```
1  // sample of arrays to sort
2  const arrayRandom = [9, 2, 5, 6, 4, 3, 7, 10, 1, 8];
3  const arrayOrdered = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
4  const arrayReversed = [10, 9, 8, 7, 6, 5, 4, 3, 2, 1];
5
6  let countOuter = 0;
7  let countInner = 0;
8  let countSwap = 0;
9
10 function resetCounters() {
11   countOuter = 0;
12   countInner = 0;
13   countSwap = 0;
14 }
15
16 // top-down implementation
17 function mergeSortTopDown(array) {
18   countOuter++;
19   if(array.length < 2) {
20     return array;
21   }
22
23   const middle = Math.floor(array.length / 2);
24   const left = array.slice(0, middle);
25   const right = array.slice(middle);
26
27   return mergeTopDown(mergeSortTopDown(left), mergeSortTopDown(right));
28 }
29
30 function mergeTopDown(left, right) {
31   const array = [];
32
33   while(left.length && right.length) {
34     countInner++;
35     if(left[0] < right[0]) {
36       array.push(left.shift());
37     } else {
38       array.push(right.shift());
39     }
40   }
41   return array.concat(left.slice()).concat(right.slice());
42 }
43
44 mergeSortTopDown(arrayRandom.slice()); // => outer: 19 inner: 24 swap: 0
45 console.log('outer:', countOuter, 'inner:', countInner, 'swap:', countSwap);
46 resetCounters();
47
48 mergeSortTopDown(arrayOrdered.slice()); // => outer: 19 inner: 15 swap: 0
49 console.log('outer:', countOuter, 'inner:', countInner, 'swap:', countSwap);
50 resetCounters();
51
52 mergeSortTopDown(arrayReversed.slice()); // => outer: 19 inner: 19 swap: 0
53 console.log('outer:', countOuter, 'inner:', countInner, 'swap:', countSwap);
54 resetCounters();
55
56 // bottom-up implementation
57 function mergeSortBottomUp(array) {
58   let step = 1;
59   while (step < array.length) {
60     countOuter++;
```

```
    let left = 0;
    while (left + step < array.length) {
      countInner++;
      mergeBottomUp(array, left, step);
      left += step * 2;
    }
    step *= 2;
  }
  return array;
}
function mergeBottomUp(array, left, step) {
  const right = left + step;
  const end = Math.min(left + step * 2 - 1, array.length - 1);
  let leftMoving = left;
  let rightMoving = right;
  const temp = [];

  for (let i = left; i <= end; i++) {
    if ((array[leftMoving] <= array[rightMoving] || rightMoving > end) &&
        leftMoving < right) {
      temp[i] = array[leftMoving];
      leftMoving++;
    } else {
      temp[i] = array[rightMoving];
      rightMoving++;
    }
  }

  for (let j = left; j <= end; j++) {
    countSwap++;
    array[j] = temp[j];
  }
}

mergeSortBottomUp(arrayRandom.slice()); // => outer: 4 inner: 9 swap: 36
console.log('outer:', countOuter, 'inner:', countInner, 'swap:', countSwap);
resetCounters();

mergeSortBottomUp(arrayOrdered.slice()); // => outer: 4 inner: 9 swap: 36
console.log('outer:', countOuter, 'inner:', countInner, 'swap:', countSwap);
resetCounters();

mergeSortBottomUp(arrayReversed.slice()); // => outer: 4 inner: 9 swap: 36
console.log('outer:', countOuter, 'inner:', countInner, 'swap:', countSwap);
resetCounters();

```