**ARRAY TEST FOR REPEATED ELEMENTS.**

**How it works?**

We define a threshold for percentage of repetitions in the array. If the percentage of repetition is above the threshold percentage we define it having the pattern as ‘**REPEATED**’, which means that it has many elements that are repeated and the sorting algorithm that are good for array with repeated elements can be applied. Examples of algorithms that are good for this type of array includes: Quick sort, Shell Sort, Heap Sort.

**IMPLEMENTATION:**

**Array generation:**

1. Get the array using ‘***DataGenerator****’* class. Here I have saved in the test array-‘***TArr’***.
2. ***‘TestReps’*** is the function that takes the array and its size as input.

**Functioning of ‘TestReps()’:**

Declarations:

1. ***‘threshold’*** is used for storing the threshold percentage.
2. ***‘count’*** acts as the counter for the number of repetitions encountered.
3. ***‘percentage’*** is used to store the percentage of repetitions encountered.
4. ***‘length’***  is the array size.

**Working:**

We create a HASHSET to store the numbers of the array. Hash sets have the property that it doesn’t contain duplicate values. When we try to add a value that is already in the hash set, it is overwritten and the add() function returns **false** since nothing was **appended**.

Whenever the add() function returns ‘false’ we increment the *‘****count’*** by 1 and add() function will return false every time it encounters a duplicate value from the array. Finally, the *‘****count’*** will give the number of repetitions in the array.

Percentage of repetitions= (Total repetitions/Total Numbers in the array)\*100

**[*‘length’* is the length of array]**

**Percentage = (count/length)\*100**

Now we compare the percentage obtained by the above formula with the threshold percentage.

Percentage > Threshold :  **Pattern of the array is “REPEATED”.**

Percentage < Threshold :  **Pattern of the array is “NOT REPEATED”**