Leetcode May Challenge DAY: 15

1. Python

```
class Solution:
    def maxSubarraySumCircular(self, xs: List[int]) -> int:
        if all(x < 0 for x in xs):
            return max(xs)

        def maxSubarraySum(xs):
            maxSum, runningMax = -float("inf"), 0
            for v in xs:
                runningMax = max(v, runningMax + v)
                maxSum = max(maxSum, runningMax)
            return maxSum

        def minSubarraySum(xs):
            minSum, runningMin = float("inf"), 0
            for v in xs:
                runningMin = min(v, runningMin + v)
                minSum = min(minSum, runningMin)
            return minSum

        maxWrapedSubarraySum = sum(xs) - minSubarraySum(xs)
        return (max(maxWrapedSubarraySum, maxSubarraySum(xs)))
```

2. C++

```
class Solution {
public:
    int maxSubarraySumCircular(vector<int>& A) {

        int n = A.size();
        int nres = INT_MIN, pres = INT_MIN, tsum = 0, psum = 0, nsum = 0;

        for (int i = 0; i < A.size(); i++) {
            tsum += A[i]; nsum += -A[i]; psum += A[i];
            nres = max(nres, nsum);
            pres = max(pres, psum);
            if (nsum < 0) nsum = 0;
            if (psum < 0) psum = 0;
        }

        return max(tsum+nres == 0 ? INT_MIN : tsum+nres, pres);
    }
};
```

3. JAVA

```
class Solution {
    public int maxSubarraySumCircular(int[] A) {
        int n = A.length;
        int[] sum = new int[2*n];
        for(int i=1;i<2*n;i++){
            sum[i]=sum[i-1]+A[(i-1)%n];
        }
        int res = Integer.MIN_VALUE;
        Deque<Integer> deque = new LinkedList<>();
        deque.offerLast(0);
        for(int i=1;i<2*n;i++){
            if (!deque.isEmpty() && deque.peekFirst()<i-n) deque.pollFirst();
            res=Math.max(res,sum[i]-sum[deque.peekFirst()]);
            while(!deque.isEmpty() && sum[deque.peekLast()]>=sum[i]) deque.pollLast();
            deque.offerLast(i);
        }
        return res;
    }
}
```