**NAME:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ DATE:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

***Note:*** This exercise tests the understanding of following topics in ES6:

1. Let & Const
2. Template literals
3. Default parameters
4. Destructuring
5. Rest & Spread

# Let & Const

Following code uses ‘var’ instead of ‘const’ and ‘let’. Refactor the code to use new keywords. Be sure to consider whether the variable should be declared using ‘const’ or ‘let’ depending on whether the variable gets reassigned or not

var responses = [

{ code: 200, message: 'OK' },

{ code: 400, message: 'Bad request' },

{ code: 401, message: 'Not authorized' },

{ code: 500, message: 'Server error' }

];

var successCode = 200;

var slNo = 1;

var errorSummary = '';

for (var i = 0; i < responses.length; i++) {

if (responses[i].code !== successCode) {

errorSummary += '(' + slNo++ + ')' + ' ' + responses[i].message + '\n';

}

}

var totalErrors = slNo - 1;

console.log('Error Summary:');

console.log('-------------------');

console.log(errorSummary);

console.log('-------------------');

console.log('Total Errors: ' + totalErrors + ' / ' + responses.length);

console.log('-------------------');

# Template Literals

Refactor the previous code to include template literal syntax. Replace the occurrences of string concatenation with template literals.

# Default Parameters

Refactor the following code to use default function arguments

function sum(a, b) {

if (a === undefined) {

a = 0;

}

if (b === undefined) {

b = 0;

}

return a + b;

}

console.log(sum());

console.log(sum(20));

console.log(sum(5, 10));

function addOffset(style) {

if (!style) {

style = {};

}

style.offset = '10px';

return style;

}

console.log(addOffset());

console.log(addOffset({ length: 10 }));

# Destructuring

The 'customers' variable holds an array of arrays, where each array represents a single customer. Convert this array of arrays into an array of objects, where each object has the keys 'name', 'phone', 'email' and 'city' and assign the result to 'customersAsObject'. Use array destructuring and the map helper.

An array for a customer has the form:

[ name, phone, email, city ]

The resulting data structure should look something like the following:

const customersAsObject = [

{

name: 'Hari',

phone: '98860 98860',

email: 'hari@abc.com',

city: 'Mumbai'

}

]

const customers = [

['Hari', '98860-98860', 'hari@abc.com', 'Bengaluru'],

['Ram', '98860-98861', 'ram@abc.com', 'Chennai'],

['Shiv', '98860-98862', 'shiv@abc.com', 'Mumbai'],

['Krish', '98860-98863', 'krish@abc.com', 'Delhi']

];

const customersAsObject;

# Rest & Spread

Refactor the following function to use ‘rest’ and ‘spread’ operators. Make then function take any number of parameters from 2nd parameter.

function unshift(array, a, b, c) {

return [a, b, c].concat(array);

}

console.log(unshift([10, 20, 30], 'A', 'B', 'C'));