Solution Review: Sorting an Array

In this review, solution of the challenge 'Sorting an Array' from the previous lesson is provided.

We'll cover the following

Solution

```
class SortArr {
    public static void sortAsc(int[] arr) {
        int temp = 0; //a variable to store temporary value while swapping

        for (int i = 0; i < arr.length-1; i++) //for loop to hold the current element to be compar
        {
            for (int j = i + 1; j < arr.length; j++) //for loop to iterate over the other elements
            { //to get them compared with the current element
                if (arr[i] > arr[j]) //if any of the higher index element is smaller than
                { //the current element
                    temp = arr[i]; //store the current element to temp
                    arr[i] = arr[j]; //store the smaller element to the lower index position
                    arr[j] = temp; //store the current element to greater index position
                }
            }
        }
    }
    public static void main( String args[] ) {
        int array[] = {56, 9, 45, 108, 567, 21};
        System.out.println( "Array values before sorting:" );
        for (int i =0 ; i < array.length; i++){
            System.out.print(array[i]+ "  ");
        }
        System.out.println();
        sortAsc(array);
         System.out.println( "Array values after sorting:" );
        for (int i =0 ; i < array.length; i++){
            System.out.print(array[i]+ "  ");
        }
}
}
```

How does the above code work?

In the above code, we have implemented nested `for` loops and an `int` variable named *temp* to store a value which is going to be swapped.

- **Line 5 - 8:**

The *control variable* `i,` from the outer `for` loop is used as an index to access the array elements one by one. The currently accessed element `arr[i]` (using the outer for loop) is then compared with all the elements from index `j=i+1` onwards using the inner loop's control variable `j` as an index to access the next elements of the array.

> Notice that for each value of `i` the inner loop iterates from `i+1` to the end of the array.

- **Line 9:** At each iteration of the inner loop, a condition is being checked i.e.

```
if(arr[i] > arr[j])
```

- **Line 11 - 13:** The above condition checks:
    - if any of the *higher index* element has **lesser** integer value than the *lower index* element.
    - If it is so, the element with **greater** value is stored into `temp` and then the element with **greater** value at the *lower index* `arr[i]` will be replaced with the element with **lesser** value at the *higher index* `arr[j]`.
    - After this, the **greater** value in the `temp`stored at the *higher index*. This *swapping* is repeated till all the elements are sorted in ascending order i.e. the element with the least `int` value at the start of the array and the element with the maximum value at the end of the array.

In the next lesson, we will try to print the values stored in a matrix.