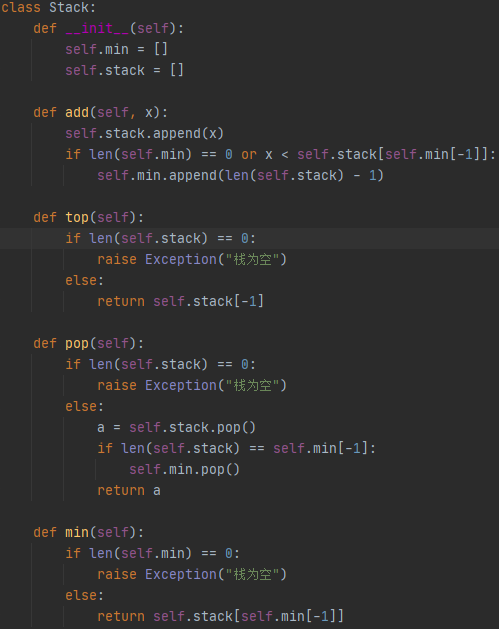
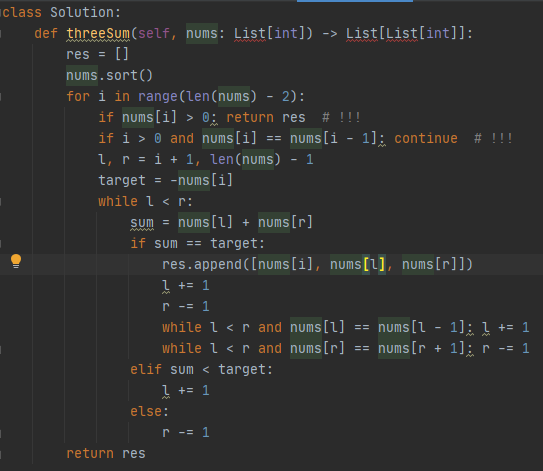
# 内功修炼

1. 在栈中实现函数找到栈中的最小值



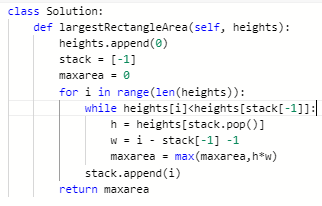
* 1. 用空间换时间，引入一个记录最小值**索引**的min\_stack，并维护这个栈
  2. 增加的时候，如果值比栈中的数还小，那么就加入这个值在主栈中的索引
  3. 删除的时候，如果这个值的索引和min\_stack中的最小值的索引相等，则pop出min\_stack中的栈顶元素
  4. 如果操作的时候遇到空栈，就需要返回一个异常

1. 三数之和



* 1. 优化速度，对于第一个数字i，遇到重复的数字，第一次要先做，第二次才能略过，即nums[i] = =nums[i-1] 而不是nums[i] ==nums[i+1]，不然会导致漏可能性
  2. 尽量不要在if判断的时候，采取加减法，效率很慢

1. 柱状图中最大的矩形



* 1. 用了一个哨兵思想，在stack开头放了一个-1 在heights结尾放了一个0
  2. stack中有一个-1的作用，在单调栈中找前一个小于自己的索引时，可以完美的处理边界问题
  3. 结尾放0，在最后的时候，用0来表示栈中元素的右边界，可以清空栈，直到剩下最后的-1

1. 设计循环双端队列
   1. head端是先放值，再++，tail端是先--，再放值
   2. 要多设计一位容积k+1，用来区分是空还是满队列

