1191. K-Concatenation Maximum Sum

Given an integer array `arr` and an integer `k`, modify the array by repeating it `k` times.

For example, if `arr = [1, 2]` and `k = 3` then the modified array will be `[1, 2, 1, 2, 1, 2]`.

Return the maximum sub-array sum in the modified array. Note that the length of the sub-array can be `0` and its sum in that case is `0`.

As the answer can be very large, return the answer **modulo** `10<sup>9</sup> + 7`.

Example 1:

```
Input: arr = [1,2], k = 3
Output: 9
```

Example 2:

```
Input: arr = [1,-2,1], k = 5
Output: 2
```

Example 3:

```
Input: arr = [-1,-2], k = 7
Output: 0
```

```
class Solution:
    def kConcatenationMaxSum(self, arr: List[int], k: int) -> int:
        if k==1:
            return self.kadansAlgo(arr)%(10**9 + 7)
        elif sum(arr)<0:
            return self.kadansAlgoTwice(arr)%(10**9 + 7)
        else:
            return (self.kadansAlgoTwice(arr) + (k-2)*sum(arr))%(10**9 +
7)



    def kadansAlgo(self,arr):
        currSum = 0
        maxSum = 0
```

```
        for i in range(len(arr)):
            currSum = max(currSum+arr[i],arr[i])
            maxSum = max(maxSum,currSum)
        return maxSum

    def kadansAlgoTwice(self,arr):
        arr = arr + arr
        currSum = 0
        maxSum = 0
        for i in range(len(arr)):
            currSum = max(currSum+arr[i],arr[i])
            maxSum = max(maxSum,currSum)
        return maxSum
```