CS19001/CS19002 PROGRAMMING AND DATA STRUCTURES LABORATORY
Assignment No: 6
Last Date of Submission: 16–March–2015

Let $A[\,]$ be an array of n integers. We assume that all the members of $A[\,]$ are non-zero, and that $A[\,]$ consists of any mix of positive and negative integers. First, ask the user to supply the value of n, and then read (or randomly generate) the array elements. Print the array $A[\,]$.

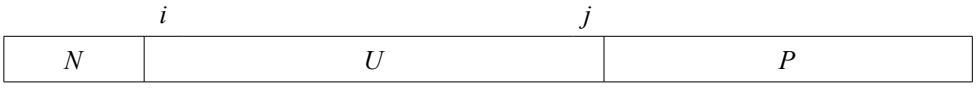
In the rest of this assignment, you solve some computational problems pertaining to this array.

Part 1

Your first task is to separate the positive members of $A[\,]$ from its negative members. Write a function to populate a second array $B[\,]$, in which all the negative members of $A[\,]$ appear before all the positive members of $A[\,]$. The order of the negative (or positive) elements in $B[\,]$ need not have any relation with the order of the elements in $A[\,]$. Maintain two indices i and j in $B[\,]$ for writing from the beginning and from the end. Both i and j move from a boundary (start or end) of $B[\,]$, and advance toward the interior of $B[\,]$. Make a pass through the array $A[\,]$, and whenever a negative member is encountered, it is added at the index i, and whenever a positive member is encountered, it is added at the index j. After the entire array $A[\,]$ is read, print the array $B[\,]$.

Part 2

In this part, you solve the same problem as in Part 1, but this time you are not allowed to use any extra array (B). You modify $A[\,]$ itself so that its negative members come before its positive members. To do this, you maintain two indices i and j in $A[\,]$. The elements $A[0 \ldots i-1]$ are already *known* to be negative. Likewise, the elements $A[j+1 \ldots n-1]$ are already *known* to be positive. The part $A[i \ldots j]$ is unprocessed.



In each iteration, you look at the element $A[i]$. If this is negative, let the region N grow by one cell. If $A[i]$ is positive, swap it with $A[j]$, and let the region P grow by one cell. After exactly n iterations, all elements are processed (that is, U shrinks to an empty block), and N contains all the negative integers of $A[\,]$ sitting before the block P consisting of all the positive integers of $A[\,]$. Write a function to implement this algorithm. Print the array $A[\,]$ after the function returns.

Part 3

In this part, write a function that takes an array $C[\,]$ and its size n as the arguments. It is assumed that all the negative elements of $C[\,]$ appear before all the positive elements of $C[\,]$. Your task is to fond out whether there exist indices i and j (with $i < j$) in $C[\,]$ such that

$$C[i] + C[i+1] + \ldots + C[j] = 0.$$

Clearly, such indices, if existent, define a block in $C[\,]$ that straddles the N-P boundary in $C[\,]$. First, locate the N-P boundary in $C[\,]$. Maintain a sum. Keep on including elements from N or P starting from the N-P boundary and moving outward. If the sum is positive, include the next negative element (from N) in the sum. If the sum is negative, include the next positive element (from P) in the sum. If the sum ever reaches the value zero, you are done.

Call this function on both the arrays $A[\,]$ (prepared in the correct format in Part 2) and $B[\,]$ (available from Part 1).

Submit a single C file solving all the three parts.

Sample Output

```
n = 15
+++ Initial array A:
 -4 -10   2 -12  10  -3   4  -1   1  -2  -2  17  -6  13  -7
+++ New array B:
 -4 -10 -12  -3  -1  -2  -2  -6  -7  13  17   1   4  10   2
+++ Modified array A:
 -4 -10  -7 -12  -6  -3  -2  -1  -2   1  17   4  13  10   2
Array A: i = 0, j = 14
 -4 -10  -7 -12  -6  -3  -2  -1  -2   1  17   4  13  10   2
Array B: i = 7, j = 9
 -6  -7  13
```