Problem 464: Can I Win

Problem Information

Difficulty: Medium
Acceptance Rate: 30.86%
Paid Only: No
Tags: Math, Dynamic Programming, Bit Manipulation, Memoization, Game Theory, Bitmask

Problem Description

In the "100 game" two players take turns adding, to a running total, any integer from `1` to `10`. The player who first causes the running total to **reach or exceed** 100 wins.

What if we change the game so that players **cannot** re-use integers?

For example, two players might take turns drawing from a common pool of numbers from 1 to 15 without replacement until they reach a total >= 100.

Given two integers `maxChoosableInteger` and `desiredTotal`, return `true` if the first player to move can force a win, otherwise, return `false`. Assume both players play **optimally**.

Example 1:

Input: maxChoosableInteger = 10, desiredTotal = 11 **Output:** false **Explanation:** No matter which integer the first player choose, the first player will lose. The first player can choose an integer from 1 up to 10. If the first player choose 1, the second player can only choose integers from 2 up to 10. The second player will win by choosing 10 and get a total = 11, which is >= desiredTotal. Same with other integers chosen by the first player, the second player will always win.

Example 2:

Input: maxChoosableInteger = 10, desiredTotal = 0 **Output:** true

Example 3:

Input: maxChoosableInteger = 10, desiredTotal = 1 **Output:** true

Constraints:

* `1 <= maxChoosableInteger <= 20` * `0 <= desiredTotal <= 300`

Code Snippets

C++:

```
class Solution {
public:
bool canIWin(int maxChoosableInteger, int desiredTotal) {

}
};
```

Java:

```
class Solution {
public boolean canIWin(int maxChoosableInteger, int desiredTotal) {

}
}
```

Python3:

```
class Solution:
def canIWin(self, maxChoosableInteger: int, desiredTotal: int) -> bool:
```