Sort on Pancake

*Description:*

Assume that there have n different Pancakes with different sizes, so sort according to: the small pancake is on top and large one is underneath by one hand. The normal method is to reverse several pancakes together and makes the large pancake on the top and small pancake below.

*Given Pancake Array as below:*

[3, 2, 1, 6, 5, 4, 9, 8, 7, 0] with size of 10.

*Solution One:*

*Rule:*

Revert several pancakes together and sort the array as [0, 1, 2, 3, 4, 5, 6, 7, 8, 9].

*First Round:*

Step One - [9, 4, 5, 6, 1, 2, 3, 8, 7, 0] (Revert integers from 3 to 9.)

Step Two - [0, 7, 8, 3, 2, 1, 6, 5, 4, 9] (Revert integers from 9 to 0.)

Number 9 is at the right position now.

*Second Round:*

Step One - [8, 7, 0, 3, 2, 1, 6, 5, 4, 9] (Revert integers from 0 to 8.)

Step Two - [4, 5, 6, 1, 2, 3, 0, 7, 8, 9] (Revert integers from 8 to 4.)

Number 8 is at the right position now.

*Third Round:*

No Changes - Number 7 is at the right position now.

*Forth Round:*

Step One - [6, 5, 4, 1, 2, 3, 0, 7, 8, 9] (Revert integers from 4 to 6.)

Step Two - [0, 3, 2, 1, 4, 5, 6, 7, 8, 9] (Revert integers from 6 to 0.)

Number 6 is at the right position now.

*Fifth Round:*

No Changes - Number 5 is at the right position now.

*Sixth Round:*

No Changes - Number 4 is at the right position now.

*Seventh Round:*

Step One - [3, 0, 2, 1, 4, 5, 6, 7, 8, 9] (Revert integers from 0 to 3.)

Step Two - [1, 2, 0, 3, 4, 5, 6, 7, 8, 9] (Revert integers from 3 to 1.)

Number 3 is at the right position now.

*Eighth Round:*

Step One - [2, 1, 0, 3, 4, 5, 6, 7, 8, 9] (Revert integers from 1 to 2.)

Step Two - [0, 1, 2, 3, 4, 5, 6, 7, 8, 9] (Revert integers from 2 to 0.)

Number 2 is at the right position now.  
*Ninth Round:*

No Changes - Number 1 is at the right position now.

*Tenth Round:*

No Changes - Number 0 is at the right position now.

*Pseudo code - Necessity:*

*// Return the Biggest Integer among Array A[ i, ..., j ] and the Subscript of Array.*

Int Search\_Biggest\_Number\_Array ( Array A, int i, int j )

{

int maximum = A [ i ];

For ( temp from i to j )

{

If ( A [ temp ] <= maximum )

{

maximum = A [ temp ];

}

}

}

*// Exchange all elements of the Array A [ i, ..., j ], swap A [ i ] with A [ j ], A [ i + 1 ] with A [ j - 1 ] until last element.*

Array Swap\_Array ( Array A, int i, int j )

{

while ( i != j )

{

int swap = A [ i ];

A [ i ] = A [ j ];

A [ j ] = swap;

i ++;

j --;

}

Return A;

}

*// Main function, which is used to find the ith biggest value among Array [ Start, ..., End ].*

void Sort\_Pancake ( Array, int i, int j )

{

For ( i from End to Start )

{

int index = Search\_Biggest\_Number\_Array(Array, 0, i);

If ( index == i )

{

Element Array[i] is at the right position.

}

Else

{

*// Exchange the Biggest Element to the first place of the Array.*

Swap\_Array( Array[0, ..., index], 0, index );

*// Make sure that the value Array[i] is at the right place.*

Swap\_Array( Array[0, ... i], 0, i );

}

}

}

*Solution Two:*