重庆南开中学

# 递归&递推&应用

南开中学信息学竞赛教练组

重庆南开中学

# 目录
contents


01 作业讲评

02 一道例题

03 又一道例题

04 ？？

PART 01

# 作业讲评

- 全都讲一讲!

重庆南开中学

# 作业3127 – 何老板爬楼梯

## 题目大意

共有n步楼梯，何老板每次可以跨1步、2步、3步或4步，计算恰好走完n步台阶的方案数。

数据范围：$n \le 30$。

## 解析

- 递归方法：
  - n=0, 1, 2, 3直接返回方案数1, 1, 2, 4
  - n⩾4时，递归计算
  - 用数组记录计算结果，避免多余的计算
  - 时间复杂度$O(n)$

```
int climb(int n) {
    static int f[MAX_N] = {1, 1, 2, 4};
    return f[n] ? f[n] : (f[n] = climb(n - 1) + climb(n - 2) + climb(n - 3) + climb(n - 4));
}
```

# 作业3127 - 何老板爬楼梯

重庆南开中学

## 题目大意

共有n步楼梯，何老板每次可以跨1步、2步、3步或4步，计算恰好走完n步台阶的方案数。

数据范围：$n \le 30$。

## 解析

- 递归方法：
  - n=0, 1, 2, 3直接返回方案数1, 1, 2, 4
  - n⩾4时，递归计算
  - 用数组记录计算结果，避免多余的计算
  - 时间复杂度$O(n)$

```
int climb(int n) {
    static int f[MAX_N] = {1, 1, 2, 4};
    return f[n] ? f[n] : (f[n] = climb(n - 1) + climb(n - 2) + climb(n - 3) + climb(n - 4));
}
```

- 递推方法
  - 设数组f[0..n]中的f[i]表示走完i步的方案数
  - f[0]=1, f[1]=1, f[2]=2, f[3]=4
  - i⩾4时f[i]=f[i-1]+f[i-2]+f[i-3]+f[i-4]
  - 时间复杂度$O(n)$

```
static int f[MAX_N] = {1, 1, 2, 4};
for (int i = 4; i <= n; i ++) {
    f[i] = f[i - 1] + f[i - 2] + f[i - 3] + f[i - 4];
}
```

重庆南开中学

# 作业3131 - 自然数拆分

## 题目大意

把n拆分成若干个自然数的和，拆分相同顺序不同视为一种方案，有多少种方案？

数据范围：$n \le 35$。

## 解析

- 递归方法：
  - 函数decom(n, m)计算把n拆分成若干个不超过m的数的方案数；

$$decom(n,m)=\begin{cases}1 & n\le 1 \text{ or } m=1\\ decom(n,n) & n<m\\ decom(n-m,m)+decom(n,m-1) & \text{other}\end{cases}$$

  - 最终答案是decom(n, n)-1
  - 用数组避免多余的计算
  - 时间复杂度$O(n^2)$

```
int decom(int n, int m) {
    static int f[MAX_N][MAX_N];
    if (n <= 1 || m == 1) {
        return 1;
    } else if (f[n][m]) {
        return f[n][m];
    } else if (n < m) {
        return f[n][m] = decom(n, n);
    } else {
        return f[n][m] = decom(n - m, m) + decom(n, m - 1);
    }
}
```

# 作业1334 - 分解自然数

## 题目大意

把n拆分成若干个自然数的和，拆分相同顺序不同视为一种方案，有多少种方案？

数据范围：$n \le 5000$。

## 解析

- 递推方法：
  - 数组f[0..n][0..n]的f[i][j]是把j拆分成不超过i的数的方案数；
  - f[0][0]=1，f[0][j]=0
  - 递推公式：
    - f[i][j]=f[i-1][j]　　　　(1⩽i⩽n，0⩽j⩽i-1)
    - f[i][j]=f[i][j-i]+f[i-1][j]　(1⩽i⩽n，i⩽j⩽n)
  - 最终答案是f[n][n]-1
  - 好像没啥区别？

# 作业1334 - 分解自然数

## 题目大意

把n拆分成若干个自然数的和，拆分相同顺序不同视为一种方案，有多少种方案？

数据范围：$n \le 5000$。

## 解析

- 递推方法：
  - 数组f[0..n][0..n]的f[i][j]是把j拆分成不超过i的数的方案数；
  - f[0][0]=1，f[0][j]=0
  - 递推公式：
    - f[i][j]=f[i-1][j]　　　　　　(1⩽i⩽n，0⩽j⩽i-1)
    - f[i][j]=f[i][j-i]+f[i-1][j]　(1⩽i⩽n，i⩽j⩽n)
  - 最终答案是f[n][n]-1
  - 好像没啥区别？

```c
int main() {
    scanf("%d", &N);
    f[0][0] = 1;
    for (int i = 1; i <= N; i ++) {
        for (int j = 0; j < i; j ++) {
            f[i][j] = f[i - 1][j];
        }
        for (int j = i; j <= N; j ++) {
            int t = f[i][j - i] + f[i - 1][j];
            f[i][j] = t < P ? t : t - P;
        }
    }
    printf("%d\n", (f[N][N] - 1 + P) % P);
    return 0;
}
```

# 作业1334 – 分解自然数



## 题目大意

把n拆分成若干个自然数的和，拆分相同顺序不同视为一种方案，有多少种方案？

数据范围：$n \le 5000$。

## 解析

- 递推方法●优化：
  - 观察递推公式：
    - f[i][j]=f[i-1][j]　　　　　　(1⩽i⩽n，0⩽j⩽i-1)
    - f[i][j]=f[i][j-i]+f[i-1][j] (1⩽i⩽n，i⩽j⩽n)

# 作业1334 - 分解自然数

重庆南开中学

## 题目大意

把n拆分成若干个自然数的和，拆分相同顺序不同视为一种方案，有多少种方案？

数据范围：$n \leq 5000$。

## 解析

- 递推方法●优化：
  - 观察递推公式：
    - f[i][j]=f[i-1][j]　　　　　　(1⩽i⩽n, 0⩽j⩽i-1)
    - f[i][j]=f[i][j-i]+f[i-1][j] (1⩽i⩽n, i⩽j⩽n)
  - 如果外层for循环从小到大枚举i，任意时刻同时使用的数组只有f[i][]和f[i-1][]两行！

重庆南开中学

# 作业1334 - 分解自然数

## 题目大意

把n拆分成若干个自然数的和，拆分相同顺序不同视为一种方案，有多少种方案？

数据范围：$n \le 5000$。

## 解析

- 递推方法●优化：
  - 观察递推公式：
    - f[i][j]=f[i-1][j]　　　　　　(1⩽i⩽n, 0⩽j⩽i-1)
    - f[i][j]=f[i][j-i]+f[i-1][j] (1⩽i⩽n, i⩽j⩽n)
  - 如果外层for循环从小到大枚举i，任意时刻同时使用的数组只有f[i][]和f[i-1][]两行！
  - 而且同时使用的是f[i][]的前半部分和f[i-1][]的后半部分，如果内层for循环从小到大枚举j就只需要一维数组！！

# 作业1334 - 分解自然数

重庆南开中学

## 题目大意

把n拆分成若干个自然数的和，拆分相同顺序不同视为一种方案，有多少种方案？

数据范围：$n \leq 5000$。

## 解析

- 递推方法●优化：
  - 观察递推公式：
    - f[i][j]=f[i-1][j]　　　　(1≤i≤n, 0≤j≤i-1)
    - f[i][j]=f[i][j-i]+f[i-1][j] (1≤i≤n, i≤j≤n)
  - 如果外层for循环从小到大枚举i，任意时刻同时使用的数组只有f[i][]和f[i-1][]两行！
  - 而且同时使用的是f[i][]的前半部分和f[i-1][]的后半部分，如果内层for循环从小到大枚举j就只需要一维数组！！
  - 时间复杂度$O(n^2)$，空间复杂度$O(n)$。

```
int main() {
    scanf("%d", &N);
    f[0] = 1;
    for (int i = 1; i <= N; i ++) {
        for (int j = i; j <= N; j ++) {
            int t = f[j - i] + f[j];
            f[j] = t < P ? t : t - P;
        }
    }
    printf("%d\n", (f[N] - 1 + P) % P);
    return 0;
}
```

# 作业1334 - 分解自然数

## 题目大意

把n拆分成若干个自然数的和，拆分相同顺序不同视为一种方案，有多少种方案？

数据范围：$n \leq 5000$。

## 解析

- 递推方法2：
  - 数组f[0..n][0..n]的f[i][j]表示把i拆分成不超过j个数的方案数；

# 作业1334 - 分解自然数

## 题目大意

把n拆分成若干个自然数的和，拆分相同顺序不同视为一种方案，有多少种方案？

数据范围：$n \le 5000$。

## 解析

- 递推方法2:
  - 数组f[0..n][0..n]的f[i][j]表示把i拆分成不超过j个数的方案数；
  - f[i][j]的方案分成两类:
    - 拆分成不超过j-1个数：f[i][j-1]种
    - 拆分成恰好j个数：假设将j个数每个都减少1，然后舍弃0。

      剩下的数加起来等于i-j，不超过j个。

      恰好对应f[i-j][j]的所有方案！

# 作业1334 - 分解自然数

重庆南开中学

## 题目大意

把n拆分成若干个自然数的和，拆分相同顺序不同视为一种方案，有多少种方案？

数据范围：$n \le 5000$。

## 解析

- 递推方法2:
  - 数组f[0..n][0..n]的f[i][j]表示把i拆分成不超过j个数的方案数；
  - f[i][j]的方案分成两类：
    - 拆分成不超过j-1个数：f[i][j-1]种
    - 拆分成恰好j个数：假设将j个数每个都减少1，然后舍弃0。
      剩下的数加起来等于i-j，不超过j个。
      恰好对应f[i-j][j]的所有方案！
  - 递推公式：f[i][j]=f[i][j-1]+f[i-j][j]
  - 看递推公式发现，实际上只是把递推方法1的i和j交换了一下，但换个角度思考问题非常重要。

# 作业5119 – Ackermann函数

## 题目大意

Ackermann函数如图所示，输入m, n，计算A(m, n)。

数据范围：$m \le 3, n \le 10$。

$$A(m,n)=\begin{cases} n+1 & m=0 \\ A(m-1,1) & m>0, n=0 \\ A(m-1, A(m,n-1)) & m>0, n>0 \end{cases}$$

## 解析

- 递归计算：按照题目要求直接计算即可。
- 画蛇添足：数组记录避免重复计算？
  - 计算过程中的n会远远超过输入时的范围n≤10
  - 事实上，根本就不会有重复的计算。

重庆南开中学

# 作业5119 - Ackermann函数

## 题目大意

Ackermann函数如图所示，输入m, n，计算A(m, n)。

$$A(m,n)=\begin{cases} n+1 & m=0 \\ A(m-1,1) & m>0, n=0 \\ A(m-1, A(m,n-1)) & m>0, n>0 \end{cases}$$

数据范围：$m \le 3, n \le 10$。

## 解析

- 递归计算：按照题目要求直接计算即可。
- 画蛇添足：数组记录避免重复计算？
  - 计算过程中的n会远远超过输入时的范围n≤10
  - 事实上，根本就不会有重复的计算。
- Ackermann函数的一些基本规律：
  - $m \ge 4$后，增长速度极快

$$\begin{aligned} Ackermann(0,n) &= n+1 \\ Ackermann(1,n) &= n+2 \\ Ackermann(2,n) &= 2n+3 \\ Ackermann(3,n) &= 2^{n+3}-3 \\ Ackermann(4,n) &= \underbrace{2^{2^{\cdot^{\cdot^{2}}}}}_{n+3\text{个}2}-3 \end{aligned}$$

PART 02

# 一道例题

- ? ?

重庆南开中学

# 例题

## 例题【NKOJ3584 集合划分】

n个元素的集合{1, 2, •••, n}可以划分为若干个非空子集。

例如，当n=3时，集合{1, 2, 3}可以有5个不同的非空子集划分方案：

- 3个子集有1个方案：{{1}, {2}, {3}}
- 2个子集有3个方案：{{1, 2}, {3}}、{{1, 3}, {2}}、{{1}, {2, 3}}
- 1个子集有1个方案：{{1, 2, 3}}

给定正整数n，计算出n个元素的集合{1, 2, ..., n}有多少个不同的非空子集划分方案。

数据范围：$n \leq 25$

样例输入：
3

样例输出：
5

样例输入：
4

样例输出：
15

# 例题

## 例题【NKOJ3584 集合划分】

## 解析

- 递归方法1
- 函数$F(n)$计算把{1, 2, . . . , n}划分成若干个子集的方案数
- 计算$F(n)$时，考虑n所在的集合有多少个数：
  - 若n所在集合只有n这1个数，剩下的数有$F(n-1)$种方案；

# 例题

重庆南开中学

例题【NKOJ3584 集合划分】

解析

- 递归方法1
- 函数$F(n)$计算把$\{1, 2, \ldots, n\}$划分成若干个子集的方案数
- 计算$F(n)$时，考虑n所在的集合有多少个数：
  - 若n所在集合只有n这1个数，剩下的数有$F(n-1)$种方案；
  - 若n所在集合有2个数$\{n, a\}$，剩下的数有$F(n-2)$种方案，
    $1 \sim n-1$每个数都可以作为a，所以共有$(n-1) \times F(n-2)$种方案；

# 例题

## 例题【NKOJ3584 集合划分】

## 解析

- 递归方法1
- 函数$F(n)$计算把$\{1, 2, ..., n\}$划分成若干个子集的方案数
- 计算$F(n)$时，考虑n所在的集合有多少个数：
  - 若n所在集合只有n这1个数，剩下的数有$F(n-1)$种方案；
  - 若n所在集合有2个数$\{n, a\}$，剩下的数有$F(n-2)$种方案，
    $1\sim n-1$每个数都可以作为a，所以共有$(n-1)\times F(n-2)$种方案；
  - 若n所在集合有3个数$\{n, a, b\}$，剩下的数有$F(n-3)$种方案，
    $1\sim n-1$任选2个数作为a, b，共有$\frac{(n-1)(n-2)}{2}F(n-3)$种方案；

# 例题

重庆南开中学

## 例题【NKOJ3584 集合划分】

## 解析

- 递归方法1
- 函数$F(n)$计算把$\{1, 2, ..., n\}$划分成若干个子集的方案数
- 计算$F(n)$时，考虑n所在的集合有多少个数：
  - 若n所在集合只有n这1个数，剩下的数有$F(n-1)$种方案；
  - 若n所在集合有2个数$\{n, a\}$，剩下的数有$F(n-2)$种方案，
    $1 \sim n-1$每个数都可以作为a，所以共有$(n-1) \times F(n-2)$种方案；
  - 若n所在集合有3个数$\{n, a, b\}$，剩下的数有$F(n-3)$种方案，
    $1 \sim n-1$任选2个数作为a, b，共有$\frac{(n-1)(n-2)}{2}F(n-3)$种方案；
  - 若n所在集合有4个数$\{n, a, b, c\}$，共有$\frac{(n-1)(n-2)(n-3)}{6}F(n-4)$种方案；
  - ………….

重庆南开中学

# 例题

例题【NKOJ3584 集合划分】

解析

- 递归方法1
- 计算$F(n)$时，考虑n所在的集合有多少个数：
  - 若n所在集合有$k+1$个数，共有$\binom{n-1}{k}F(n-1-k)$种方案

    其中$0 \le k \le n-1$。

# 例题

## 例题【NKOJ3584 集合划分】

## 解析

- 递归方法1
- 计算$F(n)$时，考虑n所在的集合有多少个数：
  - 若n所在集合有$k+1$个数，共有$\binom{n-1}{k}F(n-1-k)$种方案

    其中$0 \le k \le n-1$。
- 边界：$F(0)=F(1)=1$。

# 例题

## 例题【NKOJ3584 集合划分】

## 解析

- 递归方法1
- 计算$F(n)$时，考虑n所在的集合有多少个数：
    - 若n所在集合有$k+1$个数，共有$\binom{n-1}{k}F(n-1-k)$种方案

      其中$0 \le k \le n-1$。
- 边界：$F(0)=F(1)=1$。
- 时间复杂度$O(n^2)$，空间复杂度$O(n)$。

```
int F(int n) {
    static int f[MAX_N] = {1, 1};
    if (!f[n]) {
        for (int k = 0, c = 1; k <= n - 1; k ++) {
            f[n] += c * F(n - 1 - k);
            c = c * (n - 1 - k) / (k + 1);
        }
    }
    return f[n];
}
```

# 例题

## 例题【NKOJ3584 集合划分】

## 解析

- 递推方法1（由递归方法1改编）
- 数组f[0..n]中的f[i]表示输入i时原题的答案
  - f[0]=1
  - $i \geq 2$时，$f[i] = \binom{i-1}{0} f[i-1] + \binom{i-1}{1} f[i-2] + \cdots + \binom{i-1}{i-1} f[0]$

# 例题

## 例题【NKOJ3584 集合划分】

## 解析

- 递推方法1（由递归方法1改编）
- 数组f[0..n]中的f[i]表示输入i时原题的答案
  - f[0]=1
  - $i \ge 2$时，$f[i] = \binom{i-1}{0} f[i-1] + \binom{i-1}{1} f[i-2] + \cdots + \binom{i-1}{i-1} f[0]$
- 时间复杂度$O(n^2)$，空间复杂度$O(n)$。

```
int main() {
    int n;
    scanf("%d", &n);
    f[0] = 1;
    for (int i = 1; i <= n; i ++) {
        for (int k = 0, c = 1; k <= i - 1; k ++) {
            f[i] += c * f[i - 1 - k];
            c = c * (i - 1 - k) / (k + 1);
        }
    }
    printf("%d\n", f[n]);
    return 0;
}
```

# 例题

## 例题【NKOJ3584 集合划分】

## 解析

- 递归方法2
- 函数F(n,m)计算把{1,2,...,n}划分成m个子集的方案数
    - 例如F(3,1)=1, F(3,2)=3, F(3,3)=1

# 例题

## 例题【NKOJ3584 集合划分】

## 解析

- 递归方法2
- 函数F(n,m)计算把{1,2,...,n}划分成m个子集的方案数
  - 例如F(3,1)=1， F(3,2)=3， F(3,3)=1
- 计算F(n,m)，把方案分成两类
  - $n$单独在一个子集，$1\sim n-1$被划分为m-1个子集
    方案数是F(n-1,m-1)
  - $n$所在子集还有其他数
    相当于先把$1\sim n-1$划分为m个子集，再把$n$加入进去
    方案数是m×F(n-1,m)

重庆南开中学

# 例题

## 例题【NKOJ3584 集合划分】

## 解析

- 递归方法2
- 函数F(n,m)计算把{1,2,...,n}划分成m个子集的方案数
  - 例如F(3,1)=1, F(3,2)=3, F(3,3)=1
- 计算F(n,m)，把方案分成两类
  - $n$单独在一个子集，$1\sim n-1$被划分为m-1个子集
    方案数是F(n-1,m-1)
  - $n$所在子集还有其他数
    相当于先把$1\sim n-1$划分为m个子集，再把$n$加入进去
    方案数是m×F(n-1,m)

以F(4,2)为例

1. 4单独一个子集，1,2,3划分为1个子集：
   {**{4}**,{1,2,3}}
2. 4所在集合还有其他数
   {{**4**,1},{2,3}}, {{1},{**4**,2,3}}
   {{**4**,2},{1,3}}, {{2},{**4**,1,3}}
   {{**4**,3},{1,2}}, {{3},{**4**,1,2}}
3. 所以F(4,2)=F(3,1)+2×F(3,2)
   =1+2×3
   =7

重庆南开中学

# 例题

## 例题【NKOJ3584 集合划分】

## 解析

- 递归方法2
- 函数F(n,m)计算把{1,2,...,n}划分成m个子集的方案数
  - 例如F(3,1)=1, F(3,2)=3, F(3,3)=1
- 计算F(n,m)，把方案分成两类
  - $n$单独在一个子集，$1{\sim}n-1$被划分为m-1个子集
    方案数是F(n-1,m-1)
  - $n$所在子集还有其他数
    相当于先把$1{\sim}n-1$划分为m个子集，再把$n$加入进去
    方案数是m×F(n-1,m)
- 最终答案=F(n,1)+F(n,2)+...+F(n,n)

重庆南开中学

# 例题

## 例题【NKOJ3584 集合划分】

## 解析

- 递归方法2
- 考虑边界情况
  - m=1时，答案是1
  - m=0时，答案是0
  - n<m时，答案是0
  - 其他时候，递归处理
    - $F(n,m)=F(n-1,m-1)+m\times F(n-1,m)$

# 例题

重庆南开中学

例题【NKOJ3584 集合划分】

## 解析

- 递归方法2
- 考虑边界情况
  - m=1时，答案是1
  - m=0时，答案是0
  - n<m时，答案是0
  - 其他时候，递归处理
    - F(n, m)=F(n-1, m-1)+m×F(n-1, m)
- 时间复杂度$O(n^2)$，空间复杂度$O(n^2)$。

```
int F(int n, int m) {
    static int f[MAX_N][MAX_N];
    if (m == 1) {
        return 1;
    } else if (m == 0 || n < m) {
        return 0;
    } else if (f[n][m]) {
        return f[n][m];
    } else {
        return f[n][m] = F(n - 1, m - 1) + m * F(n - 1, m);
    }
}
```

# 例题

重庆南开中学

## 例题【NKOJ3584 集合划分】

## 解析

- 递推方法2（由递归方法2改编）
- 数组f[0..n][0..n]中的f[i][j]表示把{1,2,...,i}划分成j个子集的方案数
  - f[1][1]=1
  - 其他时候：$f[i][j] = f[i-1][j-1]+j\times f[i-1][j]$

# 例题

## 例题【NKOJ3584 集合划分】

## 解析

- 递推方法2（由递归方法2改编）
- 数组f[0..n][0..n]中的f[i][j]表示把{1,2,...,i}划分成j个子集的方案数
  - f[1][1]=1
  - 其他时候：f[i][j] = f[i-1][j-1]+j×f[i-1][j]
- 时间复杂度$O(n^2)$，空间复杂度$O(n^2)$。

```
int main() {
    int n;
    scanf("%d", &n);
    f[1][1] = 1;
    for (int i = 2; i <= n; i ++) {
        for (int j = 1; j <= i; j ++) {
            f[i][j] = f[i - 1][j - 1] + j * f[i - 1][j];
        }
    }
    int ans = 0;
    for (int j = 1; j <= n; j ++) {
        ans += f[n][j];
    }
    printf("%d\n", ans);
    return 0;
}
```

# 例题

## 例题【NKOJ3584 集合划分】

## 解析

- 数学家们怎么说？
- 递归方法1的$F(n)$，被称为贝尔数；
  - 利用一些很厉害的推导，可以$O(n\log n)$计算$F(1)\sim F(n)$的每一项。
- 递归方法2的$F(n,m)$，被称为第二类斯特林数；
  - 利用一些很厉害的推导，可以$O(n\log n)$计算$F(n,m)$这一项。
- 有兴趣的同学可以去了解一下。

PART

03

# 又一道例题

- ？？

# 例题

重庆南开中学

## 例题（自然数拆分•改）

输入整数n，把n拆分成若干个**互不相同**的自然数的和，顺序不同算同种方案，计算方案数mod 1,000,000,007的结果。

样例输入:
6

样例输出:
4

样例解释:
6 = 6
6 = 5 + 1
6 = 4 + 2
6 = 3 + 2 + 1

样例输入:
7

样例输出:
5

样例解释:
7 = 7
7 = 6 + 1
7 = 5 + 2
7 = 4 + 3
7 = 4 + 2 + 1

# 例题

## 例题（自然数拆分•改）

输入整数n，把n拆分成若干个**互不相同**的自然数的和，顺序不同算同种方案，计算方案数mod 1,000,000,007的结果。

## 解析

- 递归方法:
- decom(n,m)表示把n拆分成不超过m的数的方案数
  - 当 $m(m+1)/2 < n$ 时，答案是0
  - 当 $n=0$ 或 $n=1$ 时，答案是1
  - 其他时候，decom(n,m)=decom(n-m,**m-1**)+decom(n,m-1)

重庆南开中学

# 例题

## 例题（自然数拆分●改）

输入整数n，把n拆分成若干个**互不相同**的自然数的和，顺序不同算同种方案，计算方案数mod 1,000,000,007的结果。

## 解析

- 递归方法：
- decom(n,m)表示把n拆分成不超过m的数的方案数
  - 当 m(m+1)/2 < n 时，答案是0
  - 当 n=0 或 n=1 时，答案是1
  - 其他时候，decom(n,m)=decom(n−m,**m−1**)+decom(n,m−1)
- 时间复杂度$O(n^2)$。

```
//把n拆成最大不超过m
int dec(int n, int m) {
    static int f[MAX_N][MAX_N];
    int x = m * (m + 1) / 2;
    if (x < n) {
        return 0;
    } else if (n == 0 || n == 1) {
        return 1;
    } else if (f[n][m]) {
        return f[n][m];
    } else if (n < m) {
        return f[n][m] = dec(n, n);
    } else {
        int t = dec(n - m, m - 1) + dec(n, m - 1);
        return f[n][m] = t < P ? t : t - P;
    }
}
```

# 例题



## 例题（自然数拆分●改）

输入整数n，把n拆分成若干个**互不相同**的自然数的和，顺序不同算同种方案，计算方案数mod 1,000,000,007的结果。

## 解析

- 递归方法2？

# 例题

## 例题（自然数拆分●改）

输入整数n，把n拆分成若干个**互不相同**的自然数的和，顺序不同算同种方案，计算方案数mod 1,000,000,007的结果。

## 解析

- 递归方法2:
- decom(n,m)表示把n拆分成m个数的方案数
  - 当 $m(m+1)/2 > n$ 或 $m=0$ 时，答案是0
  - 当 $m(m+1)/2 = n$ 或 $m=1$ 时，答案是1
  - 其他时候，假设把每个数都减1，可以得到 decom(n,m)=decom(n-m,m)+decom(n-m,m-1)

# 例题



## 例题（自然数拆分●改）

输入整数n，把n拆分成若干个**互不相同**的自然数的和，顺序不同算同种方案，计算方案数mod 1,000,000,007的结果。

## 解析

- 递归方法2:
- decom(n,m)表示把n拆分成m个数的方案数
  - 当 $m(m+1)/2 > n$ 或 $m=0$ 时，答案是0
  - 当 $m(m+1)/2 = n$ 或 $m=1$ 时，答案是1
  - 其他时候，假设把每个数都减1，可以得到 decom(n,m)=decom(n-m,m)+decom(n-m,m-1)

```
//把n拆成m个数
int dec(int n, int m) {
    static int f[MAX_N][MAX_M];
    int x = m * (m + 1) / 2;
    if (n < x || m == 0) {
        return 0;
    } else if (n == x || m == 1) {
        return 1;
    } else if (f[n][m]) {
        return f[n][m];
    } else {
        int t = dec(n - m, m - 1) + dec(n - m, m);
        return f[n][m] = t < P ? t : t - P;
    }
}
```

# 例题

## 例题（自然数拆分●改）

输入整数n，把n拆分成若干个**互不相同**的自然数的和，顺序不同算同种方案，计算方案数mod 1,000,000,007的结果。

## 解析

- 递归方法2：
- decom(n,m)表示把n拆分成m个数的方案数
    - 当 m(m+1)/2 > n 或 m=0 时，答案是0
    - 当 m(m+1)/2 = n 或 m=1 时，答案是1
    - 其他时候，假设把每个数都减1，可以得到

      decom(n,m)=decom(n-m,m)+decom(n-m,m-1)
- 最终答案=decom(n,1)+decom(n,2)+...+decom(n,m)
    - 其中m是满足x(x+1)/2 <= n的最大的x

```
//把n拆成m个数
int dec(int n, int m) {
    static int f[MAX_N][MAX_M];
    int x = m * (m + 1) / 2;
    if (n < x || m == 0) {
        return 0;
    } else if (n == x || m == 1) {
        return 1;
    } else if (f[n][m]) {
        return f[n][m];
    } else {
        int t = dec(n - m, m - 1) + dec(n - m, m);
        return f[n][m] = t < P ? t : t - P;
    }
}
```

```
int main() {
    scanf("%d", &N);
    int ans = 0;
    for (int m = 1; m * (m + 1) / 2 <= N ; m ++) {
        ans += dec(N, m);
        ans -= ans >= P ? P : 0;
    }
    printf("%d\n", ans);
    return 0;
}
```

# 例题

重庆南开中学

## 例题（自然数拆分●改）

输入整数n，把n拆分成若干个**互不相同**的自然数的和，顺序不同算同种方案，计算方案数mod 1,000,000,007的结果。

## 解析

- 递归方法2：
- decom(n,m)表示把n拆分成m个数的方案数
  - 当 m(m+1)/2 > n 或 m=0 时，答案是0
  - 当 m(m+1)/2 = n 或 m=1 时，答案是1
  - 其他时候，假设把每个数都减1，可以得到

    decom(n,m)=decom(n-m,m)+decom(n-m,m-1)
- 最终答案=decom(n,1)+decom(n,2)+...+decom(n,m)
  - 其中m是满足x(x+1)/2 <= n的最大的x
- 时间复杂度？数组f[n][m]每个数都只被算一次
  - $O(nm) = O(n^{1.5})$

```
//把n拆成m个数
int dec(int n, int m) {
    static int f[MAX_N][MAX_M];
    int x = m * (m + 1) / 2;
    if (n < x || m == 0) {
        return 0;
    } else if (n == x || m == 1) {
        return 1;
    } else if (f[n][m]) {
        return f[n][m];
    } else {
        int t = dec(n - m, m - 1) + dec(n - m, m);
        return f[n][m] = t < P ? t : t - P;
    }
}
```

```
int main() {
    scanf("%d", &N);
    int ans = 0;
    for (int m = 1; m * (m + 1) / 2 <= N ; m ++) {
        ans += dec(N, m);
        ans -= ans >= P ? P : 0;
    }
    printf("%d\n", ans);
    return 0;
}
```

# 例题

重庆南开中学

## 例题（自然数拆分●改）

输入整数n，把n拆分成若干个**互不相同**的自然数的和，顺序不同算同种方案，计算方案数mod 1,000,000,007的结果。

## 解析

- 递推方法（由递归方法1改编）
- 数组f[0..n][0..n]
    - f[i][j]表示用把j拆分成不超过i的数的方案数
    - f[0][0]=1，当j>0时f[0][j]=0

重庆南开中学

# 例题

## 例题（自然数拆分●改）

输入整数n，把n拆分成若干个**互不相同**的自然数的和，顺序不同算同种方案，计算方案数mod 1,000,000,007的结果。

## 解析

- 递推方法（由递归方法1改编）
- 数组f[0..n][0..n]
    - f[i][j]表示用把j拆分成不超过i的数的方案数
    - f[0][0]=1，当j>0时f[0][j]=0
    - 递归公式：$1\leqslant i\leqslant N$
        - 当i>j时， f[i][j]=f[i-1][j]
        - 当i≤j时，f[i][j]=f[i-1][j-i]+f[i-1][j]

# 例题

重庆南开中学

## 例题（自然数拆分●改）

输入整数n，把n拆分成若干个**互不相同**的自然数的和，顺序不同算同种方案，计算方案数mod 1,000,000,007的结果。

## 解析

- 递推方法（由递归方法1改编）
- 数组f[0..n][0..n]
  - f[i][j]表示用把j拆分成不超过i的数的方案数
  - f[0][0]=1，当j>0时f[0][j]=0
  - 递归公式：1⩽i⩽N
    - 当i>j时， f[i][j]=f[i-1][j]
    - 当i⩽j时，f[i][j]=f[i-1][j-i]+f[i-1][j]
- 时间复杂度$O(n^2)$。

```
int main() {
    int N;
    scanf("%d", &N);
    f[0][0] = 1;
    for (int i = 1; i <= N; i ++) {
        for (int j = 0; j < i; j ++) {
            f[i][j] = f[i - 1][j];
        }
        for (int j = i; j <= N; j ++) {
            f[i][j] = (f[i - 1][j - i] + f[i - 1][j]) % P;
        }
    }
    printf("%d\n", f[N][N]);
    return 0;
}
```

重庆南开中学

# 例题

## 例题（自然数拆分●改）

输入整数n，把n拆分成若干个**互不相同**的自然数的和，顺序不同算同种方案，计算方案数mod 1,000,000,007的结果。

## 解析

- 递推方法（由递归方法1改编）●优化
- 观察数组访问情况
  - 外层for循环从小到大枚举i，数组只使用f[i][]和f[i-1][]两行！
  - 用到f[i-1][]行的前半部分，所以内层for循环改成从大到小枚举j！！

# 例题



## 例题（自然数拆分●改）

输入整数n，把n拆分成若干个**互不相同**的自然数的和，顺序不同算同种方案，计算方案数mod 1,000,000,007的结果。

## 解析

- 递推方法（由递归方法1改编）●优化
- 观察数组访问情况
  - 外层for循环从小到大枚举i，数组只使用f[i][]和f[i-1][]两行！
  - 用到f[i-1][]行的前半部分，所以内层for循环改成从大到小枚举j！！
- 时间复杂度$O(n^2)$，空间复杂度$O(n)$

```
int main() {
    int N;
    scanf("%d", &N);
    f[0] = 1;
    for (int i = 1; i <= N; i ++) {
        for (int j = N; j >= i; j --) {
            f[j] = (f[j] + f[j - i]) % P;
        }
    }
    printf("%d\n", f[N]);
    return 0;
}
```

# 例题

## 例题（自然数拆分●改）

输入整数n，把n拆分成若干个**互不相同**的自然数的和，顺序不同算同种方案，计算方案数mod 1,000,000,007的结果。

## 解析

- 递推方法（由递归方法2改编）
- 数组f[0..m][0..n]
  - f[i][j]表示把j拆成i个数的方案数
  - f[0][0]=1，当j>0时f[0][j]=0
  - 递推公式：1≤i，i(i+1)/2≤n
    - f[i][j]=f[i-1][j-i]+f[i][j-i]

# 例题

重庆南开中学

## 例题（自然数拆分•改）

输入整数n，把n拆分成若干个**互不相同**的自然数的和，顺序不同算同种方案，计算方案数mod 1,000,000,007的结果。

## 解析

- 递推方法（由递归方法2改编）
- 数组f[0..m][0..n]
  - f[i][j]表示把j拆成i个数的方案数
  - f[0][0]=1，当j>0时f[0][j]=0
  - 递推公式：1≤i，i(i+1)/2≤n
    - f[i][j]=f[i-1][j-i]+f[i][j-i]
- 答案=f[1][n]+f[2][n]+...+f[m][n]
- 时间复杂度$O(n^{1.5})$。

```
int main() {
    int N;
    scanf("%d", &N);
    f[0][0] = 1;
    for (int i = 1; i * (i + 1) / 2 <= N; i ++) {
        for (int j = i; j <= N; j ++) {
            f[i][j] = (f[i - 1][j - i] + f[i][j - i]) % P;
        }
    }
    int ans = 0;
    for (int i = 1; i * (i + 1) / 2 <= N; i ++) {
        ans = (ans + f[i][N]) % P;
    }
    printf("%d\n", ans);
    return 0;
}
```

重庆南开中学

# 例题

## 例题（自然数拆分●改）

输入整数n，把n拆分成若干个**互不相同**的自然数的和，顺序不同算同种方案，计算方案数mod 1,000,000,007的结果。

## 解析

- 递推方法（由递归方法2改编）●优化
- 数组同时使用的是f[i][]和f[i-1][]两行
- 不能改成一维数组，但可以用滚动数组
  - 开f[2][N]大小的数组
  - 用i%2代替i做数组下标

# 例题

## 例题（自然数拆分●改）

输入整数n，把n拆分成若干个**互不相同**的自然数的和，顺序不同算同种方案，计算方案数mod 1,000,000,007的结果。

## 解析

- 递推方法（由递归方法2改编）●优化
- 数组同时使用的是f[i][]和f[i-1][]两行
- 不能改成一维数组，但可以用滚动数组
  - 开f[2][N]大小的数组
  - 用i%2代替i做数组下标
- 时间复杂度$O(n^{1.5})$，空间复杂度$O(n)$

```
int main() {
    int N, ans = 0;
    scanf("%d", &N);
    f[0][0] = 1;
    for (int i = 1; i * (i + 1) / 2 <= N; i ++) {
        int ii = i % 2;
        for (int j = 0; j < i; j ++) {
            f[ii][j] = 0;
        }
        for (int j = i; j <= N; j ++) {
            f[ii][j] = (f[1 - ii][j - i] + f[ii][j - i]) % P;
        }
        ans = (ans + f[ii][N]) % P;
    }
    printf("%d\n", ans);
    return 0;
}
```

# 例题

## 例题（自然数拆分•改）

输入整数n，把n拆分成若干个**互不相同**的自然数的和，顺序不同算同种方案，计算方案数mod 1,000,000,007的结果。

## 效率对比

- 测试时设置n=20000，每个程序都算出了正确答案818602017。

| 方法 | 时间(ms) | 空间(MB) |
|---|---|---|
| 递归方法1 | 3412 | 1600 |
| 递归方法2 | 81 | 16 |
| 递推方法1 | 1776 | 1600 |
| 递推方法1+优化 | 516 | 小于1 |
| 递推方法2 | 22 | 16 |
| 递推方法2+优化 | 10 | 小于1 |

大幅优化

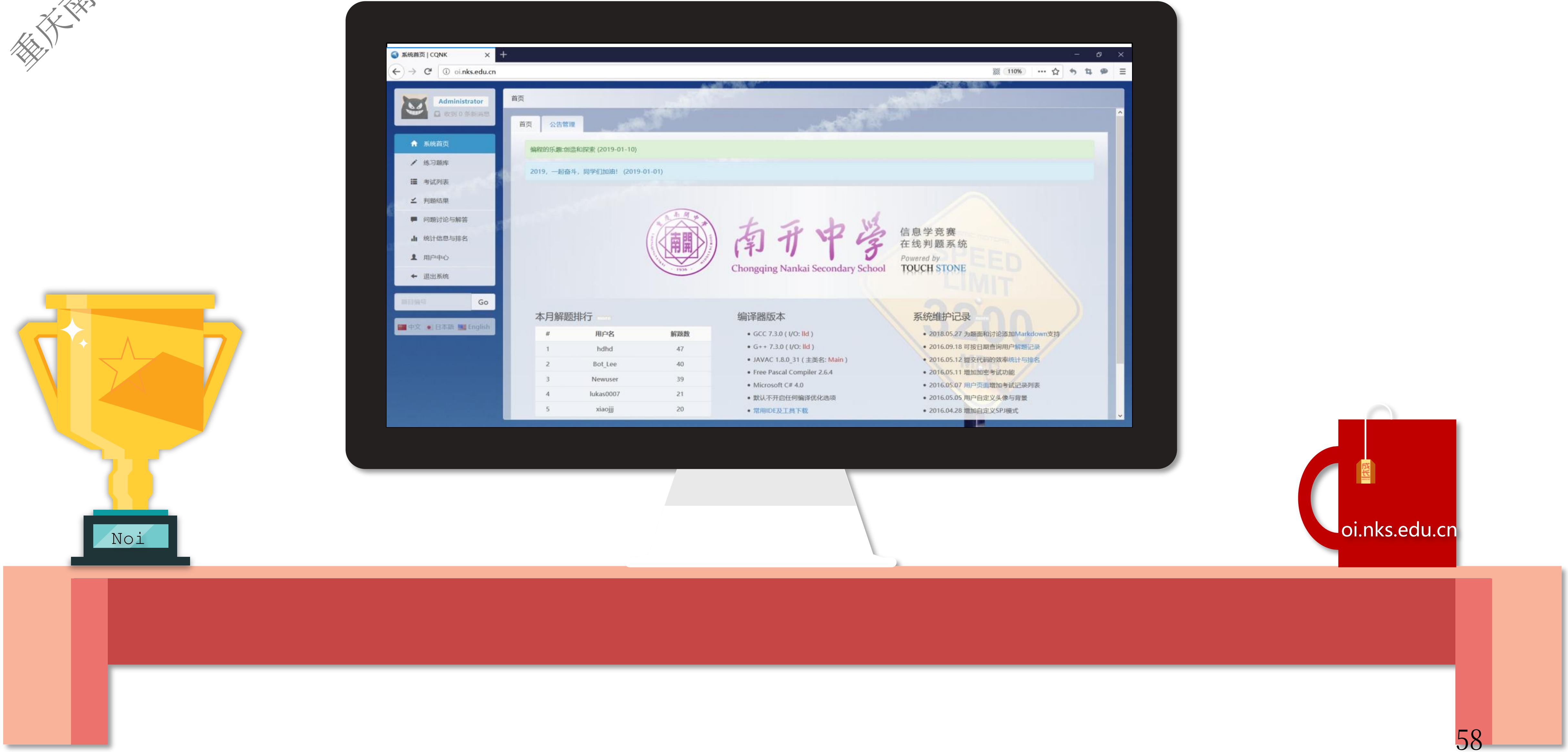
重庆南开中学

# 课后练习

## 递归、递推习题

NKOJ3526　　放苹果
NKOJ1334　　自然数拆分（递推方法）
NKOJ5900　　自然数拆分+
NKOJ5901　　集合划分+

重庆南开中学



系统首页 | CQNK
oi.nks.edu.cn
Administrator
首页
公告管理
系统首页
练习题库
考试列表
判题结果
问题讨论与解答
统计信息与排名
用户中心
退出系统
Go
中文
日本語
English
编程的乐趣:创造和探索 (2019-01-10)
2019，一起奋斗，同学们加油！ (2019-01-01)
南开中学
Chongqing Nankai Secondary School
信息学竞赛
在线判题系统
Powered by
TOUCH STONE
本月解题排行
| # | 用户名 | 解题数 |
|---|---|---|
| 1 | hdhd | 47 |
| 2 | Bot_Lee | 40 |
| 3 | Newuser | 39 |
| 4 | lukas0007 | 21 |
| 5 | xiaojjj | 20 |
编译器版本
GCC 7.3.0 ( I/O: lld )
G++ 7.3.0 ( I/O: lld )
JAVAC 1.8.0_31 ( 主类名: Main )
Free Pascal Compiler 2.6.4
Microsoft C# 4.0
默认不开启任何编译优化选项
常用IDE及工具下载
系统维护记录
2018.05.27 为题面和讨论添加Markdown支持
2016.09.18 可按日期查询用户解题记录
2016.05.12 提交代码的效率统计与排名
2016.05.11 增加加密考试功能
2016.05.07 用户页面增加考试记录列表
2016.05.05 用户自定义头像与背景
2016.04.28 增加自定义SPJ模式
Noi
oi.nks.edu.cn