	Facultad de Tecnología y Cs. Aplicadas – U.N.Ca. – 13/09/2024	CARRERA:
	APELLIDO Y NOMBRE:	M.U.:

## CÁTEDRA: PROGRAMACIÓN III / DISEÑO DE SOFTWARE III

### PRIMER PARCIAL PRÁCTICO (HTML, CSS y JavaScript)

#### Introducción

La evaluación consta de ejercicios de selección múltiple, ejercicios para completar código y ejercicios de interpretación de código.

La puntuación máxima es de 10 puntos. Los ejercicios de selección múltiple tienen una puntuación baja, mientras que los ejercicios para completar código y los que requieren interpretación de código serán fundamentales para determinar su aprobación.

#### 1. Escribe la url relativa para enlazar el recurso “script\_lista.js” desde “lista-articulos.html”

```

subastas
├── templates
│   └── lista-articulos.html
├── estaticos
│   └── js
│       └── script_lista.js

```

URL RELATIVA: \_\_\_\_\_

#### 2. Completar el formato de una petición HTTP donde se envían a la url: “/persona/listar”, los siguientes parámetros mediante la petición POST: nombre=Lucas, edad=19

```

_____ HTTP/1.1
Host: misitio.com
User-Agent: Mozilla/5.0 (Win NT 10.0;)

```

#### 3. Escriba la fracción de código HTML correspondiente para crear un formulario web de acuerdo a los siguientes requisitos:

##### 3.1 Definir los siguientes campos junto con los atributos correspondientes para que puedan ser procesados en el servidor:

- 3.1.1. un campo de texto para ingresar un nombre (obligatorio)
- 3.1.2. un campo de correo electrónico (obligatorio)
- 3.1.3. un campo de selección de género (obligatorio, opciones: F o M)
- 3.1.4. un campo para subir una foto del dni de una persona (obligatorio)
- 3.1.5. un área de comentarios

##### 3.2 Los datos ingresados en los campos del formulario deben ser procesados (junto con el archivo adjunto) por el recurso del lado del servidor: “/procesar-datos/”. Además, los datos deben enviarse dentro del cuerpo de la petición HTTP. **Definir los atributos y/o campos correspondientes para cumplir estos requisitos.**



```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body>
```

```

    </body>
</html>
```

4. El siguiente código HTML no es correcto, explicar por qué:

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="UTF-8">
    <h1>Título Principal</h1>
  </head>
  <body>
    <p>Este es un párrafo.</p>
    <a src="/listados" value="Click aquí para ver listados">
  </body>
</html>
```

**Respuesta:**


---

---

---

---

---

	Facultad de Tecnología y Cs. Aplicadas – U.N.Ca. – 13/09/2024	CARRERA:
	APELLIDO Y NOMBRE:	M.U.:

5. De acuerdo a la especificidad, ¿Qué selector de CSS es más específico para aplicar un estilo a un elemento `<p>` que tiene una clase llamada `resaltado` y se encuentra dentro de un `<div>` con id `contenedor`? Explicar por qué.

- a) `p.resaltado`
- b) `div .resaltado`
- c) `#contenedor .resaltado`
- d) `#contenedor p`

6. Considerando la siguiente porción de código, ¿qué valor de ancho concreto se computará para el elemento `<table>`?

```
--CSS--

main {width: 750px;}
section.datos {width: 90%;}
section.datos table {width: 75%}
```

```
--HTML--

<main>
  <section class="datos">
    <table>...</table>
  </section> </main>
```

**Respuesta:**

---



---

7. A partir de las descripciones siguientes, añadir los **selectores CSS** que correspondan para aplicar los estilos deseados según los comentarios de cada regla:

<input type="text"/>	/* Todos los párrafos que sean descendientes directos (hijos) de un elemento <code>seccion</code> ( <code>section</code> ) */
<input type="text"/>	/* Todas las listas <code>&lt;ul&gt;</code> con un atributo <code>id="principal"</code> y que se encuentren dentro de la etiqueta <code>&lt;nav&gt;</code> */
<input type="text"/>	/* Elementos <code>"span"</code> que se encuentran inmediatamente después de elementos <code>"a"</code> */
<input type="text"/>	/* Todos los elementos <code>input</code> con atributo <code>type="radio"</code> y que estén seleccionados ( <code>check</code> ) */

8. ¿Cuál de las siguientes opciones describe correctamente la diferencia entre **let** y **var** en JavaScript?

- a) **let** define una variable global, mientras que **var** define una variable de bloque.
- b) **let** permite redefinir una variable en el mismo ámbito, mientras que **var** no.
- c) **var** tiene un alcance de función, mientras que **let** tiene un alcance de bloque.
- d) **var** y **let** son completamente equivalentes.

9. Complete el siguiente fragmento de código JavaScript para implementar un sistema de búsqueda en tiempo real. Filtre una lista de elementos `<li>` que se encuentra dentro de una lista `<ul>`, según el texto ingresado en un campo de entrada:



--- HTML ---

```
<input type="text" id="txt_busqueda" placeholder="Ingrese texto para filtrar lista">
<button id="btn_busqueda" >Filtrar</button>
<ul id="lista">
  <li>Python</li>
  <li>Javascript</li>
  <li>Python/Django</li>
  <li>HTML</li>
</ul>
```

--- JS ---

```
/* Asignar el evento correspondiente al botón para que se realice la tarea */
document.getElementById('btn_busqueda')._____ {
  /* Obtener el valor del texto a buscar */
  const txt_busqueda = _____
  const items = document.querySelectorAll('ul#lista li');

  /* Recorrer la lista de ítems y dejar visible solo los elementos <li> que coincidan con la búsqueda, al resto de los
  elementos ocultarlos */

}
});
```

10. Completa el siguiente script donde se debe crear una clase para implementar una cuenta bancaria. La clase debe poseer dos propiedades: nombre del titular y saldo, y dos métodos: ingresar() y retirar(). El primero incrementa el saldo en la cantidad indicada en el argumento y el segundo lo reduce. No se puede sacar más de lo que exista en el saldo.

/\* Completar aquí con la definición de la clase o función constructor de objetos \*/

```
let micuenta = new Cuenta('Juan', 0);
micuenta.ingresar(1000);
console.log("El saldo actual es " + micuenta.saldo);
micuenta.retirar(100);
console.log("El saldo actual es " + micuenta.saldo);
```