Author: Xiaoqiang Yan

LeetCode 486 Predict the Winner

Given an array of scores that are non-negative integers. Player 1 picks one of the numbers from either end of the array followed by the player 2 and then player 1 and so on. Each time a player picks a number, that number will not be available for the next player. This continues until all the scores have been chosen. The player with the maximum score wins.

Given an array of scores, predict whether player 1 is the winner. You can assume each player plays to maximize his score.

Example 1:

```
Input: [1, 5, 2]
Output: False
Explanation: Initially, player 1 can choose between 1 and 2.
If he chooses 2 (or 1), then player 2 can choose from 1 (or 2
) and 5. If player 2 chooses 5, then player 1 will be left wi
th 1 (or 2).
So, final score of player 1 is 1 + 2 = 3, and player 2 is 5.
Hence, player 1 will never be the winner and you need to retu
rn False.
```

Example 2:

```
Input: [1, 5, 233, 7]
```

```
Output: True
Explanation: Player 1 first chooses 1. Then player 2 have to
choose between 5 and 7. No matter which number player 2 choos
e, player 1 can choose 233.
Finally, player 1 has more score (234) than player 2 (12), so
 you need to return True representing player1 can win.
```

Note:
1 <= length of the array <= 20.
Any scores in the given array are non-negative integers and will not exceed 10,000,000.
If the scores of both players are equal, then player 1 is still the winner.

Solution

we have an array `[4,5,7,1]`, if player1 wants to get max and go firstly, he could pick `4` or `1`, sub_array for player2 are `[5,7,1]` or `[4,5,7]`. Thus, we find sub_problems.
For player1, RES[4,5,7,1] = max(RES[5,7,1]+4, RES[4,5,7]+1);
For player2, RES[5,7,1] = min(RES[7,1], RES[5,7]);
We find we need 2 variable to stand the array-start and end; we also need one more variable to stand for the players.

Code

```
bool PredictTheWinner(vector<int>& nums) {
    //if player 1 get more than half of the total sum, he wil
l win.
```

```
    float total = 0;  // total should be float, otherwise total/2 will be int.
    for(auto it: nums)
        total += it;
    vector<vector<vector<int>>> cache(nums.size(),vector<vector<int>>(nums.size(), vector<int>(2,0)));
    return helper(nums, cache, 0, nums.size()-1, 1) >= total/2;
}
int helper(vector<int>& nums, vector<vector<vector<int>>>& cache, int start, int end, int player){
    if(end < start) return 0;
    if(cache[start][end][player] != 0 )
        return cache[start][end][player];
    int optimal = 0;
    //if it's player1's turn, he will get the max of two situations(pick first or pick last, then it is player2's turn.
    if(player == 1){
        optimal = max(nums[start] + helper(nums, cache, start+1, end, 0), nums[end] + helper(nums, cache, start, end-1, 0));
    }
    else if(player == 0)
        optimal = min(helper(nums, cache, start+1, end, 1), helper(nums, cache, start, end-1, 1)); // if player2 wants to get max, he should let player1 get min.
    cache[start][end][player] = optimal;
    return optimal;
}
```

Note

`variable total_sum should be float.`

`for[1,3,1], if total is int, 5/2 = 2, not 2.5, then 2>=5/2`

`return ture;`

Summary

3D DP