# Examen DWEC Tema 4

**Pon tu nombre en h1 y para cada pregunta debe aparecer antes de cada uno <h2> ejercicio i </h2>**
**Todas las salidas deben ser realizadas con la propiedad InnerHtml en algún elemento. Las entradas de datos seran realizadas con un input**

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| 0,75 | 0,75 | 1,75 | 0,75 | 0,75 | 0,75 | 1 | 1 | 1,25 | 1,25 |

1. **Arrays .**
   1. Crea una funcion que reciba un numero **indeterminado** de Arrays de tamaños **distintos** crea una funcion function ej1Mezclar(array1, array2 ) que los mezcle y los muestre en la web.

   let array1 = [1,2,4,5] ; let array2 = [6,7,1]; let array3 = [100]; let array4 = [6,7,1]
   ej1Mezclar( array1, array2) // [1,6,2,7,4,1,5]
   ej1Mezclar( array1, array2,array3) // [1,6,100,2,7,4,1,5]

2. **Funciones, Toma el documento Ejercicio2.html** Crea en el documento la función propuesta de cuatro maneras diferentes:
   - tradicional
   - anonima asignada a una variable
   - anonima autoejecutable
   - anonima en forma flecha asignada a una variable
   - anonima flecha con parametros.

3. **Conjuntos**

**a)** Crea una función crearConjunto que dado un numero **indeterminado** de parametros, nos devuelva un array que con solo los que **NO** estan repetidos.
CrearConjunto(7,2,3,4,555,5,5,5,5,5,5,5,5,5,3,3,3,1,1,1) debe devolver [7,2,3,4,5,1]

**b)** Añade una funcion unirConjuntos que una ambos conjuntos eliminando los repetidos
unirConjuntos( [7,2,3,6,5,1] ,[7,2,3,4,5,1,10,9] ) debe devolver [7,2,3,4,5,1]

**c)** Añade una funcion disjuntosConjuntos que una ambos conjuntos eliminando los que estan en ambos : disjuntosConjuntos( [7,2,3,6,5,1] ,[7,2,3,4,5,1,10,9] ) debe devolver [6,10,9]

**d)** 4. Añade una funcion MostrarConjunto() muestre como un string el array
Se valorara el uso de las funciones propias de los conjunto

4. **Metodos Funcionales 1 (los ejercicio de esto punto deben hacerse con metodos funcionales si no no vale)**

   - Crea una función tresMitad(array) que recorra un array de cadenas y nos devuelva la cadena de tres que aparecen en la mitad y los dos laterales. Si tiene menos de 3 se devuelve todo. Si tiene pares para mitad se escoge mitad-1.
     tresMitad(['pepes','Lopez','c','javascript','java']) // 'epe','ope','c','asc','jav',

5. **Metodos Funcionales 2**
   - Crea una función sumarPotencias2Indices que nos devuelva el resultado de multiplicar cada numero por su posicion elevado a 2 y sumarlos por ejemplo
     sumarPotencias2Indices([1,3,4,2,1]) // [1*1 + 3*2 + 2*4 + 2*8 + 1*16] = 1+6 + 8 + 16 + 16 = 47

6. **Metodos Funcionales 3**

# Examen DWEC Tema 4

- Crea una función ordenTamaño que tenga por entrada un array de strings y nos lo devuelva **ordenados por su tamaño y si son iguales por el orden alfabetico de la *primera letra***
  ordenTamaño ([“hola”, “mv”, “adios”, “xz” , “ b”]) => [“ b”, “mv”, “xz” , “hola”,“adios”]

7. **Metodos Funcionales 4**
   - Crea una función sumasLetras(ArrayDoble) que sume todos los elementos de un array de arrays . Puede haber letras con los valores Hexadecimales A = 10 B=11 … F= 15 … si sale algo mas que F es un error
     **sumasLetras ([ 1,2], [“B”,1, 1], [1,3,”A”] ]) => 1+2+11+1+1+1+3+10 = 30**

8. **Metodos Funcionales 5**
   - **Crea una función contarCadenas(array) que** Dado un array con palabras repetidas, debes **agruparlas y contar cuántas veces aparece cada una**, usando únicamente reduce..

     const palabras = [ "js", "html", "css", "js", "css", "js", "react" ];

     Debes usar reduce para genera un **array de arrays** donde Cada sub-array tenga el formato: ["palabra", cantidad]

   - **contarCadenas(palabras) //** [ ["js", 3], ["html", 1], ["css", 2], ["react", 1] ]
   - Ordena el resultado de mayor a menor cantidad:

     [ ["js", 3], ["css", 2], ["html", 1], ["react", 1] ]

9. **Arrays Multidimensional.** Dado un valor num haz una funcion que nos devuelva un array del triangulo de pascal y muestralo por pantalla en forma de tabla o de triangulo

   Para hacer el triangulo solo debes ver que
   triangulo[i] = (triangulo[i] + triangulo[i-1]) si esta definido si no 0. Por ejemplo para n=6

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 2 | 1 | 0 | 0 | 0 | 0 |
| 1 | 3 | 3 | 1 | 0 | 0 | 0 |
| 1 | 4 | 6 | 4 | 1 | 0 | 0 |
| 1 | 5 | 10 | 10 | 5 | 1 | 0 |
| 1 | 6 | 15 | 20 | 15 | 6 | 1 |

10 **fibonaci** modifica el anterior para Calcular un termino de la secuencia de fibonaci usando el triangulo de pascal sabiendo que se calcula con las diagonales de este triangulo

fib(n) = triangulo[n] [1] + … + triangulo[1] [n]