A non-empty zero-indexed array A consisting of N integers is given.

A triplet (X, Y, Z), such that 0 ≤ X < Y < Z < N, is called a *double slice*.

The *sum* of double slice (X, Y, Z) is the total of A[X + 1] + A[X + 2] + ... + A[Y − 1] + A[Y + 1] + A[Y + 2] + ... + A[Z − 1].

For example, array A such that:

A[0] = 3

A[1] = 2

A[2] = 6

A[3] = -1

A[4] = 4

A[5] = 5

A[6] = -1

A[7] = 2

contains the following example double slices:

* double slice (0, 3, 6), sum is 2 + 6 + 4 + 5 = 17,
* double slice (0, 3, 7), sum is 2 + 6 + 4 + 5 − 1 = 16,
* double slice (3, 4, 5), sum is 0.

The goal is to find the maximal sum of any double slice.

Write a function:

class Solution { public int solution(int[] A); }

that, given a non-empty zero-indexed array A consisting of N integers, returns the maximal sum of any double slice.

For example, given:

A[0] = 3

A[1] = 2

A[2] = 6

A[3] = -1

A[4] = 4

A[5] = 5

A[6] = -1

A[7] = 2

the function should return 17, because no double slice of array A has a sum of greater than 17.

Assume that:

* N is an integer within the range [3..100,000];
* each element of array A is an integer within the range [−10,000..10,000].

Complexity:

* expected worst-case time complexity is O(N);
* expected worst-case space complexity is O(N), beyond input storage (not counting the storage required for input arguments).

Elements of input arrays can be modified.