### TRABAJO PRÁCTICO NRO. 04 – JAVASCRIPT

* Objetos a partir de clases
* Objetos literales
* Rest Operator
* Spread Operator
* Conversión a formato JSON
* Destructuring

**Especificaciones:**

* Utilizar manejo del DOM
* Crear documento HTML y CSS a gusto del alumno
* Definir el archivo controlador.js que capture los elementos con los que interactuará el usuario y agregar los comportamientos de los objetos/elementos HTML utilizando “addEventListener”.
* Si el programa, necesita una ó mas funciones debe (expresarlas como arrow function). Y si son funciones específicas de cálculos, funciones específicas de la resolución del problema debe aislarlas en un archivo separado denominado “modelo.js”. allí deberá tener su conjunto de funciones y exportarlas para que el controlador.js pueda consumirlas.
* Desde el archivo “controlador.js” deberá importar esas funciones que están en el archivo “modelo.js”.

### Ejercicio Nro. 08: (instanciación de objetos)

El Concepto de la POO – programación orientada a objetos es un tema fundamental que un programador debe incorporar, ya sea en el rol de “constructor de clases” o como “consumidor de clases realizadas por terceros”. En esta oportunidad únicamente asumiremos el rol de consumidor de clases realizadas por alguna empresa que nos facilita la creación de componentes avanzados o complejos y que realizarlos por nuestra propia cuenta nos llevaría muchísimo tiempo.

Para este ejemplo utilizaremos el siguiente ejemplo, donde tenemos un proyecto que utiliza la biblioteca chart y que nos permite un gran abanico de posibilidades para realizar gráficos en nuestros documentos HTML. Dejaré un proyecto donde estará incorporado un gráfico circular, viendo como es el código que permite la instanciación de ese objeto y graficarlo en el DOM; toma los siguientes datos que te paso a continuación y grafica otro (**sin eliminar el anterior**) debajo del mismo, modifica los rótulos, los datos de las empresas, los colores a gusto.

Link Proyecto: <https://drive.google.com/file/d/1pLQ-Mhp53xm2gZPflvkfeYrwFZYxQuc0/view?usp=sharing>

    // Etiquetas para las empresas

    const labelsEmpresas = [

        "EMPRESA ALPHA",

        "EMPRESA BETA",

        "EMPRESA GAMMA",

        "EMPRESA DELTA",

        "EMPRESA EPSILON",

        "EMPRESA ZETA"

    ];

    // Porcentajes de participación

    const dataEmpresas = [20, 25, 15, 10, 20, 10];

    // Colores para cada segmento

    const coloresEmpresas = [

        "#FF5733", // Rojo

        "#33FF57", // Verde

        "#3357FF", // Azul

        "#FF33A1", // Rosa

        "#FFC300", // Amarillo

        "#DAF7A6"  // Verde claro

    ];

### Ejercicio Nro. 09: ( spread operator … y métodos de vectores)

Un profesor de programación tiene 4 curso o cohortes, en donde en cada cohorte existen alumnos diferentes que cursaron su módulo de “fundamentos de programación con JavaScript”, el profesor tiene en un vector diferente los alumnos y sus notas finales del cursado. “Se aprueba con notas > 4”. El listado de alumnos es el siguiente:

const cohorte01 = [

{ dni: "12345678", nombre: "Juan", apellido: "Pérez", nota\_final: 8 },

{ dni: "23456789", nombre: "Ana", apellido: "Gómez", nota\_final: 6 },

{ dni: "34567890", nombre: "Carlos", apellido: "López", nota\_final: 5 },

{ dni: "45678901", nombre: "María", apellido: "Martínez", nota\_final: 7 }

];

const cohorte02 = [

{ dni: "56789012", nombre: "Sofía", apellido: "Ramírez", nota\_final: 9 },

{ dni: "67890123", nombre: "Luis", apellido: "Fernández", nota\_final: 4 },

{ dni: "78901234", nombre: "Elena", apellido: "Torres", nota\_final: 5 }

];

const cohorte03 = [

{ dni: "89012345", nombre: "Pedro", apellido: "Álvarez", nota\_final: 10 },

{ dni: "90123456", nombre: "Clara", apellido: "Méndez", nota\_final: 3 },

{ dni: "12345679", nombre: "Jorge", apellido: "Salinas", nota\_final: 6 },

{ dni: "23456780", nombre: "Andrea", apellido: "Cruz", nota\_final: 2 },

{ dni: "34567891", nombre: "Sergio", apellido: "Paredes", nota\_final: 8 }

];

const cohorte04 = [

{ dni: "45678902", nombre: "Lucía", apellido: "Ortiz", nota\_final: 7 },

{ dni: "56789013", nombre: "Miguel", apellido: "Vega", nota\_final: 4 },

{ dni: "67890124", nombre: "Raquel", apellido: "Silva", nota\_final: 9 },

{ dni: "78901235", nombre: "Hugo", apellido: "Moreno", nota\_final: 5 },

{ dni: "89012346", nombre: "Natalia", apellido: "Quinteros", nota\_final: 6 },

{ dni: "90123457", nombre: "Diego", apellido: "Arce", nota\_final: 3 }

];

1. Realice una función, que reciba como parámetro un vector de alumnos (puede ser de cualquier cohorte) los recorra con un **forEach** y muestre por consola, el apellido y el nombre.

Invoque esa misma función, pasándole como parámetro el vector de la cohorte01, luego el vector de la cohorte02 y así sucesivamente hasta la cuarta cohorte.

1. El profesor desea unificar todos los alumnos en un solo vector, dado que estructuralmente son iguales, utilice **spread operator** para “unificar el contenido de las cuatro cohortes” en un solo vector. Muestre por consola el vector resultante.
2. Sobre los datos unificados, cree dos vectores nuevos, uno puede llamarse alumnosAprobados y el otro alumnosDesaprobados.

Para crear el vector de alumnos aprobados, aplique el método **filter** sobre el Vector que tiene todos los alumnos cuando la nota > 5;

Para crear el vector de alumnos desaprobados, aplique el método **filter** sobre el Vector que tiene todos los alumnos cuando la nota <= 5;

1. Sobre el vector Total de alumnos (cuatro cohortes), aplique el método **“sort”** = ordenar y ordene los datos de todos esos alumnos por nota de menor a mayor.
2. Sobre el vector Total de alumnos, realice una búsqueda de los siguientes alumnos aplicando el método **“find”** que tienen los vectores incorporados y busque algún alumno por su “dni”. El que ud. Desee.
3. Sobre el vector Total de alumnos, deseamos “contabilizar” la cantidad de alumnos aprobados, es decir la cantidad de alumnos que tienen nota > 5; para ello utilicen el método reduce que tienen los vectores incorporados para contabilizar. Realice esto de dos formas
   1. Usando una arrow function que a dentro utilice un condicional if/else para saber si la nota > 5 y por lo tanto contabilizar los aprobados;
   2. Usando una arrow function que a dentro utilice un operador ternario para saber si la nota > 5 y por lo tanto contabilizar los aprobados;

Ambas formas deben llegar al mismo resultado, únicamente que se implementaron de dos formas diferentes.