//Thi Thanh Tra Kieu #261066512

1. Merge Sort

Input: array type int not sorted
Output: array type int sorted

```
function mergeSort(array){
    declare and initialize n = length of array;                          =>O(1)+O(2^1) + O(2^2)+...+ O(2^k)
    if(n <=1) return;                                                    =>O(1
    declare and initialize mid = n/2;
    declare and initialize leftArray with length = mid;                  =>O(1)+O(2^1) + O(2^2)+...+ O(2^k)
    declare and initialize rightArray with length = n - mid;             =>O(1)+O(2^1) + O(2^2)+...+ O(2^k)
    declare and initialize type int i =0, j = 0;                         =>O(1)+O(2^1) + O(2^2)+...+ O(2^k)

    while(i <n){
        if(i < mid)
            leftArray at i = array at i;                                 =>O(n/2)+O(n/4)+O(n/8)+...+O(2)
            increment i by 1;                                            =>O(n/2)+O(n/4)+O(n/8)+...+O(2)
        else
            rightArray at j = array at i;                                =>O(n/2)+O(n/4)+O(n/8)+...+O(2)
            increment i by 1;                                            =>O(n/2)+O(n/4)+O(n/8)+...+O(2)
            increment j by 1;                                            =>O(n/2)+O(n/4)+O(n/8)+...+O(2)
    }
    use recursive mergeSort(leftArray);
    use recursive mergeSort(rightArray);
    call funtion merge(leftArray, rightArray, array);                     =>O(n/2)+O(n/4)+O(n/8)+...+O(2)
}
```

Input: 3 arrays type int: leftArray, rightArray, array
Output: array (the result from merging leftArray and rightArray)

```
funtion merge(leftArray, rightArray, array){
    declare and initialize index, leftIndex, rightIndex = 0;
    declare and initialize length of leftArray = array.length/2;
    decalre and initialize length of rightArray = array.length - leftSide;

    while(leftIndex < length of leftArray && rightIndex < length of rightArray){
        if(leftArray at leftIndex < rightArray at rightIndex)
            array at index = leftArray at leftIndex;
            increment index by 1;
            increment leftIndex by 1;
        else
            array at index = rightArray at rightIndex;
            increment index by 1;
            increment rightIndex by 1;
    }

    while(leftIndex < length of leftArray)
        array at index = leftArray at leftIndex;
            increment index by 1;
            increment leftIndex by 1;
    while(rightIndex < length of rightArray){
        array at index = rightArray at rightIndex;
            increment index by 1;
            increment rightIndex by 1;
}
```

T(n) = O(1) + 4*(O(1)+O(2^1) + O(2^2)+...+ O(2^k)) + 6*(O(n/2)+O(n/4)+O(n/8)+...+O(2)) = O(nlogn).