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

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

**const** mergeObjects = (obj1, obj2) => ({...obj1, ...obj2})

**const** doubleAndReturnArgs = (arr, ...args) => [...arr, ...args.map(v => v \*2)]

*/\*\* remove a random element in the items array*

*and return a new array without that item. \*/*

**const** removeRandom = items => {

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

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

}

*/\*\* Return a new array with every item in array1 and array2. \*/*

**const** extend = (array1, array2) => {

**return** [...array1, ...array2];

}

*/\*\* Return a new object with all the keys and values*

*from obj and a new key/value pair \*/*

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

*/\*\* Return a new object with a key removed. \*/*

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

*/\*\* Combine two objects and return a new object. \*/*

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

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

}

*/\*\* Return a new object with a modified key and value. \*/*

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