩阵, $\text{like}[i][j]=1$ 表示第 i 个小孩喜欢第 j 个糖果, $\text{like}[i][j]=0$ 表示第 i 个小孩不喜欢第 j 个糖果。如果把第 j 个糖果给第 i 个小孩吃, 他喜欢, 则得到 K (K 不变且 $K \leq 10$) 的欢喜值, 不喜欢则只得到1的价值, 问: 是否能够合理分配糖果, 使得每个小孩的欢喜值超过 $B[i]$ 。



谢谢大家！

下节课再见！

