

## Recap: Problem definition: details

**Problem definition:** Given a sequence of n integers  $A_1, A_2, ..., A_n$ ,  $(n \ge 0$ , some numbers may be negative). Find the maximum sum of a consecutive subsequence

 $S_{i,j} = \sum_{k=i}^{j} A_k$ . If i>j or if all of the numbers  $A_i, \dots, A_j$  are negative, we define the

sum to be zero. We use the abbreviation  $A_{i,j}$  to stand for  $A_i, \ldots, A_j$ . Note that  $S_{i,j}$  is the sum of the numbers in  $A_{i,j}$ .

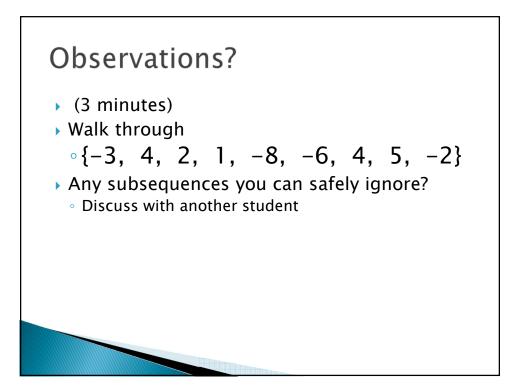
Examples (from the DS book): What is  $S_{2,4}$ ? {-2, 11, -4, 13, -5, 2} (maximum subsequence sum is 20), {1, -3, 4, -2, -1, 6} (maximum subsequence sum is 7).

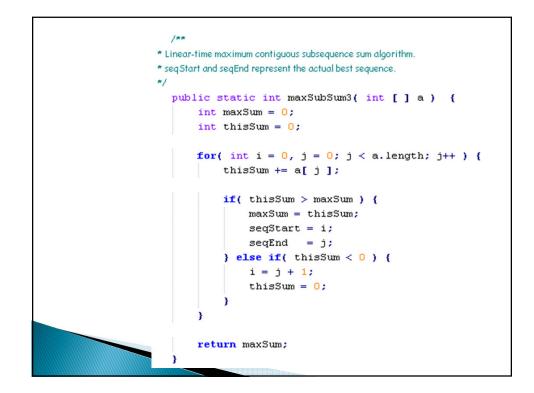
When we store the sequence in a Java array, the subscripts begin with 0.

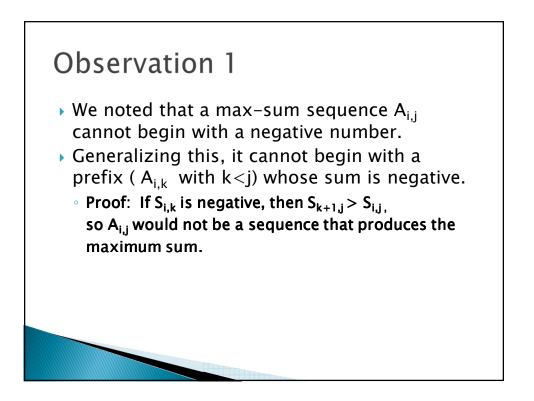
When we refer to the problem in the abstract here, the subscripts will begin with 1. (because the Weiss DS book does it this way, as do most mathematicians)

```
Recap: Eliminate the most obvious
inefficiency, get Θ(N<sup>2</sup>)

for( int i = 0; i < a.length; i++ ) {
    int thisSum = 0;
    for( int j = i; j < a.length; j++ ) {
        thisSum += a[ j ];
        if( thisSum > maxSum ) {
            maxSum = thisSum;
            segEnd = j;
        }
        Can we do
        even better?
```







## Observation 2

- All contiguous subsequences that border the maximum contiguous subsequence must have negative (or zero) sums.
- Proof: If one of them had a positive sum, we could simply append (or prepend) it to get a sum that is larger than the maximum. Impossible!

## **Observation 3** For any i, Let $A_{i,j}$ be the *first* subsequence (starting with $A_i$ ) whose sum $S_{i,j}$ is negative. Then for any p and q such that $i \le p \le j$ and $p \le q$ , either • $A_{p,q}$ is not a maximum contiguous subsequence of $A_1, A_2, ..., A_n$ , or

 A<sub>p,q</sub> has the same sum as a maximum contiguous subsequence that has been previously observed.

