UNIVERSIDAD NACIONAL DE SAN AGUSTÍN DE AREQUIPA
Laboratorio de Lenguages de programación

CARRERA	CURSO	AMBIENTE
Ingeniería de Sistemas	Lenguages de programación	305

PRACTICA No	NOMBRE DE PRÁCTICA	CODIGO DE LAB.	DURACION (HORAS)
07	Programación funcional Java script	305	2

REVISION	FECHA	DESCRIPCIÓN
1	14/05/2019	Revisión de Guías de Laboratorio

1. OBJETIVOS

- Dar a conocer al estudiante de conceptos básicos programación funcional

2. TEMAS A TRATAR

- Programación funcional con Java script

3. MATERIALES, EQUIPOS, SOFTWARE

- Ordenador
- Navegador google chrome o mozilla
- Editor de texto Sublime o Notepad ++.

4. PROBLEMAS PROPUESTOS

1. Make a function called *composedValue* that takes two functions *square* and *double* and a value and returns *square (double (value))*, i.e., the first function called on the result of the second function called on the value.

```
function square(x) { return(x*x); }
function double(x) { return(x*2); }
```

2. Make a function called *compose* that takes two functions *f1* and *f2* and returns a new function that, when called on a value, will return *f1(f2(value))*. Assume that *f1* and *f2* each take exactly one argument.

```
var f3 = compose(square, double);
f3(5); --> 100
f3(10); --> 400
var f4 = compose(double, square);
f4(5); --> 50
f4(10); --> 200
```

3. Make a function called "*find*" that takes an array and a test function, and returns the first element of the array that "passes" (returns non-false for) the test. Don't use map, filter, or reduce

```
function isEven(num) { return(num%2 == 0); }
isEven(3) --> false
isEven(4) --> true
find([1, 3, 5, 4, 2], isEven); --> 4
```

4. Make a function called “mymap” that takes an array and a function, and returns a new array that is the result of calling the function on each element of the input array. Don’t use map, filter, or reduce.

   mymap([1, 2, 3, 4, 5], square); --> [1, 4, 9, 16, 25]
   mymap([1, 4, 9, 16, 25], Math.sqrt); --> [1, 2, 3, 4, 5]
   Hint: remember the push method of arrays.

5. Make a “pure” recursive version of find. That is, don’t use any explicit loops (e.g. for loops or the forEach method), and don’t use any local variables (e.g., var x = ...) inside the functions. Hint: remember the slice method of arrays.

   function isEven(num) { return(num%2 == 0); }
   isEven(3) --> false
   isEven(4) --> true
   find([1, 3, 5, 4, 2], isEven); --> 4

6. **Make a “pure” recursive version of mymap. Hint: remember the slice and concat methods of arrays.**

   mymap([1, 2, 3, 4, 5], square); --> [1, 4, 9, 16, 25]
   mymap([1, 4, 9, 16, 25], Math.sqrt); --> [1, 2, 3, 4, 5]