递归二叉树

人员

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

上周作业检查

上周作业链接: https://cppoj.kids123code.com/contest/1480

王向东老师周日三点半C++递归全排列

刷新

#	用户名	姓名	编程分	时间	A	B	C	D	E	F	G
1	hanyuchen	韩昱辰	700	259	100	100	100	100	100	100	100
2	liuliwei	柳力玮	700	1312	100	100	100	100	100	100	100
3	zhaoshufan	赵书梵	690	177	100	100	100	90	100	100	100
4	yuanzhao	苑钊	600	1156	100	100	100	100	100	100	
5	tianxinyi	田心一	600	1193	100	100	100	100	100	100	
6	liruihan	李瑞涵	600	1212	100	100	100	100	100	100	
7	wenhaodong	温郝冬	500	1048	100	100	100	100	100	0	
8	gaojianhuan	高健桓	500	1195	100	100	100	100	0	100	
9	jiangshuzhang	蒋叔璋	500	1199	100	100	100	100		100	
10	liuchenxi	刘宸熙	400	1142	100	100	100	100			
11	chujinyang	初锦阳	320	157	100	100	100	20	0		

您正在共享屏幕 结束共享

本周作业

https://cppoj.kids123code.com/contest/1589 (课上讲了 A ~ E 题, 课后作业是 F 题必做)

课堂表现

这节课的 A 题、D 题相对比较抽象一些, 同学们课下要好好看看这两道题

同学们上周的作业普遍都完成的比较好, 对同学们提出表扬!

课堂内容

烤鸡 (上周作业)

递归 10 层, 每层填 1~3 之间的数, 最后如果正好是 n, 说明符合题意, 进行记录

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

using namespace std;

const int maxn = 10 + 5;
```

```
int a[maxn];
int n, res = 0;;

void dfs(int u, int sum, bool flag) {
    if (u == 11) {
        if (sum == n) {
            if (flag == false) res++;
            else {
                for (int i = 1; i <= 10; i++) cout << a[i] << " ";
                cout << endl;
            }
        }
        return;
    }
    for (int i = 1; i <= 3; i++) {
        a[u] = i;
        dfs(u+1, sum+i, flag);
    }
}

int main()
{
    cin >> n;
    if (n<10 || n>30) cout << 0 << endl;
    else {
        dfs(1, 0, false);
        cout << res << endl;
        dfs(1, 0, true);
    }
    return 0;
}
```

赦免战俘

递归构建图形, 最后进行输出

分成 右上、左下、右下 3 部分进行递归

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

using namespace std;

const int maxn = 1024 + 5;
int a[maxn][maxn];

void dfs(int x1, int y1, int x2, int y2, int n) {
    if (n == 0) { a[x1][y1] = 1; return; }

    int x_ = (x1+x2)/2, y_ = (y1+y2)/2;

    dfs(x1,y_+1,x_,y2,n-1);
```

```
    dfs(x_+1,y1,x2,y_,n-1);
    dfs(x_+1,y_+1,x2,y2,n-1);
}

int main()
{
    int n; cin >> n;
    dfs(1,1,1<<n,1<<n,n);
    for (int i = 1; i <= (1<<n); i++) {
        for (int j = 1; j <= (1<<n); j++) cout << a[i][j] << " ";
        cout << endl;
    }
  return 0;
}
```

二叉树的遍历

建 二叉树 过程可以用二维数组来建, 然后递归输出二叉树的 前序、中序、后序 即可

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

using namespace std;

const int maxn = 1e6 + 5;
int tr[maxn][2];

void dfs_pre(int u) {
  cout << u << " ";
  for (int i = 0; i < 2; ++i) {
    if (tr[u][i]) dfs_pre(tr[u][i]);
  }
}

void dfs_mid(int u) {
  if (tr[u][0]) dfs_mid(tr[u][0]);
  cout << u << " ";
  if (tr[u][1]) dfs_mid(tr[u][1]);
}

void dfs_suf(int u) {
  for (int i = 0; i < 2; ++i) {
    if (tr[u][i]) dfs_suf(tr[u][i]);
  }
  cout << u << " ";
}

int main()
{
  int n; cin >> n;
  for (int i = 1; i <= n; ++i) cin >> tr[i][0] >> tr[i][1];
```

```
  dfs_pre(1); cout << endl;
  dfs_mid(1); cout << endl;
  dfs_suf(1); cout << endl;
  return 0;
}
```

【深基16.例3】二叉树深度

递归过程中记录每个点的深度, 最后求一个最大深度即可

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

using namespace std;

const int maxn = 1e6 + 5;
int tr[maxn][2], f[maxn];

void dfs(int u, int depth) {
  f[u] = depth;
  if (tr[u][0]) dfs(tr[u][0], depth+1);
  if (tr[u][1]) dfs(tr[u][1], depth+1);
}

int main()
{
  int n; cin >> n;
  for (int i = 1; i <= n; ++i) cin >> tr[i][0] >> tr[i][1];

  dfs(1, 1);
  int res = 0;
  for (int i = 1; i <= n; ++i) res = max(res, f[i]);
  cout << res << endl;
  return 0;
}
```

汉诺塔

经典 汉诺塔 问题, 分成 3 次递归来处理

```
#include <iostream>

using namespace std;

void f(int n, char a, char b, char c) {
    if (n == 1) {
        cout << a << " To " << c << endl;
        return;
    }
    f(n-1, a, c, b);  // n-1: a->b
```

```
    f(1, a, b, c);  // 1: a->c
    f(n-1, b, a, c);  // n-1: b->c
}

int main() {
    int n;
    cin >> n;
    f(n, 'A', 'B', 'C');
    return 0;
}
```

自然数的拆分问题

一层一层往下递归, 每一层的数至少要 >= 上一层的数, 要 <=n

递归过程中, 总和超过 n 立马 return, 等于 n 就输出并 return

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

using namespace std;

int n;
int w[15];

void dfs(int u, int sum) {
    if (sum > n) return;
    if (sum == n) {
        if (u == 2) return;
        cout << w[1];
        for (int i = 2; i <= u-1; i++) cout << "+" << w[i];
        cout << endl;
        return;
    }

    for (int i = w[u-1]; i <= n; i++) {
        w[u] = i;
        dfs(u+1, sum+i);
    }
}

int main()
{
    cin >> n;
    w[0] = 1;
    dfs(1, 0);
  return 0;
}
```