题目：

Given an integer array with all positive numbers and no duplicates, find the number of possible combinations that add up to a positive integer target.

Example:

```
nums = [1, 2, 3]
target = 4

The possible combination ways are:
(1, 1, 1, 1)
(1, 1, 2)
(1, 2, 1)
(1, 3)
(2, 1, 1)
(2, 2)
(3, 1)

Note that different sequences are counted as different combinations.

Therefore the output is 7.
```

Follow up:

What if negative numbers are allowed in the given array?

How does it change the problem?

What limitation we need to add to the question to allow negative numbers?

思路：

一开始想到用 DFS 来做，但是有个问题就是这种方式得到的答案各个数字排列是无序的，也就是 1, 3 和 3, 1 这种只是一个答案，然后又想把数字保存起来，在得到一个答案的时候对这些数字再求一次总共排列的个数，这种方式还有问题就是在求总排列个数的时候比如 2, 1, 1 三个加一起等于 4，总的排列个数即为(3!/2!)，但是当数字个数很多的时候阶乘太大，根本无法计算.

然后就想到可以用动态规划来做，也是一个背包问题，求出[1, target]之间每个位置有多少种排列方式，这样将问题分化为子问题．状态转移方程可以得到为:

dp[i] = sum(dp[i - nums[j]]), (i-nums[j] > 0);

如果允许有负数的话就必须要限制每个数能用的次数了，不然的话就会得到无限大的排列方式，比如 1, -1, target = 1;

1.时间：O（ ）;空间：O（N）　　　-->超时

```
class Solution {
public:
    int combinationSum4(vector<int>& nums, int target) {
        if (target < 1 || nums.empty()) return 0;
        std::sort(nums.begin(), nums.end());
        return dfs(nums, target);
    }
private:
    int dfs(const std::vector<int>& nums, int target){
        if (target == 0) return 1;
        int count = 0;
        for (int i = 0; i < nums.size(); ++i){
            if (target < nums[i]) break;
            count += dfs(nums, target - nums[i]);
        }
        return count;
    }
};
```

2.时间：O（max（target*N, NLOGN））; 空间：O（N）

/* 背包问题：dp[i] = sum(dp[i-nums[k]])，i >= nums[k]

```
 */
class Solution {
public:
    int combinationSum4(vector<int>& nums, int target) {
        if (target < 1 || nums.empty()) return 0;
        std::sort(nums.begin(), nums.end());
        std::vector<int> dp(target + 1, 0);
        dp[0] = 1;
        for (int i = 1; i <= target; ++i){
            for (int k = 0; k < nums.size() && nums[k] <= i; ++k){
                dp[i] += dp[i - nums[k]];
            }
        }
        return dp[target];
    }
};
```