Homework -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

Answer

```
//homework 1

Array.prototype.even= function(){
   return this.filter(m => m %2 ==0
   )
    }

    Array.prototype.odd= function(){
        return this.filter(m=>  m % 2!==0
        )
    }
    console.log('start');
    let inputArr = [1, 2, 3, 4, 5, 6, 7, 8];
    let evenOutPut = inputArr.even(); // select the even values only
    console.log(evenOutPut);
    let oddOutPut = inputArr.odd(); //  select the odd values only
    console.log(oddOutPut);
    console.log('end');
```

Homework -2

// Fix the slow function to be asynchronous/non-blocking

function slow(callback){

for(let i=0; i<= 5e8; i++){}

if (Math.random() > 0.5) {

return callback("Error",null)

}

callback(null, {id:12345})

}

function exec(fn){

// Complete the code here to implement chaining with callback

}

exec(slow).done(function(data){ console.log(data); })

.fail(function(err){ console.log("Error: " + err); });

Answer

```
function slow(callback){
    for(let i=0; i<= 5e8; i++){}
    if (Math.random() > 0.5) {
    return callback("Error",null)
    }
    callback(null, {id:12345})
    }
    function exec(fn){
    // Complete the code here to implement chaining with callback
    let val = {};

    fn(function (err, data) {
        val.done = function (cb) {
            if (err === null) {
                cb(data);
            }
            return this;
        }
        val.fail = function (cb) {
            if (err !== null) {
                cb(err);
            }
            return this;
        }
    });
    return val;
    }
    exec(slow).done(function(data){ console.log(data); })
    .fail(function(err){ console.log("Error: " + err); });
```

Homework -3

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

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

3. Name 10 global modules/methods available in Node environment.

Answer

1. SetTimeout is simply like calling the function after delayed has finished.Whenever the function is called it is not

executed immediately but queued so that its is executed after all the executing and currently queued eventhandlers finished

first but setImmediate is similar except it doesn't use queue of the functions.We use setImmediate if we want to queue the function

behind whatever I/O event callbacks that are already in the event queue and will schedule a callback to run at check phase of the event loop after IO events' callbacks.

2. The main difference between setImmediate and process.nextTick is that setImmediate queues its callbacks on the event loop while

process.nextTick doesn't.

3.

- Require
- Module
- Exports
- setInterval
- Buffer
- clearInterval
- setTimeout
- clearTimeout
- global