Coding Question-6

move zeros to the end

Interviewquestion-56

JS

Follow on in

@Duvvuru Kishore

Move zeros to the end

Input = [1, 0, 2, 0, 3, 0, 4]

Output: [1, 2, 3, 4, 0, 0, 0]

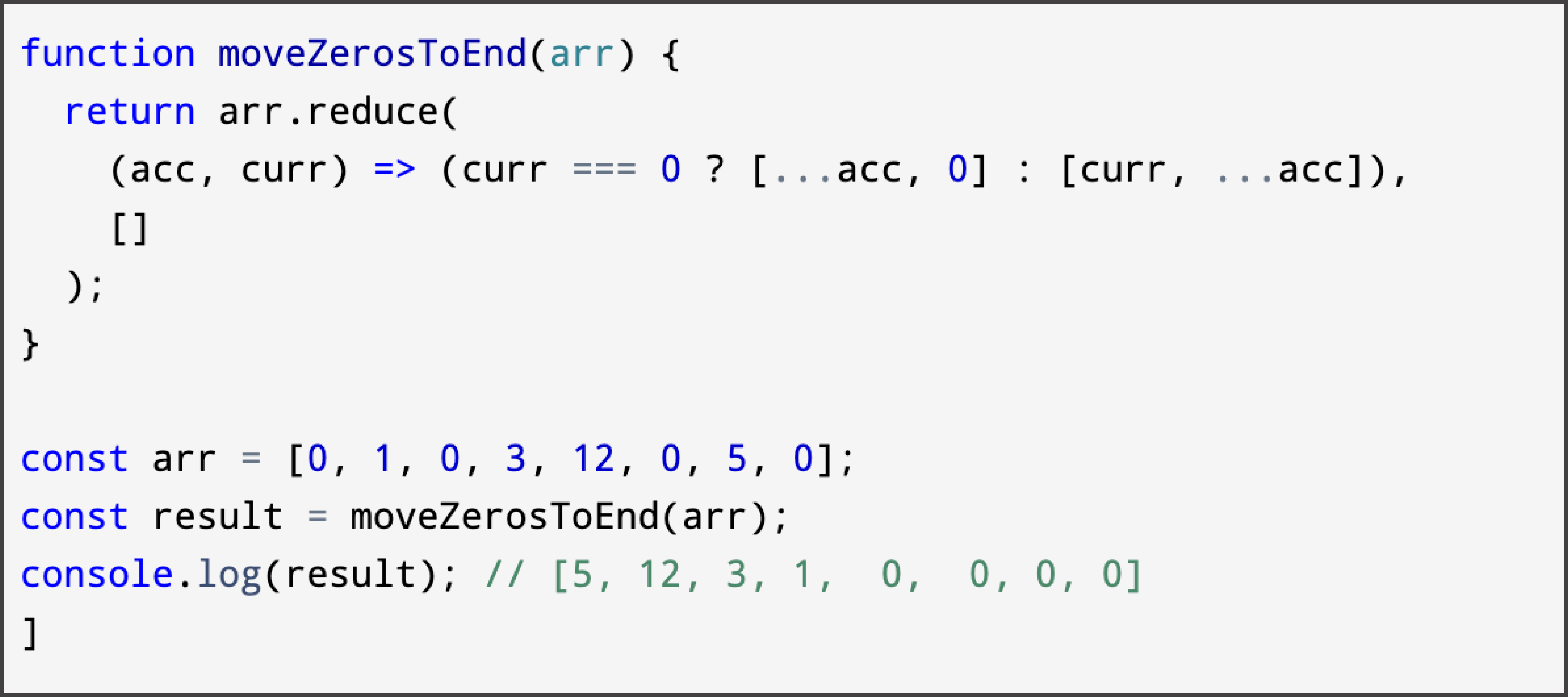
approach 👉

Approach1

```javascript
function moveZerosToEnd(arr) {
  const nonZeros = arr.filter(item => item !== 0);
  const zeros = arr.filter(item => item === 0);
  return nonZeros.concat(zeros);
}

const array = [1, 0, 2, 0, 3, 0, 4];
const result = moveZerosToEnd(array);
console.log(result); // Output: [1, 2, 3, 4, 0, 0, 0]
```

I truly believe that this code doesn’t need explanation

Appoarch 2



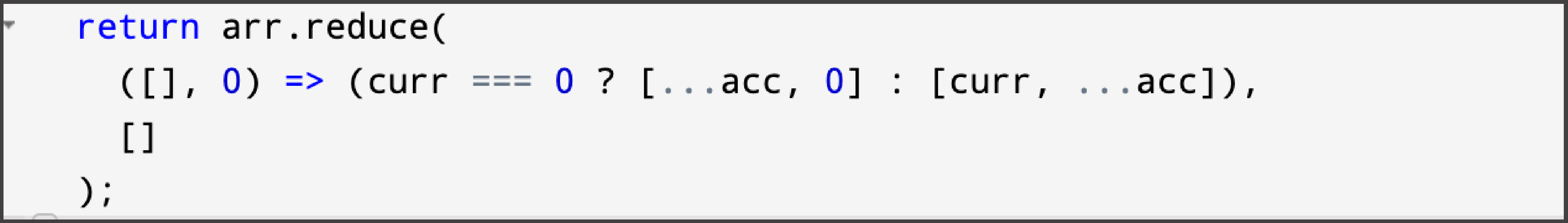
```javascript
function moveZerosToEnd(arr) {
  return arr.reduce(
    (acc, curr) => (curr === 0 ? [...acc, 0] : [curr, ...acc]),
    []
  );
}

const arr = [0, 1, 0, 3, 12, 0, 5, 0];
const result = moveZerosToEnd(arr);
console.log(result); // [5, 12, 3, 1,  0,  0, 0, 0]
]
```

Don’t worry ,
explanation is there in next slide

acc (accumulator): [ ] (an empty array)
curr (current element): 0(as of array input)

Iteration 1:

curr is 0 (input we receive) , so it will pass the condition curr===0 and moves to [...acc,0] ,now acc is empty[] and curr is 0 . now the result is [0]



```
return arr.reduce(
  ([], 0) => (curr === 0 ? [...acc, 0] : [curr, ...acc]),
  []
);
```

Iteration 2:

curr is 1 (from input we receive) , so it will fail the condition curr===0 and moves to [curr,...acc] ,now curr is 1 and acc is 0. now the result is [1,0]

```
return arr.reduce(
  ([0], 1) => (curr === 0 ? [...acc, 0] : [1, 0]),
  []
);
```

Iteration 3

curr is 0 (from input we receive) , so it will pass the condition curr===0 and moves to [...acc,0] ,now acc is [1,0]. now the result is [1,0,0](last zero is added from [...acc,0])

```
return arr.reduce(
  ([1,0], 0) => (curr === 0 ? [1,0, 0] : [curr, ...acc]),
  []
);
```

Iteration 4

curr is 3 (from input we receive) , so it will fail the condition curr===0 and moves to [curr,...acc] ,now curr is 3 and acc is [1,0,0]. now the result is [3,1,0,0]

```
return arr.reduce(
  ([1,0,0], 3) => (curr === 0 ? [...acc, 0] : [3, 1,0,0]),
  []
);
```

Iteration 5

curr is 12 (from input we receive) , so it will fail the condition curr===0 and moves to [curr,...acc] ,now curr is 3 and acc is [3,1,0,0]. now the result is [12,3,1,0,0]

```
return arr.reduce(
  ([3,1,0,0], 12) => (curr === 0 ? [...acc, 0] : [12, 3,1,0,0]),
  []
);
```

In the same way code completes the over all iterations and moves zeros to last

Follow on

@Duvvuru Kishore