494. Target Sum
n**2

```
class Solution:
    def findTargetSumWays(self, nums: List[int], S: int) -> int:
        total = sum(nums)
        return 0 if total < S or (total + S) % 2 == 1 else self.getTargetNums(nums, (total + S) // 2)

    def getTargetNums(self, nums, target):
        dp = [0] * (target + 1)
        dp[0] = 1
        for num in nums:
            for i in range(target, num - 1, -1):
                dp[i] += dp[i - num]
        return dp[target]
```

69. Sqrt(x)
Binary search

```
class Solution:
    def mySqrt(self, x: int) -> int:
        l, r = 0, x
        while l < r:
            mid = (l + r) // 2
            cur_val = mid ** 2
            if cur_val == x:
                return mid
            elif cur_val < x:
                l = mid + 1
            else:
                r = mid - 1
        return l if l ** 2 <= x else l - 1
```

67. Add Binary

```
class Solution:
    def addBinary(self, a: str, b: str) -> str:
        arr_a, arr_b = list(a), list(b)
        result, carry = [], 0
        while arr_a and arr_b:
            num_a = int(arr_a.pop())
            num_b = int(arr_b.pop())
            carry, ret= divmod((num_a + num_b + carry), 2)
            result.append(ret)
        arr_a = arr_a or arr_b
        while arr_a:
            num_a = int(arr_a.pop())
            carry, ret = divmod((num_a + carry), 2)
```

```
            result.append(ret)
        if carry:
            result.append(carry)
        return ''.join(map(str, result[::-1]))
```

415. Add Strings

```
class Solution:
    def addStrings(self, num1: str, num2: str) -> str:
        arr_a, arr_b = list(num1), list(num2)
        result, carry = [], 0
        while arr_a and arr_b:
            num_a = int(arr_a.pop())
            num_b = int(arr_b.pop())
            carry, ret= divmod((num_a + num_b + carry), 10)
            result.append(ret)
        arr_a = arr_a or arr_b
        while arr_a:
            num_a = int(arr_a.pop())
            carry, ret = divmod((num_a + carry), 10)
            result.append(ret)
        if carry:
            result.append(carry)
        return ''.join(map(str, result[::-1]))
```

2. Add Two Numbers

```
class Solution:
    def addTwoNumbers(self, l1: ListNode, l2: ListNode) -> ListNode:
        head = ListNode(0)
        carry = 0
        head_mem = head
        while l1 and l2:
            carry, val = divmod(l1.val + l2.val + carry, 10)
            head.next = ListNode(val)
            head = head.next
            l1 = l1.next
            l2 = l2.next
        l1 = l1 or l2
        while l1:
            carry, val = divmod(l1.val + carry, 10)
            head.next = ListNode(val)
            head = head.next
            l1 = l1.next
        if carry:
            head.next = ListNode(carry)
```

```
return head_mem.next
```