**Exercises**

*Lecture 3:*

ES6 - part 2

1. **Default + rest + spread**
   1. Write a function which returns an array of characters of a string.

const test = () => {

let result = [“a”,”b”,”c”,”d”];

return rusult;

}

1.2. Given 2 arrays:

var array1 = [1, 2, 3];

var randomValue = 'something';

var array2 = [4, 5, 6];

Write a function which returns [1, 2, 3, 'something', 4, 5, 6].

*var array1 = [1, 2, 3];*

*var randomValue = 'something';*

*var array2 = [4, 5, 6];*

*const merge = (...arr) => arr;*

*console.log(merge(...array1,randomValue,...array2));*

function merge(array1, randomValue, array2){

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

}

console.log(merge(array1, randomValue, array2));

* 1. Write a function which returns sum of its arguments (number only).

*const sum = (function() {*

*"use strict";*

*return function sum(...args) {*

*return args.filter(a => Number.isInteger(a)).reduce((a, b) => a + b, 0);*

*};*

*})();*

*console.log(sum(1, 2,"Phat", 3,4,5,6,2,"Nguyen"));*

let arr = [1,2,3,4,5,6,7,8,9,10,”a”,”b”];

const add = (...args) => {

return args.filter(a => Number.isInteger(a)).reduce((a, b) => a + b, 0);

}

console.log(add(...arr));

1.4. Write a function which returns sum of its arguments except for the first one (number only).

function sum\_1(a,...args){

args = args.splice(1,args.length);

return args.filter(a => Number.isInteger(a)).reduce((a, b) => a + b, 0);

}

console.log("sum1:"+sum\_1(5,1,2,3,4));

*const add = (...args) => {*

*args = args.splice(1,args.length);*

*return args.filter(a => Number.isInteger(a)).reduce((a, b) => a + b, 0);*

*}*

*console.log(add(*5,1,2,3,4*));*

1.5. Write a function which returns sum of its arguments except for the last one (number only).

*function sum\_2(...args){*

*args = args.splice(0,args.length-1);*

*return args.filter(a => Number.isInteger(a)).reduce((a, b) => a + b, 0);*

*}*

*console.log(“sum2:”+sum\_2(1,2,3,4,5));*

*const add = (...args) => {*

*args = args.splice(0,args.length -1);*

*return args.filter(a => Number.isInteger(a)).reduce((a, b) => a + b, 0);*

*}*

*console.log(add(*1,2,3,4,5*));*

1.6. Write a function which returns sum of its first 3 arguments (number only).

*function sum\_3(...args){*

*args = args.splice(0,3);*

*return args.filter(a => Number.isInteger(a)).reduce((a, b) => a + b, 0);*

*}*

*console.log("sum3:"+sum\_3(1,4,5,10));*

*const add = (...args) => {*

*args = args.splice(0,3);*

*return args.filter(a => Number.isInteger(a)).reduce((a, b) => a + b, 0);*

*}*

Console.log(add(*1,4,5,10));*

1. **Let + const**

Read some articles about let, const and var to have a better understanding about them.

Recommended articles:

* <https://medium.com/javascript-scene/javascript-es6-var-let-or-const-ba58b8dcde75>
* <https://ponyfoo.com/articles/var-let-const>
* <http://wesbos.com/let-vs-const/>

1. **Iterators + for .. of**

3.1. Create a countdown iterator that counts from 9 to 1. **Use iterator, do not use generator.**

let getCountdownIterator = function(number){

this.\_number = number;

}

getCountdownIterator.prototype[Symbol.iterator] = function(){

let number = this.\_number;

return {

next : function(){

if(1 <= number){

return {value : number--, done: false};

}

return { value : undefined , done: true};

}

};

}

let count\_down = new getCountdownIterator(9);

console.log([...count\_down]);

> [9, 8, 7, 6, 5, 4, 3, 2, 1]

function count\_d(n){

let count = n;

return {

next : function() {

if(1 <= count){

return {value: count--, done:false};

}

return {value:undefined, done:true};

}

}

}

let dm = count\_d(10);

while(true){

let result = dm.next();

console.log(result.value);

if(result.done){

break;

}

}3.2. Create an infinite sequence that generates the next value of the Fibonacci sequence.

The Fibonacci sequence is defined as follows:

fib( 0 ) = 0

fib( 1 ) = 1

for n > 1, fib( n ) = fib( n - 1 ) + fib( n - 2 )

*let fibonaci = {*

*[Symbol.iterator](){*

*let pre = 0;*

*let cur = 1;*

*return {*

*next(){*

*[pre,cur] = [cur,cur+pre];*

*return {done:false,value:cur}*

*}*

*}*

*}*

*}*

*for(let n of fibonaci){*

*console.log(n);*

*}*

1. **Generators**

4.1. Create a countdown iterator that counts from 9 to 1. Use **generator** functions!

let getCountdownIterator = function \*() {

let i = 10;

while( i > 1 ) {

yield --i;

}

}

console.log( [ ...getCountdownIterator() ] );

> [9, 8, 7, 6, 5, 4, 3, 2, 1]

4.2. Use **generator**, create an infinite sequence that generates the next value of the Fibonacci sequence.

The Fibonacci sequence is defined as follows:

fib( 0 ) = 0

fib( 1 ) = 1

for n > 1, fib( n ) = fib( n - 1 ) + fib( n - 2 )

*function \*fibonacci() {*

*let pre = 0, cur = 1;*

*yield pre;*

*yield cur;*

*while( true ) {*

*[pre, cur] = [cur, cur + pre];*

*yield cur;*

*}*

*}*

*let fib = fibonacci();*

*fib.next();*

4.3. Create a lazy filter generator function. Filter the elements of the Fibonacci sequence by keeping the **even values only**.

function \*filter( iterable, filterFunction ) {

for( let element of iterable ) {

if ( filterFunction( element ) ){

yield element;

}

}

}

let arr = [11,22,55,44,66,88,33,22,34,1,222,4,246,-4,-6,1,1,3];

let iterators = arr[Symbol.iterator]();

function\* filterGenerator(iterator,filterFunction){

for(let it of iterator){

if(filterFunction(it)){

yield it;

}

}

}

let fils = filterGenerator(iterators,x => x%2===0 && x > 0);

for(let i of fils){

console.log(i);

}