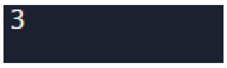
27. Given a circular integer array nums of length n, return the maximum possible sum of a non-empty subarray of nums.A circular array means the end of the array connects to the beginning of the array. Formally, the next element of nums[i] is nums[(i + 1) % n] and the previous element of nums[i] is nums[(i - 1 + n) % n].A subarray may only include each element of the fixed buffer nums at most once. Formally, for a subarray nums[i], nums[i + 1], ..., nums[j], there does not exist i <= k1, k2 <= j with k1 % n == k2 % n.

PROGRAM:

```
def maxSubarraySumCircular(nums):
    def kadane(nums):
        max_sum = float('-inf')
        current_sum = 0
        for num in nums:
            current_sum = max(num, current_sum + num)
            max_sum = max(max_sum, current_sum)
        return max_sum
    max_sum_within = kadane(nums)
    total_sum = sum(nums)
    negated_nums = [-num for num in nums]
    min_sum_within = kadane(negated_nums)
    max_sum_wrap = total_sum + min_sum_within  # Maximum sum wrapping around
    if max_sum_wrap == 0:
        return max_sum_within
    return max(max_sum_within, max_sum_wrap)
nums = [1,-2,3,-2]
print(maxSubarraySumCircular(nums))
```

OUTPUT: 

```
3
```

TIME COMPLEXITY: O(n)