递归

TEACHING COURSEWARE POWPOINT

授课时间：2025.08.01

目录

01

概念引入

TEACHING
COURSEWARE

TEACH

概念引入

找钥匙

大盒子中盒子小盒子钥匙

递 归

02

概念分析

TEACHING
COURSEWARE

TEACH

概念分析

递归的概念
递归（Recursion）是一种算法思想，指函数通过调用自身来解决问题的方式。其核心是将一个复杂问题分解为与原问题结构相同但规模更小的子问题，直到子问题小到可以直接解决。
递归必须满足两个要素：
基准情况：存在至少一种无需递归就能直接解决的情况（终止条件），
避免无限递归。

03

经典例题

TEACHING
COURSEWARE

TEACH

解题思路

题目描述

求 $n!$，也就是 $1 \times 2 \times 3 \cdots \times n$。

挑战：尝试不使用循环语句（for、while）完成这个任务。

输入格式

第一行输入一个正整数 n。

输出格式

输出一个正整数，表示 $n!$。

输入数据 1

```
3
```

Copy

输出数据 1

```
6
```

Copy

提示

数据保证，$1 \leq n \leq 12$。

解题思路

递归关系：

$n!=n\times(n-1)!$

标准程序


```cpp
#include <iostream>
using namespace std;

int f(int n) {
    if (n <= 1) {
        return 1;  // 基准情况：0!和1!都为1
    } else {
        int s = f(n - 1);  // 计算子问题n-1的阶乘
        return s * n;      // 当前n的阶乘 = 子问题结果 x n
    }
}
int main() {
    int n;
    cin >> n;
    cout << f(n) << endl;
    return 0;
}
```

解题思路

说明

树老师爬楼梯，他可以每次走1级或者2级，输入楼梯的级数，求不同的走法数。
例如：楼梯一共有3级，他可以每次都走一级，或者第一次走一级，第二次走两级，也可以第一次走两级，第二次走一级，一共3种方法。

输入格式

输入包含若干行，每行包含一个正整数N，代表楼梯级数，$1 \leq N \leq 30$。

输出格式

不同的走法数，每一行输入对应一行输出。

样例

输入数据 1

Copy

```
5
8
10
```

输出数据 1

Copy

```
8
34
89
```

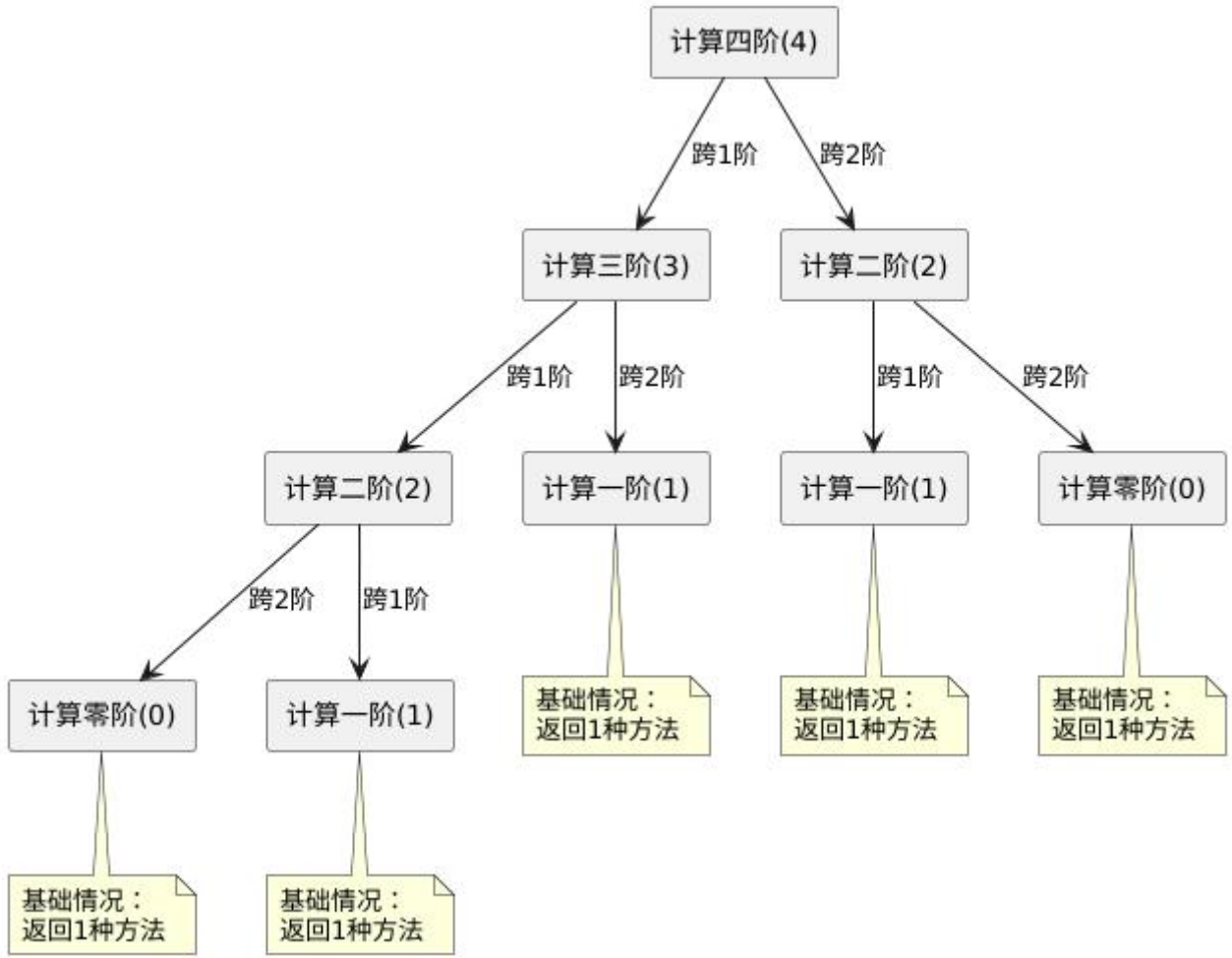
标准程序


```cpp
#include <iostream>
using namespace std;

// 递归函数：返回n级台阶的走法数
int climb(int n) {
    // 基准情况：n=1或n=2时直接返回结果
    if (n == 1) {
        return 1;
    } else if (n == 2) {
        return 2;
    }
    // 递归关系：f(n) = f(n-1) + f(n-2)
    return climb(n - 1) + climb(n - 2);
}

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

爬楼梯递归调用图（以4阶为例）



04

Q&A

TEACHING
COURSEWARE

TEACH

Q&A

Q:递归边界

Q:找钥匙的问题，递归边界是什么？