记忆化搜索

人员

洪晨栋、洪晨棋、郭栩睿、宋吉相 到课, 陶汇笙、罗启宸 线上

作业检查

上周作业链接: https://cppoj.kids123code.com/contest/129

2025-0628周六10:30 (递归存储输出二叉树)

刷新

#	用户名	姓名	总分	选择分	编程分	时间	A	B	C	D
1	hongchendong	洪晨栋	400	0	400	889	100	100	100	100
2	taohuisheng	陶汇笙	400	0	400	910	100	100	100	100
3	guoxurui	郭栩睿	300	0	300	844	100	100	100	0
4	hongchenqi	洪晨棋	300	0	300	871	100	100	100	

作业

https://cppoj.kids123code.com/contest/158 (课上讲了 A ~ D 这些题, 课后作业是 D E 题)

课堂表现

今天讲了 记忆化搜索 这个内容, 记忆化搜索 是有些难度的, 一开始不容易理解, 一些同学可能课上只理解了一部分, 没有完全理解, 课下要好好再复习复习课上的几道题。

课堂内容

新二叉树

把字符转整数, 然后用上节课的方法存储二叉树并前序遍历输出就可以了

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

using namespace std;

const int maxn = 200 + 5;
int tr[maxn][2];

int get_int(char x) {
    if (x == '*') return 0;
    return x;
}

void dfs1(int u) {
    if (u == 0) return;
    cout << char(u);
```

```
    dfs1(tr[u][0]);
    dfs1(tr[u][1]);
}

int main()
{
    int n; cin >> n;
    int root;
    for (int i = 1; i <= n; i++) {
        string s; cin >> s;
        int a = get_int(s[0]), b = get_int(s[1]), c = get_int(s[2]);
        tr[a][0] = b, tr[a][1] = c;
        if (i == 1) root = a;
    }
    dfs1(root);
    return 0;
}
```

【深基7.例7】计算阶乘

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

using namespace std;

int f(int n) {
  if (n == 1) return 1;
  return n * f(n-1);
}

int main()
{
  int n; cin >> n;
  cout << f(n) << endl;
  return 0;
}
```

斐波那契数列

记忆化搜索, 如果之前已经计算过斐波那契数列的第 n 项了, 就把第 n 项的值存在 f[n] 里, 后续不需要每次重新算了

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

using namespace std;

const int maxn = 30 + 5;
int f[maxn];
```

```
int fib(int n) {
  if (n==1 || n==2) return 1;
  if (f[n]) return f[n];
  f[n] = fib(n-1) + fib(n-2);
  return f[n];
}

int main()
{
  int T; cin >> T;
  while (T -- ) {
    int n; cin >> n;
    cout << fib(n) << endl;
  }
  return 0;
}
```

[IOI 1994] 数字三角形 Number Triangles

dfs(x,y): 求从点 (x,y) 到第 n 行的最大的值是多少

那么 dfs(x,y) 就是 a[x][y] 的值 加上 max(dfs(x+1,y), dfs(x+1,y+1))

dfs(x,y) 永远是固定的, 英雌可以记忆化

这样, 一共有 n^2 种状态, 总时间复杂度也是 n^2

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

using namespace std;

const int maxn = 1000 + 5;
int w[maxn][maxn], f[maxn][maxn];
int n;

int dfs(int x, int y) {
  if (x == n) return w[x][y];
  if (f[x][y] != -1) return f[x][y];
  f[x][y] = w[x][y] + max(dfs(x+1,y), dfs(x+1, y+1));
  return f[x][y];
}

int main()
{
  memset(f, -1, sizeof(f));
  cin >> n;
  for (int i = 1; i <= n; ++i) {
    for (int j = 1; j <= i; ++j) cin >> w[i][j];
  }
  cout << dfs(1, 1) << endl;
```

```
    return 0;
}
```

Function

按照题目的 4 个要求, 进行对应的 记忆化搜索 即可

除去 a<=0, b<=0, c<=0, a>20, b>20, c>20 的这些情况, 剩的情况只有 20 * 20 * 20 种情况

因此, 可以开一个 f[25][25][25] 的数组来进行记忆化, 确保每种状态只搜一遍

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

using namespace std;

typedef long long LL;
const int maxn = 20 + 5;
LL f[maxn][maxn][maxn];
bool st[maxn][maxn][maxn];

LL dfs(LL a, LL b, LL c) {
    if (a<=0 || b<=0 || c<=0) return 1;
    if (a>20 || b>20 || c>20) return dfs(20, 20, 20);
    if (st[a][b][c]) return f[a][b][c];
    st[a][b][c] = true;

    if (a<b && b<c) f[a][b][c] = dfs(a,b,c-1)+dfs(a,b-1,c-1)-dfs(a,b-1,c);
    else f[a][b][c] = dfs(a-1,b,c)+dfs(a-1,b-1,c)+dfs(a-1,b,c-1)-dfs(a-1,b-1,c-1);
    return f[a][b][c];
}

int main()
{
    LL a, b, c;
    while (true) {
        cin >> a >> b >> c;
        if (a==-1 && b==-1 && c==-1) break;
        printf("w(%lld, %lld, %lld) = %lld\n", a, b, c, dfs(a,b,c));
    }
    return 0;
}
```