Taylar Conway

Rest/ Spread Operator Exercise

**Refactor using rest operator and arrow function**

const filterOutOdds = (...*args*) => *args*.filter((*v*) => *v* % 2 === 0);

**Find Min**

const findMin = (...*args*) => Math.min(...*args*);

**Merge Objects**

const mergeObjects = (*object1*, *object2*) => ({ ...*object1*, ...*object2* });

**DoubleAndReturnArgs**

const doubleAndReturnArgs = (*arr*, ...*args*) => [

...*arr*,

...*args*.map((*v*) => *v* \* 2),

];

**Slice and Dice**

* **function** removeRandom(items) {}

const removeRandom = (*items*) => {

let idx = Math.floor(Math.random() \* *items*.length);

return [...*items*.slice(0, idx), ...*items*.slice(idx + 1)];

};

* **function** extend(array1, array2) {}

const extend = (*array1*, *array2*) => [...*array1*, ...*array2*];

* **function** addKeyVal(obj, key, val) {}

const addKeyVal = (*obj*, *key*, *val*) => (newObj = { ...*obj*, [*key*]: *val* });

* **function** removeKey(obj, key) {}

const removeKey = (*obj*, *key*) => {

let newObj = { ...*obj* };

delete newObj[*key*];

return newObj;

};

* **function** combine(obj1, obj2) {}

const combine = (*obj1*, *obj2*) => {

return { ...*obj1*, ...*obj2* };

};

* **function** update(obj, key, val) {}

const update = (*obj*, *key*, *val*) => {

let newObj = { ...*obj* };

newObj[*key*] = *val*;

return newObj;

};