Question 1

Write the necessary Node script to make this code work for all arrays: [1,2,3,4,5,6,7,8].even(); // [2,4,6,8] [1,2,3,4,5,6,7,8].odd(); // [1,3,5,7] Test your code in Node.JS CLI

```
Array.prototype.even = function () {

   return this.filter(n => n % 2 == 0);

}

Array.prototype.odd = function () {

   return this.filter(n => n % 2 !== 0);

}

 let arr = [1, 2, 3, 4, 5, 6, 7, 8];

let temp = arr.even();

console.log(temp);

arr = [1, 2, 3, 4, 5, 6, 7, 8];

temp = arr.odd();

console.log(temp);
```

Question 2

1. Explain why do we want sometimes to use setImmediate instead of using setTimeout?

   setTimeout runs in Timer phase, setImmediate runs in check phase. For example: If there's a case needs to run just before close phase, we can use setImmediate.

```
var fs = require('fs'); var path = require('path');

fs.readFile(path.join(__dirname, 'greet.txt'), 'utf8', function(err, data) {

setTimeout(() => { console.log('timeout'); }, 0);

setImmediate(() => { console.log('immediate'); });

process.nextTick(() => console.log('nexttick')); });
```

2. Explain the difference between process.nextTick and setImmediate?

callback of process.nextTick get executed multiple chances in one iteration - highest priotiry. setImmediate only get executed once in one cycle.

how many callbacks will be executed in one iteration? process.nextTick() - all of callbacks in nextTick queue, drain out - Don't use while true loop on process.nextTick, will block event loop. setImmediate - certain, the remaining for next iteration/tick

process.nextTick - API provided natively by Node.js setImmediate - provided by libuv

3. Does Node.js has window object?

No