FACULTAD DE CIENCIAS EXACTAS
UNIVERSIDAD NACIONAL DEL CENTRO DE LA PROVINCIA DE BUENOS AIRES

TUDAI
FINAL / PREFINAL - WEB 1

14 JULIO 2022

Alumno	DNI	Firma	#hojas (escritas)
Condición: PREFINAL - FINAL - LIBRE		Comisión Práctica (1-15):	

Nota:

- *Es necesario **al menos un avance en JS para aprobar***
- *El código debe estar prolijo e indentado.*

1- Verdadero y Falso

Marque verdadero o falso y justifique todas sus respuestas brevemente. Puede emplear un ejemplo para la justificación. Sin justificación la respuesta no tiene validez

a. El DOM se utiliza solo para identificar las clases declaradas en la hoja CSS.
   F Sirve para vincular todos los elementos HTML, id´s, clases, con js
b. Los métodos para interactuar con los datos de un servicio REST son solo dos.
   F son cuatro, GET para obtener, POST para crear, PUT para modificar, DELETE para borrar
c. El uso de funciones es considerado buena práctica en JS.
   V Con el uso de funciones se puede reutilizar código, modularizar/ dividir el problema en subtareas más específicas
d. La semántica se emplea para incluir contenido multimedia en los sitios web.
   F, se utiliza para dar sentido al contenido
e. En un sitio que emplea AJAX, no existen diferencias en el uso de `fetch` para Partial Render y para el uso de un servicio REST.
   F, si existen diferencias en como se manejan los datos. Partial Render maneja datos/contenido HTML, debe ser tratado con .text(). REST maneja datos JSON debe ser tratado con .json()
f. Es indistinto incluir el vínculo de JS en el header o al final de body.
   F. No es indistinto, debe ir al final del body para que se cargue todo el contenido del DOM y los elementos asi puedan ser localizados / modificados desde JS. Puede ir al inicio o es indistinto SOLO SI desde JS encapsulamos con una función general que se ejecute luego de haber cargado el DOM con por ejemplo
   document.addEventListener("DOMContentLoaded",iniciarPagina);
g. El modelo de cajas (BOX MODEL) solo es válido para los sitios responsive.
   F. El modelo de cajas aplica a cualquier sitio responsive o no responsive y incluso el modelo es previo al diseño responsive.
h. Si en CSS se está usando herencia, no puede emplearse cascada.

F. Son dos metodologías no excluyentes para aplicar los estilos. Herencia se emplea para aplicar estilos por jerarquías mientras que cascada se usa para aplicar estilos por especificidad.

i. Las tecnologías Front-End se ejecutan solo del lado del servidor.

F. Una vez cargadas en el navegador las tecnologías Front-End se ejecutan solo del lado del navegador.

j. El debug y verificación de errores en JS solo puede verse en un editor de código (como por ejemplo VS Code).

F. Podemos usar la consola del navegador para hacer debug y verificación, podemos ver los errores por consola e incluso desde nuestro código podemos ayudarnos para imprimir mensajes por consola para hacer debug.

2. HTML + CSS

Dado el siguiente **código HTML del index.html,** y suponiendo que incluye el head correcto

```
<body>
    <header>
        <h1>Bienvenidos a Web 1</h1>
        <img src="images/logo.png" alt="logo">
        <div class="social">
            <ul>
                <li><a href="http://facebook.com/web1">Facebook</a></li>
                <li><a href="http://twitter.com/web1">Twitter</a></li>
            </ul>
        </div>
    </header>
    <nav>
        <ul>
            <li><a href="#">Inicio</a></li>
            <li><a href="programa.html">Programa</a></li>
            <li><a href="contacto.html">Contacto</a></li>
        </ul>
    </nav>

</body>
```

Escriba el **código CSS** para crear los siguientes estilos generales a un sitio web.
Se deben utilizar tags semánticos, NO se puede modificar el HTML.

i. Todos los títulos y párrafos del sitio deben ser azules, salvo el título con texto “Bienvenidos a Web 1” que debe ser rojo.
ii. Los hiperlinks de las redes sociales deben tener fondo negro y letra color blanco, y cuando se pasa por encima con el mouse el fondo debe cambiar a azul.
iii. Los hiperlinks del menu deben tener fondo amarillo y letra negra.

iv. Ademas del código de css, escriba el código html para incluir un pie de página de fondo negro que contenga un párrafo con el texto “Web 1 - 2022”. El texto de dicho párrafo debe ser de color naranja y alineado al centro.

```
/* i. */
h1, h2, h3, h4, h5, h6, p {
    color: blue;
}

header h1 {
    color: red;
}

/* ii. */
div.social ul li a {
    background-color: black;
    color: white;
}

div.social ul li a:hover {
    background-color: blue;
}

/* iii. */
nav ul li a {
    background-color: yellow;
    color: black;
}

/* iv. */
footer{
    background-color: black;
}

footer p {
    text-align: center;
    color: orange;
}
```

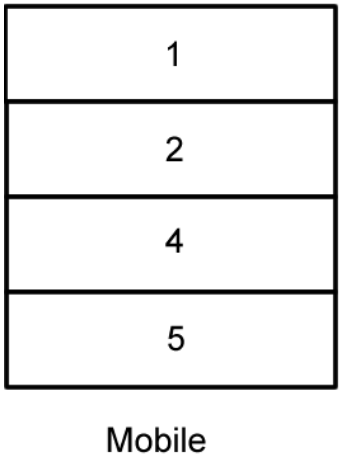
HTML

```
<footer>
    <p>Web 1 - 2022</p>
</footer>
```

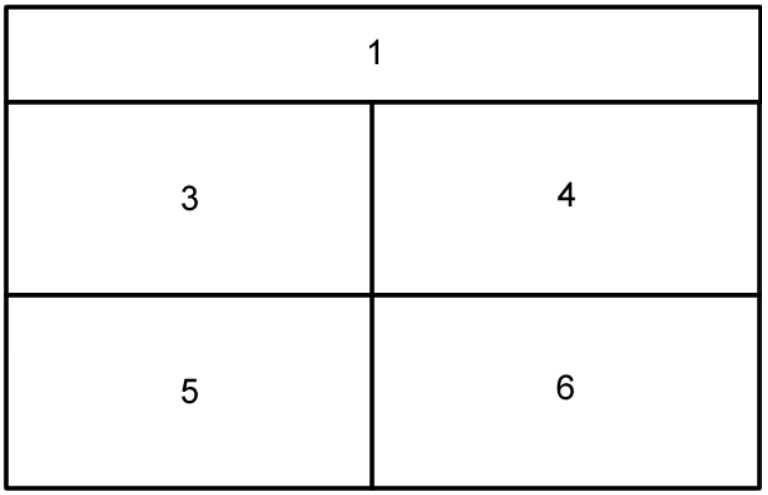
3. Diseño responsive

Escriba el código **HTML** y **CSS** necesario para crear el siguiente layout responsive.

- Se debe realizar utilizando **Mobile First**.
- No importan las dimensiones, ni bordes de los elementos.
- No se puede utilizar ningún framework CSS



Mobile



Desktop

ALTERNATIVA FLEX

HTML

```
<div class="a1">
        <h1>1</h1>
    </div>

    <div class="a2">
        <h1>2</h1>
    </div>

    <div class="a34">
        <div class="a3">
            <h1>3</h1>
        </div>
        <div class="a4">
            <h1>4</h1>
        </div>
    </div>

    <div class="a56">
        <div class="a5">
            <h1>5</h1>
        </div>

        <div class="a6">
            <h1>6</h1>
        </div>
```

CSS

```
.a3, .a6 {
    display: none;
}

/* optativo*/
.a34 {
    display: flex;
    flex-direction: column;
}

.a56 {
    display: flex;
    flex-direction: column;
}


@media only screen and (min-width: 720px) {

    .a3, .a6 {
        display: block;
    }

    .a2 {
        display: none;
    }
```

```
    </div>
```

```
    .a34 {
        display: flex;
        flex-direction: row;
    }

    .a56 {
        display: flex;
        flex-direction: row;
    }

 }
```

ALTERNATIVA GRID

```
<div class="container">

        <div class="a1">
            <h1>1</h1>
        </div>

        <div class="a2">
            <h1>2</h1>
        </div>

        <div class="a3">
            <h1>3</h1>
        </div>

        <div class="a4">
            <h1>4</h1>
        </div>

        <div class="a5">
            <h1>5</h1>
        </div>

        <div class="a6">
            <h1>6</h1>
        </div>

 </div>
```

```
.container {
    display: grid;
    grid-template-areas:"a1"
                        "a2"
                        "a4"
                        "a5";
 }

.a1 {
     grid-area: a1;

}

.a2 {
    grid-area: a2;

}

.a3 {
    grid-area: a3;
    display: none;
}

.a4 {
    grid-area: a4;
}

.a5 {
    grid-area: a5;
}

.a6 {
    grid-area: a6;
    display: none;
}
```

	@media only screen and (min-width: 720px) {
	
	.a3, .a6 {
	display: block;
	}
	
	.a2 {
	display: none;
	}
	
	.container {
	display: grid;
	grid-template-areas:"a1 a1"
	"a3 a4"
	"a5 a6";
	}
	
	}

4. Javascript

La bebida energizante PowEnergy tiene un nuevo sitio web, el cual permite ir registrando participantes para realizar un sorteo entre los inscriptos. Se propone el siguiente código html

```
<h1>PowEnergy Drink</h1>
<form id="form_registro">
    <label for="nombre">Nombre y Apellido</label>
    <input id="nombre" name="nombre" type="text">
    <label for="dni">DNI</label>
    <input id="dni" name="dni" type="number">
    <label for="edad">Edad</label>
    <input id="edad" name="edad" type="number">
    <button id="btn_registrar" type="submit">Registrarse</button>
</form>

<button id="btn_sortear">Sortear</button>

<h2>Participantes del sorteo</h2>
<ol id="lista_sorteo">
   <!-- escribir el listado de Nombres y dni de los participantes -->
</ol>
<div id="aviso_ganador">
    <!-- mensaje y aviso del ganador -->
</div>
```

Escribir el código de Javascript para cumplir los siguientes requerimientos:

- Se deben ir registrando los usuarios que se van inscribiendo a través del formulario, con el requerimiento que los usuarios deben ser mayores de 18 años y no pueden inscribirse más de una vez.
- Luego que se oprime el botón Sortear se debe mostrar un listado con los nombres y dni de cada uno de los participantes registrados en el sorteo.

- Al oprimir Sortear, el ganador debe ser evaluado al azar entre el total de las personas registradas y deben mostrarse los datos del ganador en la página (nombre, edad y dni).
- El aviso del ganador debe aparecer con letra verde acompañado del texto “PowEnergy - Beber con moderación”. En el caso que el ganador sea menor de 21 años se debe mostrar el aviso del ganador con letra de color rojo e indicando además un texto que anuncie “Para retirar el premio el gandor debe ir acompañado de un mayor de 21 años”.

NOTA:

- si utiliza alguna validación HTML5 agregue las propiedades al html en esta misma hoja
- Si hace uso de **CSS** aplicando estilos desde **JS**, debe escribir el código

JS

```
"use strict";

let registros = [];
let registrado = false;
let form = document.querySelector("#form_registro");
let btnSortear = document.querySelector("#btn_sortear");
let avisoGanador = document.querySelector("#aviso_ganador");
let listaSorteo = document.querySelector("#lista_sorteo");
btnSortear.addEventListener("click", sortear);

form.addEventListener("submit",agregar);

function agregar(e) {
   e.preventDefault();
   let formData = new FormData(form);
   let nombre = formData.get("nombre");
   let dni = formData.get("dni");
   let edad = formData.get("edad");

   for(item of registros){
        if(item.dni === dni){
            registrado = true;
        }
    }

   if (edad > 18 && !registrado) {
       let registro = {
           'nombre' : nombre,
           'dni': dni,
           'edad': edad
       }
       registros.push(registro);
   }
   else {
        alert("No se puede registrar");
   }
}
```

```
function sortear(){

    mostrarListado();

    let indexGanador = Math.floor(registros.length*Math.random());

    avisoGanador.innerHTML = "<p>PowEnergy  - Beber con moderación</p>";

     avisoGanador.innerHTML += "<p>El ganador es: " + registros[indexGanador].nombre +
" " +
    registros[indexGanador].dni + " " + registros[indexGanador].edad + "</p>";

    if(registros[indexGanador].edad < 21) {
            avisoGanador.innerHTML += "<p>Para retirar el premio el ganador debe ir
acompañado de un mayor de 21 años</p>";
        avisoGanador.classList.remove("mayor");
        avisoGanador.classList.add("menor");
    }
    else {
        avisoGanador.classList.remove("menor");
        avisoGanador.classList.add("mayor");
    }

}

function mostrarListado(){
    for(item of registros){
        listaSorteo += "<li>" + item.nombre + " " +  item.dni + "</li>";
    }
}
```

CSS

```
.mayor {
    color: green;
}

.menor {
    color: red;
}
```

5. AJAX

Un servicio API REST tiene almacenado los alumnos que rindieron examen de ingreso con una url **http://uniceningreso/api/alumnos**

Al consultar la url de dicha API se obtiene una respuesta como la siguiente: [{ “id”: 1, "nombre": "Pedro Garces", “dni”: 43222132, "calificacion": 8 }, { “id”: 2, "nombre": Sofia Fernandez", “dni”: 41325167, "calificacion": 9 }, ...]	1. Escriba una función que tenga el *id* como parámetro y borre del servicio al alumno que posea ese *id*. 2. Escriba una función para consultar el servicio REST y que permita mostrar por consola los nombres y dni de los alumnos que se encuentran aprobados. Además se debe mostrar por consola el porcentaje de aprobados evaluado sobre el total de alumnos. Se considera aprobado quien tenga nota 4 o superior.

Ayuda Fetch:

```
fetch( url, {
       "method": …,
       "headers": {"Content-Type": "application/json"},
       "body": …
     })
```

AJAX, JS

```
let miUrl = "http://uniceningreso/api/alumnos";

//1
async function borrarDato(id){

    try {
        let res = await fetch(miUrl + "/" + _id, {
            "method": "DELETE",
        });

        let json = await res.json();
        console.log(json);
        }
    catch (error) {
    console.log(error);
    }
```

```
}

//2
async function mostrarAprobados(){

    let cuentaAprobados = 0;

    try {
        let res = await fetch(miUrl);
        let json = await res.json();
        for (item of json){
            if(item.calificacion > 4){
                console.log(item.nombre);
                console.log(item.dni);
                cuentaAprobados++;
            }
        }
        console.log(100*cuentaAprobados/json.length);
        }
    catch (error) {
    console.log(error);
    }

}
```