Advanced Constructs: Advanced Function Concepts-1 Assignment Solutions

Relevel
by Unacademy

Assignment Solutions

1. Write a program to multiply the value in the given array and return a result (use array functions)
Array = [2,3,4,5,6,7,8]

Solution:

```
const arr = [1,2,3,4,5];
let res = arr.reduce((prod, curr) => prod*curr,1);
console.log(res);
```

Output
120

To run the code live and test click here.

2. Write a JavaScript program to sort by id an array of JavaScript objects.
Object =[{
Id: 45,
Name: ‘ram’
},{
Id: 4,
Name: ‘raju’
},{
Id: 90,
Name: ‘kumar’
}]

Solution:

```
const data =[ {
   Id: 45,
   Name: 'ram'
  },{
   Id: 4,
   Name: 'raju'
  },{
   Id: 90,
   Name: 'kumar'
  }]

data.sort((a, b) => (a.Id > b.Id) ? 1 : -1)
console.log(data)
```

Output:

```
[{
  Id: 4,
  Name: "raju"
}, {
  Id: 45,
  Name: "ram"
}, {
  Id: 90,
  Name: "kumar"
}]
```

To run the code live and test click here.

3. Write a program to sort an integer array with custom number as reference.

Sample Input:

2356481790

0 1 2 3 9 5 6 2 8 1 9

Sample Output:

2 2 3 5 6 8 1 1 9 9 0

Explanation:

Our current integer is not 0123456789 it is 2356481790 as per the given input format.

So the integer should be sorted in the given order and the output is 2 2 3 5 6 8 1 1 9 9 0

Solution link - https://jsfiddle.net/saravananslb/oLpwmy4v/

Solution:

```
const customSort = (order, arr) => {
    const numberObj = {};
    const resArr = [];
    order.split('').map((item, index) => {
        numberObj[item] = {
            value: [],
            order: index
        };
    })
    arr.map(item => {
        numberObj[item].value.push(item);
    });
    const sortedObj = Object.values(numberObj).sort((a, b) => (a.order > b.order) ? 1 : -1);
    Object.values(sortedObj).map(item => {
        resArr.push(...item.value)
    });
    return resArr.join(' ');
}
```

```
const order = '2356481790';

const arr = [1, 2, 9, 3, 1, 5, 6, 2, 8, 9, 0];
console.log(customSort(order, arr));
```

To run the code live and test click here.

5. Ram got an assignment from his math teacher that he needs to get the number of 1's from 0 to a given binary number. But it is very hard to find so write a program which helps Ram to find it very quickly.

Input Format

The input should be only a binary number

Output Format

It should return number as output

Sample Input 1 :

100

Explanation: The binary number between 0 to 100 are

000

001

010

011

100

In the above series 5 1's are there.

Output :

5

Solution:

```
// Function to Find Number of Ones from 0 to N
const findBinaryOne = (binary) => {
    // Convert Binary into Integer and add 1
    const intNum = parseInt(binary, 2) + 1;
    // Split the Binary number into Array
    const bn = binary.split('');
    // Initialize and Declare Variable result as 0
    let result = 0;
    bn.map((elem, index) => {
        // Add Integer number and 2 power index + 1 and floor the result
        const divNum = Math.floor(intNum / (2 ** (index + 1)));
        // Divide intNum and 2 power index + 1 and get the remainder as
result
        const remNum = intNum % (2 ** (index + 1));
        // Multiply divNum with 2 power index
        const res = (divNum * (2 ** (index)));
        // If remainder - 2 power index greater than "0" result should be
        // remainder - 2 power index else "0"
        const normalize = (remNum - (2 ** index)) > 0 ? (remNum - (2 **
index)) : 0;
```

```
    // Add the result with res and normalize
        result += res + normalize;
    });
    return result;
}

console.log(findBinaryOne("100"));
```

To run the code live and test click here.

5. Ram is a college student and he has to submit the final year project which is password validator. Write a program to help ram to Validate the given password.

Constraint
Valid password
Password should contains uppercase letters
Password should contains lowercase letters
Password should contains symbols letters – !@#$%^&*()_+~?><:"{}|
Password should contains number

Invalid password
Password should not contains uppercase letters
Password should not contains lowercase letters
Password should not contains symbols letters – !@#$%^&*()_+~?><:"{}|
Password should not contains number
Input Format
N represent the number of passwords
The input should be string separated by new line

Output Format
password is valid for valid password
password is invalid for invalid password

Sample Input 1:
3
aahfD$jhk67,
ghfjhagsju,
hsjFG$**)&

Output:
aahfD$jhk67, is valid
ghfjhagsju, is invalid
hsjFG$**)& is invalid

```
const NUMBERS = '1234567890';
const SYMBOLS = '!@#$%^&*()_+~?><:"{}|';
const LOWER_ALPHABET = 'abcdefghijklmnopqrstuvwxyz';
const UPPER_ALPHABET = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ';

const passwordValidation = (arr) => {
    const validation = [];
    const numbers = NUMBERS.split('');
    const symbols = SYMBOLS.split('');
    const lowerCase = LOWER_ALPHABET.split('');
    const upperCase = UPPER_ALPHABET.split('');
    arr.map(item => {
        let isNumber, isLowerCase, isUpperCase, isSymbol = false;
        item.split('').map(pwd => {
            if (numbers.includes(pwd))
                isNumber = true;
            else if (symbols.includes(pwd))
                isSymbol = true;
            else if (lowerCase.includes(pwd))
                isLowerCase = true;
            else if (upperCase.includes(pwd))
                isUpperCase = true;
        })
        if (isNumber && isLowerCase && isUpperCase && isSymbol){
            validation.push(`${item} is valid`);
        }
        else {
            validation.push(`${item} is invalid`);
        }

    })
    return validation;
}

console.log(passwordValidation(["aahfD$jhk67,", "ghfjhagsju,",
"hsjFG$**)&"]))
```

To run the code live and test click here.

6. Write a function that converts an array of values from miles to kilometres using the map method. In the end, add the kilometres up in a new variable called "totalDistanceInKilometers" and return this variable.

Solution:

```
function convertMilesToKilometers(arr){
  return arr.map(mile => mile*1.609);
}
```

```
function totalDistance(acc, cur){
  return acc + cur;
}

const km = convertMilesToKilometers([1,2,3,4,5,6]);
console.log(km);
let totalDistanceInKilometers = km.reduce(totalDistance, 0);
console.log(totalDistanceInKilometers);
```

To run the code live and test click here.