An array A consisting of N integers is given. Rotation of the array means that each element is shifted right by one index, and the last element of the array is moved to the first place. For example, the rotation of array A = [3, 8, 9, 7, 6] is [6, 3, 8, 9, 7] (elements are shifted right by one index and 6 is moved to the first place).

The goal is to rotate array A K times; that is, each element of A will be shifted to the right K times.

Write a function:

def solution(A, K)

that, given an array A consisting of N integers and an integer K, returns the array A rotated K times.

For example, given

A = [3, 8, 9, 7, 6] K = 3

the function should return [9, 7, 6, 3, 8]. Three rotations were made:

[3, 8, 9, 7, 6] -> [6, 3, 8, 9, 7] [6, 3, 8, 9, 7] -> [7, 6, 3, 8, 9] [7, 6, 3, 8, 9] -> [9, 7, 6, 3, 8]

For another example, given

A = [0, 0, 0] K = 1

the function should return [0, 0, 0]

Given

A = [1, 2, 3, 4] K = 4

the function should return [1, 2, 3, 4]

Assume that:

* N and K are integers within the range [0..100];
* each element of array A is an integer within the range [−1,000..1,000].

In your solution, focus on **correctness**. The performance of your solution will not be the focus of the assessment.

def solution(A, K):

if (len(A) == 0) or (K == 0) or (len(A)==K):

return A

elif K>len(A):

K = K % len(A)

A[:] = A[len(A)-K:] + A[:len(A)-K]

return A

else:

A[:] = A[len(A)-K:] + A[:len(A)-K]

return A

or

def solution(A, K):

if len(A) != 0:

K = K % len(A)

A[:] = A[ len(A) - K: ] + A[ : len(A) - K ]

return A

or

def solution(A, K):

return A[len(A)-K::]+ A[:len(A)-K]