The algorithm:

Assumption: 1.when the unsorted array has length one, we consider that single number is smaller than the integer in the adjacent location.

2. A contains finite number of integers.

def GetPosition(A):

int n lengt(A)

if n=1: // there is the first if statement.

return 1

i // i is greater or equal to 1 at this time.

j 1

if A[1]<A[2]: // there is the first if-else statement.

return A[1]

else if A[n]<A[n-1]:

return A[n]

else: // n is greater than 2 at this time.

While True:

if A[i]<A[i+1]:

j

if i j

return i

else:

i j

else: // A[i]>A[i+1]

j

if i j:

return i

else:

ij

3. As a precondition, A is an unsorted array of finite number of distinct integers, and the length of A is greater or equal to one, which means A is not empty, so A contains a smallest integer, which is smaller than the integers in the adjacent locations. So, the algorithm we want must return if the precondition is satisfied.

My function GetPosition(A) has three cases:

1. When the unsorted array has length one, we consider that single number is smaller than the integer in the adjacent location. So, we directly return 1.
2. When the unsorted array has length two, then either A[1] or A[2] is the smallest number. So we