```javascript
function heapSort (arr) {
    buildMaxHeap(arr)

    for (let i = arr.length - 1; i > 0; i--) {
        [arr[i], arr[0]] = [arr[0], arr[i]]

        heapify(arr, i, 0)
    }

    return arr
}

function buildMaxHeap(arr) {
    let n = arr.length
    for (let i = Math.floor(n/2) - 1; i >= 0; i--) {
        heapify(arr, n, i)
    }
}

function heapify(arr, n, i) {
    let largest = i // is a pointer, moves down the tree
    left = i * 2 + 1
    right = i * 2 + 2

    if (left < n && arr[left] > arr[largest]) largest = left

    if (right < n && arr[right] > arr[largest]) largest = right

    if (largest !== i) {
        [arr[largest], arr[i]] = [arr[i], arr[largest]]

        heapify(arr, n, largest)
    }
}

console.log(heapSort([1,5,3,1]))
```

the buildMaxHeap process happens down to up, we start from the last parent node, it will compare the largest all the way till the top.

Once we get a max heap, we now swap with the last index in the array. Now the root or first element of the array is compared with an already max heap so we just have to perform heap process once to get the largest on top. Now we perform this consecutively till the bottom nodes to get the max value as the first element at each step.