# 算法

## 第一章 动态规划

### 1.1 除数博弈（Week0001\_\_20190714）

**1025. Divisor Game（easy）**

Alice and Bob take turns playing a game, with Alice starting first.

Initially, there is a number N on the chalkboard. On each player's turn, that player makes a move consisting of:

Choosing any x with 0 < x < N and N % x == 0.

Replacing the number N on the chalkboard with N - x.

Also, if a player cannot make a move, they lose the game.

Return True if and only if Alice wins the game, assuming both players play optimally.

**Example 1:**

**Input:** 2

**Output:** true

**Explanation:** Alice chooses 1, and Bob has no more moves.

**Example 2:**

**Input:** 3

**Output:** false

**Explanation:** Alice chooses 1, Bob chooses 1, and Alice has no more moves.

**Note:**

1. 1 <= N <= 1000

解答1(动态规划)：本题首先我们对N = 1~10进行分析，发现后面的数值计算依赖于前面比其小的数值，所以我们可以采用动态规划算法，构建dp[]。

class Solution {

public:

bool divisorGame(int N) {

if(N > 1000 || N < 1)

return false;

vector<bool> dp(1001, false);

dp[1] = false;

for(int value = 2; value <= N; value++)

{

for(int factor = 1; factor < value; factor++)

{

if((value % factor == 0) && !dp[value-factor])

{

dp[value] = true;

break;

}

}

}

return dp[N];

}

};

解答2(数学)：须知：奇数的因子必为奇数，奇数与其因子之差必为偶数；偶数的因子或奇或偶，偶数与其因子之差或奇或偶。本题博弈中，得奇数者必败，得偶数者必胜。采用数学归纳法证明如下：

1、显然divisorGame(1)== false和divisorGame(2)== true正确；

2、假设对于任何正数n，divisorGame(2 \* n – 1)== false和divisorGame(2 \* n)== true。我们将证明它是2 \* (n + 1) -1和2 \* (n + 1)。

2.1、2 \* (n + 1) -1是奇数，所以它的任何因子都是奇数，因此差值2 \* (n + 1) -1 - x（x是任意2 \* (n + 1) -1的因子）是偶数，但正如我们所知divisorGame（2 \* n）== true对于任何正数n都是真，这意味着任何可能的Alice的动作都会引起Bob的胜利，所以divisorGame(2 \* (n + 1) -1) == false。

2.2、2 \* (n + 1)是偶数，Alice只需选择x == 1.因为我们在2.1中得到，divisorGame(2 \* (n + 1) -1) == false，所以divisorGame(2 \* (n + 1))== true。

class Solution {

public:

bool divisorGame(int N) {

if(N > 1000 || N < 1)

return false;

return !(N & 1);

}

};