Add Two Numbers

```
class ListNode:
   def __init__(self, val=0, next=None):
         self.val = val
         self.next = None

class Solution:
    def addTwoNumbers(self, l1: Optional[ListNode], l2: Optional
        stack = []
        stack2 = []
        num1 = []
        num2 = []
        while l1 is not None:
            stack.append(l1.val)
            l1 = l1.next



        while l2 is not None:
            stack2.append(l2.val)
            l2 = l2.next

        while len(stack) != 0:
            value = stack.pop()
            num1.append(value)

        while len(stack2) != 0:
            value = stack2.pop()
            num2.append(value)
```

```
        num1 = int("".join(map(str, num1)))
        num2 = int("".join(map(str, num2)))

        returnSum = (num1) + (num2)
        returnSum = str(returnSum)



        dummy_returnList = ListNode(0)
        current = dummy_returnList

        for n in returnSum[::-1]:
         node = ListNode(n)
         current.next = node
         current = node

        return dummy_returnList.next
```

- Rather than constantly switching the data type, better to keep it as 1, more memory efficient
- Can add integer by integer and just use a carry variable
- This did not need rearranging of stack, could be done a lot better

Improvements to be made :

Understand the arithmetic's involved more

Properly make use of the stack function

Better understand linked lists