In this Assignment we will write an assembly program that uses loops with indirect or indexed addressing to **permutate (re-arranges) an integer array.** The following example shows how an array is re-arranged based on the permutation:

Array before permutation:

Position/Index:	0	1	2	3	4	5	6	7	8	9
	12	9	54	20	3	38	72	17	91	25

Permutation:

8	9	5	7	2	6	0	4	3	1

Array after permutation:

91	25	38	17	54	72	12	3	20	9

Assuming Permutation[i] = j, the permutation operation re-arranges the original array such that the j^{th} element of the original array is now stored on the i^{th} position (i.e., index). For instance, in the above example Permutation[2] is 5. After permutation, the 5^{th} element in the original array (i.e., 38) is now the 2^{nd} element of the array.

In your program you are allowed to define only **two arrays** – one for the array and the other for the permutation. **Do not copy the array elements to any other array!** The runtime screen shot can be as follows (you can use different numbers):

```
Welcome! This program permutates an array of any size.

Please enter the size of the array: 10

Please enter the 10 elements of your array:
12      9      54      20      3      38      72      17      91      25

Please enter the permutation:
8       3       5       9       2       6       0       4       1       7
```

The array after permutation is:
91 20 38 25 54 72 12 3 9 17

Hint:

The following algorithm gives one possible solution to the problem:

```
Enter array size as SIZE;
for i from 0 to (SIZE – 2)
        Array[i]  <->  Array[Permutation[i]];
        for j from (i+1) to (SIZE – 1)
                if (i == permutation[j])
                        permutation[i] <-> permutation[j];
                        break;
```

NOTE: You should test more than one permutation, for instance in a size-10 array a permutation of 9,8,7,6,5,4,3,2,1,0 should reverse the array!

Requirements:

1. **Submit your source code** (.asm file) which should run correctly.
2. Necessary **comments** are needed in your code.
3. **Turn in a report**. The report should include three parts: *Introduction*, *Implementation*, and *Summary*. The introduction briefly describes the purpose of this assignment. The implementation part gives a detailed description on how you implemented the task, including the runtime screen shots, as well as necessary discussions. The summary concludes the lab, e.g., what you've learned from this lab.

Please submit through blackboard!