Let's tackle each of the tasks one by one:

### Prime Numbers Filtering

```javascript

function isPrime(num) {

if (num <= 1) return false;

for (let i = 2; i <= Math.sqrt(num); i++) {

if (num % i === 0) return false;

}

return true;

}

function filterPrimes(arr) {

let primes = [];

for (let i = 0; i < arr.length; i++) {

if (isPrime(arr[i])) {

primes.push(arr[i]);

}

}

return primes;

}

// Test case

console.log(filterPrimes([1, 2, 3, 4, 5, 6, 7, 8, 9])); // Output: [2, 3, 5, 7]

```

### Palindrome Checking

```javascript

function isPalindrome(str) {

str = str.toLowerCase().replace(/[^a-z0-9]/g, '');

for (let i = 0; i < str.length / 2; i++) {

if (str[i] !== str[str.length - 1 - i]) {

return false;

}

}

return true;

}

// Test cases

console.log(isPalindrome("level")); // Output: true

console.log(isPalindrome("hello")); // Output: false

```

### Array Reversing

```javascript

function reverseArray(arr) {

let reversed = [];

for (let i = arr.length - 1; i >= 0; i--) {

reversed.push(arr[i]);

}

return reversed;

}

// Test case

console.log(reverseArray([1, 2, 3, 4, 5])); // Output: [5, 4, 3, 2, 1]

```

### Inplace Array Reversing

```javascript

function reverseArrayInplace(arr) {

let start = 0;

let end = arr.length - 1;

while (start < end) {

let temp = arr[start];

arr[start] = arr[end];

arr[end] = temp;

start++;

end--;

}

return arr;

}

// Test case

let arr = [1, 2, 3, 4, 5];

reverseArrayInplace(arr);

console.log(arr); // Output: [5, 4, 3, 2, 1]

```

### Array & Object

```javascript

function formatArray(arr) {

let result = { females: [], males: [] };

for (let i = 0; i < arr.length; i++) {

let [name, age, gender] = arr[i].split(',').map(item => item.trim());

let person = { 'second-name': name.split(' ')[1], age: parseInt(age) };

if (gender === 'female') {

result.females.push({ [name.split(' ')[0]]: person });

} else if (gender === 'male') {

result.males.push({ [name.split(' ')[0]]: person });

}

}

return result;

}

// Test case

let people = ["Patrick wyne, 30, male", "lil wyne, 32, male", "Eric mimi, 21, female", "Dodos deck, 21, male", "Alian Dwine, 22, male", "Patrick wyne, 33, male", "Patrick wyne, 10, male", "Patrick wyne, 40, male"];

console.log(formatArray(people));

```

### Custom Sorting

```javascript

function customSort(arr) {

let result = [];

for (let i = 0; i < arr.length; i++) {

if (!isPrime(arr[i])) {

result.push(arr[i]);

}

}

return result.sort((a, b) => b - a);

}

// Test case

console.log(customSort([2, 3, 4, 5, 6, 7])); // Output: [7, 6, 4]

```

### Majority Element

```javascript

function hasMajorityElement(arr) {

let count = {};

for (let num of arr) {

count[num] = (count[num] || 0) + 1;

if (count[num] > arr.length / 2) {

return true;

}

}

return false;

}

// Test cases

console.log(hasMajorityElement([3, 1, 3, 4, 4, 5, 3, 5, 3, 3, 3, 6, 3])); // Output: true

console.log(hasMajorityElement([3, 1, 3, 4, 4])); // Output: false

```

### Asynchronous JS & Error Handling

```javascript

function setStudentAgeApi(student, age) {

return new Promise(function(resolve, reject) {

setTimeout(() => {

student.age = age;

if (age < 0) {

reject("Bad Age");

} else {

resolve(student);

}

}, 500);

});

}

// Test cases

let student = { name: "denis" };

setStudentAgeApi(student, 20)

.then(result => console.log(result))

.catch(error => console.log(error));

setStudentAgeApi(student, -5)

.then(result => console.log(result))

.catch(error => console.log(error));

```

### API with Await

```javascript

async function myApi(arr) {

await new Promise(resolve => setTimeout(resolve, 1000));

for (let obj of arr) {

obj.totalNumberofFamilyMembers = Object.keys(obj).length;

if (obj.fatherName && obj.fatherName.toLowerCase() === "yves") {

throw new Error("Yves is not an allowed dad in 2022.");

}

}

return arr;

}

// Test case

let family = [{ fatherName: "Yves", motherName: "Alice", childrenNumber: 2 }, { fatherName: "John", motherName: "Mary", childrenNumber: 1 }];

myApi(family)

.then(result => console.log(result))

.catch(error => console.log(error.message));

```

These solutions address the tasks you've provided using only primitive loops and conditions, without using any built-in functions or libraries beyond what's necessary.