**739. Daily Temperature**

Given a list of daily temperatures T, return a list such that, for each day in the input, tells you how many days you would have to wait until a warmer temperature. If there is no future day for which this is possible, put 0 instead.

For example, given the list of temperatures T = [73, 74, 75, 71, 69, 72, 76, 73], your output should be [1, 1, 4, 2, 1, 1, 0, 0].

**Note:** The length of temperatures will be in the range [1, 30000]. Each temperature will be an integer in the range [30, 100].

这次的题目不是关于二叉树的。

首先一看到题目大家想到的方法肯定是遍历，套两层循环暴力搜索气温升高的那天，然后坐标相减得出结果。但是注意Note里面说了可能会有30000天的温度，所以这条路就被否决掉了。但是这并不代表遍历不行，转换角度从温度进行遍历就会快很多。详细可参考这个微博：<https://blog.csdn.net/fuxuemingzhu/article/details/79285081>

但是作为一个优雅的程序员，光写这种代码肯定是不行的。所以今天我们就要用堆栈stack来解决这道题。

其实要我自己写我肯定也想不到要用堆栈，但是看完别人的代码后就发现用堆栈还是有一定道理的。堆栈的先进后出和pop功能可以有效的再只遍历一次的情况下得到结果。

**思路：**

我们以题目中给出的例子为例来讲解：[73, 74, 75, 71, 69, 72, 76, 73]。

我们定义一个空栈和全零的output list，然后在用for循环遍历温度的时候分别做如下处理（注：使用enumerate可以同时获得下标i和温度t。 for i, t in enumerate(T): ）：

1）如果第二天的温度比当天的低，那么我们暂时还无法确定需要等多少天才能有更高的温度，所以就将当天的温度入栈（ stack.append(i) ）。

2）如果第二天温度比当天的高，那么它需要等待的天数就是1（如当t=74，栈顶元素为73时）。此时我们需要更新栈顶元素对应的那天的等待时间，把结果放进output list里。即当t=74，栈顶元素为73时，其对应结果为1（output[cur] = i-cur），此时由于73已经找到结果，所以要被pop出；当t=75的时候同理。

当t=71时，由于71<75 情况1），所以我们将75入栈（此时栈内元素为75）。

当t=69时，由于69<71 情况1），71也同理入栈（此时栈内元素为75,71）。

当t=72时，由于69<72 情况2），所以69的对应结果为1且69找到答案被pop出栈。当前栈顶元素为71，然后71<72，所以此时可以修改71对应的等待天数，即为72对应的天数索引与71对应的天数索引，然后71出栈（此时栈内元素为75）；

当t=76时，由于71<76 情况2），结果为1，但是同时由于76大于栈顶元素75，所以修改75的等待天数，并出栈（此时栈内为空）。

当t=73时，由于76>73 情况1）76需入栈。由于再也没有比76大的元素，并且73之后也没有元素了，所以76，73对应的等待天数就都为0了，在程序中并没有做处理。

算法的空间复杂度为O(n)，时间复杂度也为O(n)，这是因为for循环里面的while循环中，每个元素最多出一次栈，所以while总的执行次数是O(n)量级的。

