

# 高阶程序设计第六次报告

姓名：陈德创

学号：19030500217

## 高阶程序设计第六次报告

总览

P1219、八皇后

P1596、Lake Counting S

P2404、自然数的拆分问题

P3915、树的分解

题目复述

解题思路

代码

复杂度分析

提交结果

总结

## 总览


✓	P1219	[USACO1.5]八皇后 Checker Challenge	搜索 深度优先搜索,DFS	普及/提高-	<div></div>
✓	P2392	kkksc03考前临时抱佛脚	搜索 贪心 递归 背包	普及-	<div></div>
✓	P1443	马的遍历	搜索 广度优先搜索,BFS 队列	普及/提高-	<div></div>
✓	P1135	奇怪的电梯	模拟 广度优先搜索,BFS 深度优先搜索,DFS	普及/提高-	<div></div>
—	P2895	[USACO08FEB]Meteor Shower S	搜索 广度优先搜索,BFS	普及/提高-	<div></div>
✓	P1036	选数	搜索 深度优先搜索,DFS 素数判断,质数,筛法	普及-	<div></div>
—	P2036	[COCI2008-2009#2] PERKET	模拟 搜索	入门	<div></div>
✓	P1433	吃奶酪	动态规划,动规,dp 状态压缩,状压	普及+/提高	<div></div>
✓	P1605	迷宫	搜索 逆推 枚举,暴力	普及-	<div></div>
✓	P1019	单词接龙	字符串 搜索	普及/提高-	<div></div>
—	P1101	单词方阵	字符串 搜索	普及-	<div></div>
✓	P2404	自然数的拆分问题	搜索	普及-	<div></div>
✓	P1596	[USACO10OCT]Lake Counting S	搜索 深度优先搜索,DFS	普及-	<div></div>

## P1219、八皇后

 LetMyself 2019-11-14 15:44	Accepted 100	P1219 [USACO1.5]八皇后 Checker Challenge	⌚ 554ms / 📄 776.00KB / 🗂 553B C++
 LetMyself 2019-11-14 15:36	Accepted 100	P1219 [USACO1.5]八皇后 Checker Challenge	⌚ 554ms / 📄 804.00KB / 🗂 553B C++

dfs入门永远的经典问题

## P1596、Lake Counting S

 LetMyself 06-01 21:37:40	Accepted 100	P1596 [USACO10OCT]Lake Counting S	⌚ 63ms / 📄 1.23MB / 🗂 832B C++
---	-----------------	-----------------------------------	--------------------------------

dfs染色，其实用bfs也行，只要能遍历都行。

## P2404、自然数的拆分问题

 LetMyself 06-01 21:51:53	Accepted 100	P2404 自然数的拆分问题	⌚ 19ms / 📄 672.00KB / 🗂 563B C++
---	-----------------	----------------	----------------------------------

给定 $n$ ，求出 $n$ 的拆分成一些正整数的和。也是dfs基础题。

## P3915、树的分解

### 题目复述

给出 $N$ 个点的树和 $K$ ，问能否把树划分成 $\frac{N}{K}$ 个连通块，且每个连通块的点数都是 $K$ 。

### 解题思路

我们要将树化为若干连通块，那么可以想见，如果这是可行的，那么一定有一些连通块是这棵树的子树，这些子树的节点数为 $k$ 。如果我们删去这些是子树的连通块，那么就会有另一些连通块是子树。如此往复，最终应当可以把整个树给删除。

下面我们考虑“删除”这个操作。

我们设 $f[i]$ 为以第 $i$ 号结点为根节点的子树的个数。显然我们有 $f[i] = 1 + \sum_{t, t \text{ 是 } i \text{ 的子节点}} f[t]$ , 遍历过程可以通过 $dfs$ 得到。

那么在 $dfs$ 过程中, 如果某一个节点的 $f[i] == k$ , 那么我们令 $f[i] = 0$ , 即“删除”了这棵子树。

### 一个小优化。

注意到每棵树只会遍历一次, 那么对于每个 $i$ ,  $f[i]$ 只会被用到一次。那么对于 $dfs(x)$ , 我们可以令其返回值为所求得的子树节点数。省下一个数组。

### !!! 注意

这是一颗无向图, 我们建图的时候可以双向建边, 然后从1号节点开始 $dfs$ , 在这时候我们要注意不要让一个子节点再去遍历其根节点, 否则导致死循环。有两个方法避免, 一个是 $vis[]$ 数组标记, 另一个是在 $dfs$ 函数中加入 $fa$ , 即 $dfs(int x, int fa)$ , 遍历节点的时候判断一下就可以了。

还有就是多组数据, 记得清空。

## 代码

```
1  #include<iostream>
2  #include<cstring>
3  using namespace std;
4  const int MAXN = 2e5 + 5; //最大值, 双向建边, 所以是2e5
5  struct EDGE{
6      int next, to;
7  }e[MAXN];
8  int head[MAXN], cnt;
9  int n, k;
10 // 前向星存图
11 void add_edge(int u, int v){
12     e[++cnt].next = head[u];
13     e[cnt].to = v;
14     head[u] = cnt;
15 }
16
17 int dfs(int x, int fa){
18     // sum = 1是因为要算自己
19     int sum = 1;
20     // 遍历子节点
21     for (int i = head[x]; i; i = e[i].next){
22         int to = e[i].to;
23         if (to == fa) continue; // 避免死循环
24         int t = dfs(to, x); // 得到以子节点为根的子树的节点数
25         sum += t;
26     }
27     // 如果正好满足条件, 就删去这棵树
28     if (sum == k)
29         return 0;
30     return sum;
31 }
32
33 void solve(){
34     // 每次清零
35     memset(head, 0, sizeof(head));
36     memset(e, 0, sizeof(e));
37     cnt = 0;
38     cin >> n >> k;
39
40     // 建图
```

```

41     for (int i = 1, u, v; i < n; i++){
42         cin >> u >> v;
43         add_edge(u, v);
44         add_edge(v, u);
45     }
46
47     // 小优化
48     if (n % k){
49         cout << "NO" << endl;
50         return;
51     }
52     // 开始dfs
53     int t = dfs(1, 0);
54     cout << (t == 0 ? "YES" : "NO") << endl;
55 }
56
57
58 int main(){
59     int t;
60     cin >> t;
61     while (t--){
62         solve();
63         return 0;
64     }

```

## 复杂度分析

单次 $dfs$ , 显然为 $O(Tn)$

## 提交结果

测试点信息

源代码

LetMyself

所属题目

P3915 树的分解

评测状态

Accepted

评测分数

100

提交时间

2020-06-02 22:04:12

测试点信息

#1 AC 6ms/2.81MB	#2 AC 8ms/2.79MB	#3 AC 6ms/2.80MB	#4 AC 9ms/2.87MB	#5 AC 5ms/3.03MB	#6 AC 11ms/2.84MB	#7 AC 124ms/2.79MB
#8 AC 95ms/2.90MB	#9 AC 159ms/2.91MB	#10 AC 268ms/2.90MB				

LetMyself 06-02 22:04:12	Accepted 100	P3915 树的分解	691ms / 3.03MB / 1.14KB C++
LetMyself 06-02 21:17:34	Accepted 100	P3915 树的分解	689ms / 2.91MB / 1.09KB C++
LetMyself 06-02 21:15:28	Unaccepted 0	P3915 树的分解	622ms / 1.87MB / 993B C++
LetMyself 06-02 21:04:24	Unaccepted 0	P3915 树的分解	643ms / 1.89MB / 1023B C++
LetMyself 06-02 21:02:32	Unaccepted 90	P3915 树的分解	727ms / 2.04MB / 1.00KB C++
LetMyself 06-02 20:55:18	Unaccepted 10	P3915 树的分解	1.58s / 125.00MB / 923B C++
LetMyself 06-02 20:52:15	Unaccepted 10	P3915 树的分解	608ms / 125.00MB / 909B C++

提交了这么多次，都是因为什么呢？哦~！原来是因为死循环，原来是因为单向边建图。QAQ

## 总结

*dfs*感觉用处还是挺多的，毕竟可以回溯。最近在练连通分量，*tarjan*算法的实现也是基于*dfs*。但是因为如果选连通分量的题的话*dfs*的部分就要略去了，所以就没写在报告里。因为我现在做的都是些连通分量入门题，套路明显，就是*tarjan*缩点成*DAG*，然后*dfs*再过一遍或者拓扑排序什么的。

*emmmmmm*说实话因为在学其他的东西，所以这次高阶没有很认真地去，选的题都是很了了的，没有太有难度的题。希望做*DP*的时候可以学点东西吧，而不是只挑自己会的做。