**Two Pointers**

**Container With Most Water** (**11**)

1. **Problem**: Given array, return maximum of (j-i)\*min(height[i], height[j]);
2. **Solution**: Time Complexity should be O(n)
   1. Set pointer 1 at the beginning and pointer 2 at the end;
   2. Since the height is determined by the smaller one, move the pointer with smaller height.

**3Sum** (**15**)

1. **Problem**: Given array, return all no duplicate triplets that sum to zero;
2. **Solution**:
   1. Use dictionary to reduce the time complexity from O(n^3) to O(n^2);
   2. Have a current pointer, know the relationship between the current one and the other two;
   3. Need to consider how to avoid duplicates.

**3Sum Closest** (**16**)

1. **Problem**: Given array and integer, return three integers that sum is closest to the integer;
2. **Solution**: Time Complexity should be O(n^2); Space Complexity should be O(n)
   1. When use two pointers, first think of sorting and a current pointer;
   2. Move the current pointer with O(n) and the other two pointers with O(n), O(n^2) in total.