|  |  |
| --- | --- |
|  | **2021** |
|  | **Facultad de Ingeniería y Tecnología Informática**  **Universidad de Belgrano** |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **PROGRAMACIÓN 3**  **GUIA DE TRABAJOS PRACTICOS** |
| **Tec. en Programación de Computadoras (560-2020)** |

|  |  |
| --- | --- |
| **Asignatura** | **Programación 3** |
| **Carrera** | **Tec. en Programación de Computadoras** |
| **Plan** | **2014** |
| **Ciclo** | **2021** |
| **Cuatrimestre** | **1ero** |
| **Trabajo Práctico** | **1** |
| **Tema** | *Desarrollo Frontend* |
| **Tipo de Práctica** | **Formación Experimental (P1) – ejercicios 1 a 30** |

**Ejercicio 1:** Crear una página donde se debe incorporar los tags: <doctype>, <head>, <title>, <body>, <h1>…<h6> y <p>. Guardar la página con el nombre ejercicio1.html y ejecutar en el navegador web.

**Ejercicio 2:** Crear una página donde exista un título, una breve introducción de un tema, una cita o referencia utilizando el tag <blockquote>, sumar también el tag <hr>. Finalmente, cerrar con un último párrafo y visualizar la página en el navegador web.

**Ejercicio 3:** Crear una página, llamada textoformateado.html, y crear diferentes párrafos aplicando cada uno de los tags visto para negrita, cursiva, énfasis, código, depuración, subrayado total y subrayado parcial.

**Ejercicio 4:** Crear una pagina combinando las tres listas: no ordenadas, ordenadas y de definición.

**Ejercicio 5:** Crear una página HTML y su respectiva página CSS utilizando los elementos limitantes en alto y ancho, con la combinación de diferentes posibilidades entre min-height, min-width, max-height y max-width.

**Ejercicio 6:** Crear una página HTML y su respectiva página CSS. Escribir un texto en cada uno de los elementos <p> y <div> y ver como se refleja este, sobre la base de las propiedades aplicadas:

<p>: Borde sólido, de color marrón. La fuente debe ser Verdana de 3 puntos. E margen izquierdo debe estar separado 10 pixeles del borde respectivo. El margen derecho debe separarse 20 pixeles del margen respectivo. El margen superior debe estar separado 30 pixeles del margen respectivo.

<div>: El borde debe ser punteado, de color chocolate. La fuente debe ser similar a la del elemento <p>. El elemento padding debe estar separado del extremo superior unos 5 pixeles. El elemento padding inferior debe separarse también 5 pixeles.

**Ejercicio 7:** Completar la web vista en la clase Diseño UI con CSS. Tomar como base la página HTML y el archivo CSS del último ejercicio, y replicar la página HTML nombrando cada nueva página como cada sección que planificamos en este ejercicio. Luego agregar contenido especifico a cada página.

* **Página principal o Home**: Agregar al botón Home, la clase active, que ya está creada en CSS. Luego agregar los hipervínculos en cada uno de los botones de la barra de navegación hacia las nuevas páginas HTML.
* **Página Empresa**: Aprovechar el contenido de texto heredado de la página principal, y modificar el mismo agregando una imagen vertical relacionada a Empresa, de unos 200 x 325 pixeles aproximadamente, con alineación a la izquierda de la pantalla y su propiedad width debe ser entre el 20% y 30%. Luego entre cada elemento <p> agregar una regla horizontal. Por último, agregar los hipervínculos en la barra de navegación, y la clase active al botón correspondiente.
* **Página Noticias**: Esta página incluirá una tabla con estructura de dos columnas, y dos filas por cada columna. En la primera fila de cada columna insertar un elemento H1, y en la segunda fila de cada columna insertar texto que complete el contenido. Las propiedades CSS a usar en el elemento table son: padding(heredado), y el border de al menos 6px, de color blanco. Por último, agregar también los hipervínculos en la barra de navegación, y la clase active al botón correspondiente.
* **Página Contacto:** Deberá tener domicilio y teléfonos relacionado con la empresa, y una o más direcciones de correo electrónico, con sus respectivos hipervínculos. La propiedad vertical-align debe ser en el extremo superior, su propiedad padding debe ser entre 60px y 80px y el formato de visualización de los párrafos debe ser table-cell. También se debe agregar los hipervínculos a la barra de navegación, y la clase active al botón Contacto.

**Ejercicio 8:** En el siguiente ejemplo de código se han declarado tres variables (a, b y resultado). Escribir el código para ejecutar operaciones matemáticas y lógicas que devolverá un resultado:

var a = 10;

var b = 20;

var resultado = 0;

// sumar dos números

resultado = a + b;

console.log(resultado);

// hacer lo mismo para restarlos, multiplicarlos, dividirlos, comparar si a es mayor o igual que b y comparar si a es distinto que b.

**Ejercicio 9:** Crear una página HTML con dos elementos <p>, y un hipervínculo. Las características adicionales del ejercicio son las siguientes:

* Debe haber un título H1.
* Debe contener dos elementos párrafo.
* Debe contener un hipervínculo.
* Debe existir una función JS dentro de la misma página.

Entonces, al presionar el hipervínculo, se deberá ejecutar la función JS. Dentro de la función JS, declarar una variable, la cual debe almacenar el texto especificado en el primer elemento <p>. Luego de capturar el texto de dicho elemento, este se mostrará en el segundo elemento <p>.

Aplicar una decoración CSS al elemento <a>

**Ejercicio 10:** Crear un nuevo documento HTML y, dentro de este, agregar un elemento DIV. Luego crear una función JavaScript, que invocará al evento createElement() para crear un nuevo elemento <p>. Tener en cuenta:

* Al integrar un elemento DIV, este deberá contar con un ID.
* Las funciones getElementById() y appendChild() se deben combinar para obtener el resultado deseado.
* Si no se controla el agregado de nuevos elementos, estos desbordaran el DIV.

**Ejercicio 11:** Este ejercicio debe contar con las siguientes características:

* Desarrollar una página web que muestre una imagen mediana al 20% de su tamaño original.
* Esta imagen deberá tener dos funciones JavaScript, una en el evento onMouseOver() y otra en el evento onMouseOut().
* Los eventos mencionados deben recibir un parámetro, que será el objeto imagen.
* Cuando se posiciona el cursor del mouse sobre la imagen, se ejecutará el primer evento, mientras que, al retirar el cursor, se ejecutará el segundo evento.
* El evento correspondiente a onMouseOver debe redimensionar la imagen al 50 o 100% de su valor.
* El evento correspondiente a onMouseOut devuelve su imagen al estado original.

**Ejercicio 12:** Agregar dos imágenes de manera que, al hacer clic obre estas, nos redireccionen a un sitio web determinado, a través de la propiedad href. Crear una función JS que se invocará desde cada imagen, pasándole como parámetro la URL hacia donde redireccionamos al usuario.

**Ejercicio 13:** Crear dos documentos HTML con su estructura básica: ejercicio.html y ventanaemergente.html. El ejercicio presentará una página HTML simple (ejercicio.html), con un elemento button que, al ser presionado, abrirá una ventana en modo popup (ventanaemergente.html) con determinadas características. En este último documento HTML, se incluirá un elemento button que invocará a una función JS para cerrar automáticamente la ventana.

**Ejercicio 14:** Probar el comportamiento de un formulario web con el atributo autofocus, modificando el proyecto actual y agregando dicho atributo en input type=”nombre”. Probar el comportamiento de dicho proyecto y, luego, elegir otro de los campos para que tome el foco inicial. La tercera modificación que se realizará en este ejercicio consiste en elegir dos de todos los inputs types existentes e ingresar en ambos el atributo autofocus. Ejecutar la página y ver que sucede. Por último, agregar el atributo autofocus solamente en input type=”submit”, y probar otra vez el comportamiento de la página.

**Ejercicio 15:** Desarrollar un ejercicio que contendrá un formulario HTML y dos inputs type del tipo email. En el primer input type integrar el atributo pattern, con el siguiente formato de caracteres: pattern=“[^ @]\*[^ @]\*”. Luego agregar un segundo input type email, en el cual se debe integrar el mismo atributo, pero cambiando el formato de validación por el siguiente: pattern=“[a-zA-Z]{3,}@[a-zA-Z]{3,}[.]{1} [a-zA-Z]{2,}[.]{1}[a-zA-Z]{2,}”. Probar el ingreso de diferentes direcciones de correo electrónico en ambos campos.

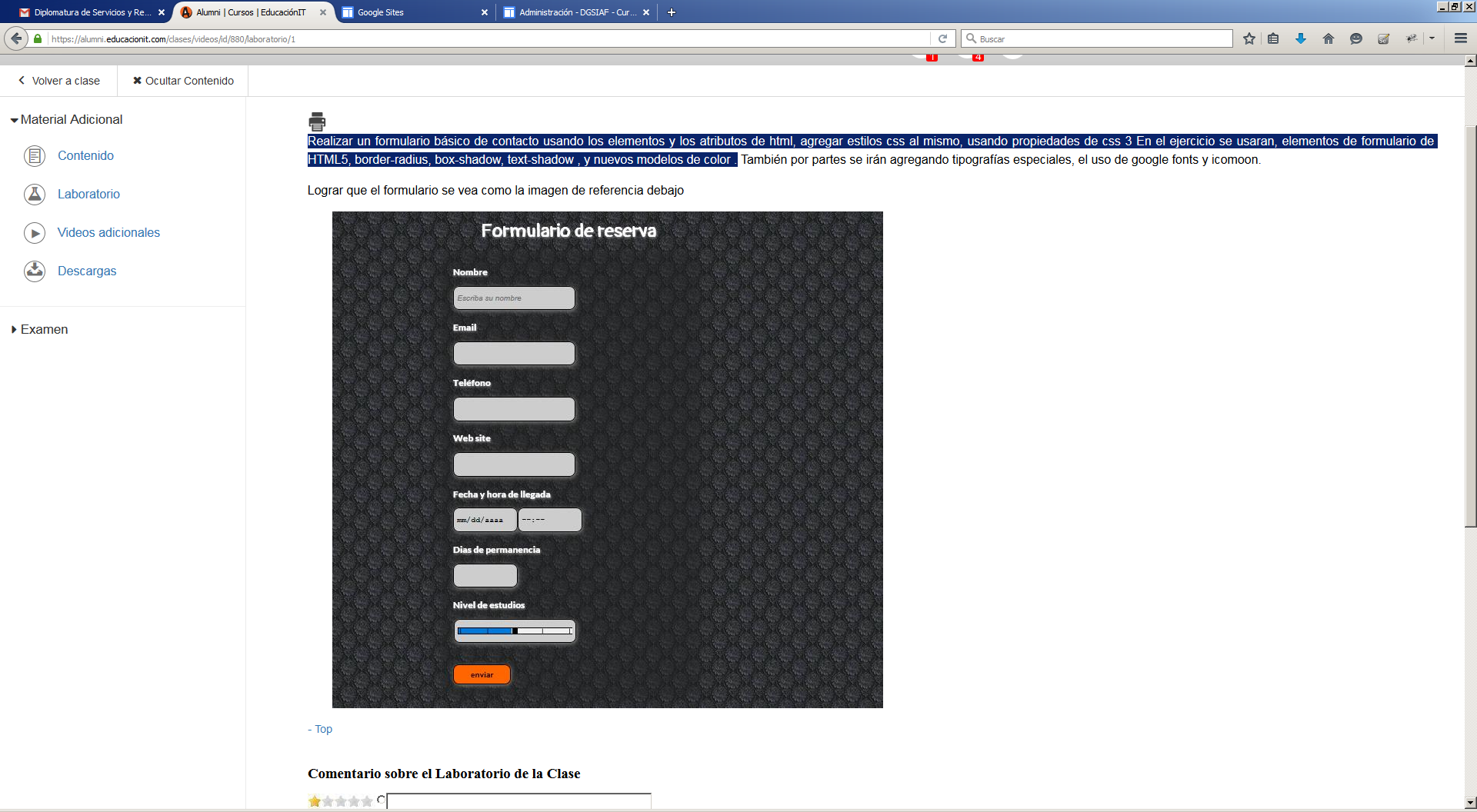
**Ejercicio 16:** La consigna del siguiente ejercicio es poder tildar múltiples checkboxes y que, al presionar el botón Aceptar, un mensaje devuelva la cantidad de elementos que se ha tildado. Tener en cuenta los siguientes puntos:

. Integrar en el formulario una tabla que permita estructurar 12 checkboxes, divididos en 3 columnas.

* Agregar al archivo HTML una referencia a un archivo JS externo.
* El archivo JS debe contener una función que cuente los checkboxes tildados, y solo muestre un mensaje de alerta cuando haya checkboxes tildados.
* Modificar el proyecto, para que todos los elementos checkbox aparezcan tildados por defecto cuando la página HTML carga.

**Ejercicio 17:** Construir un formulario web desde cero. Utilizar el método GET y un segundo archivo HTML, que contendrá el código JS que procese los datos enviados por el formulario web. Finalmente, el archivo HTML invocado mostrará en pantalla los datos procesados, cargados por el usuario.

**Ejercicio 18:** Realizar un formulario básico de contacto usando los elementos y los atributos de HTML5, agregar estilos CSS al mismo, usando propiedades de CSS3. En el ejercicio se usarán, elementos de formulario de HTML5, border-radius, box-shadow, text-shadow, y nuevos modelos de color.

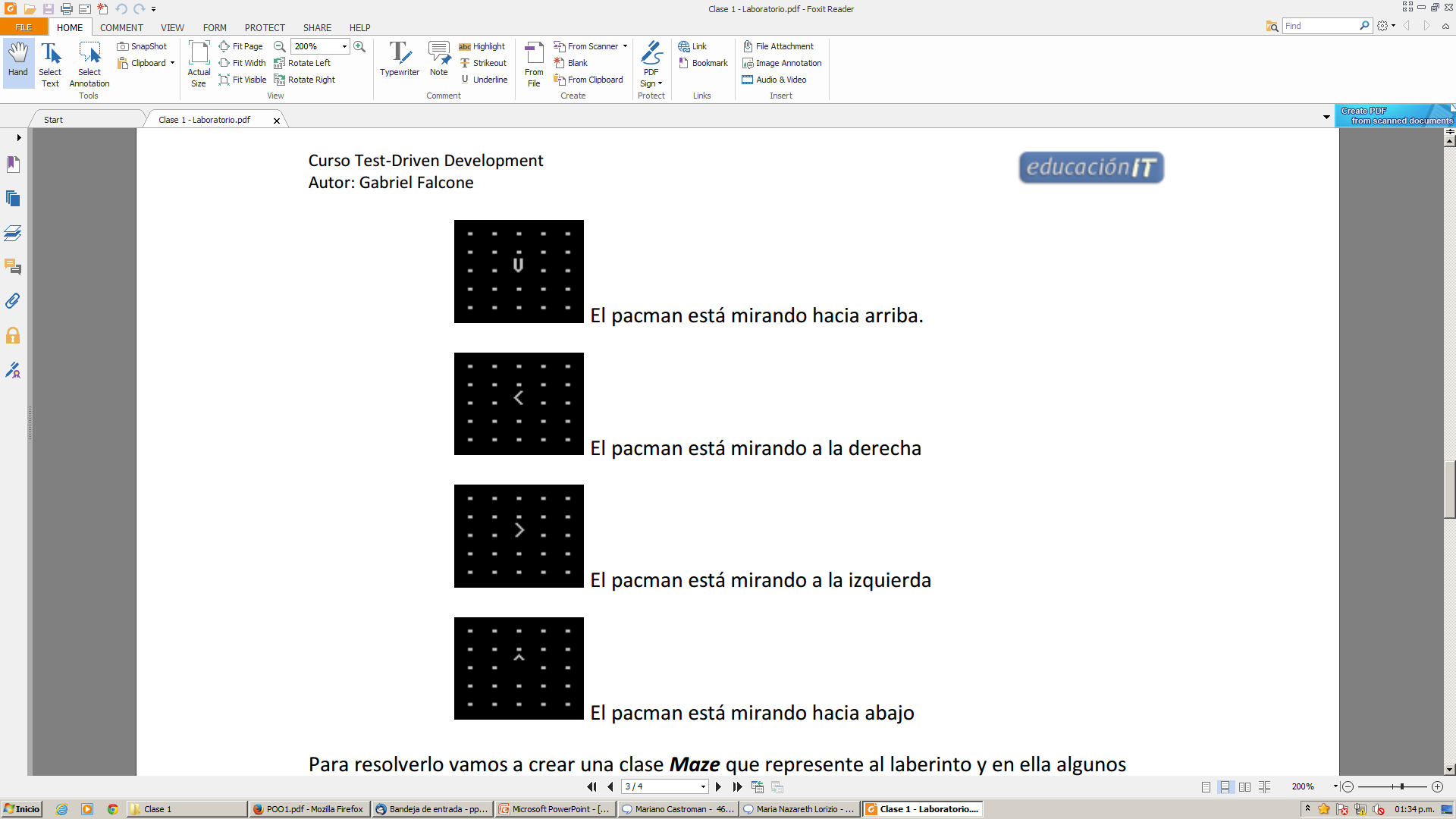
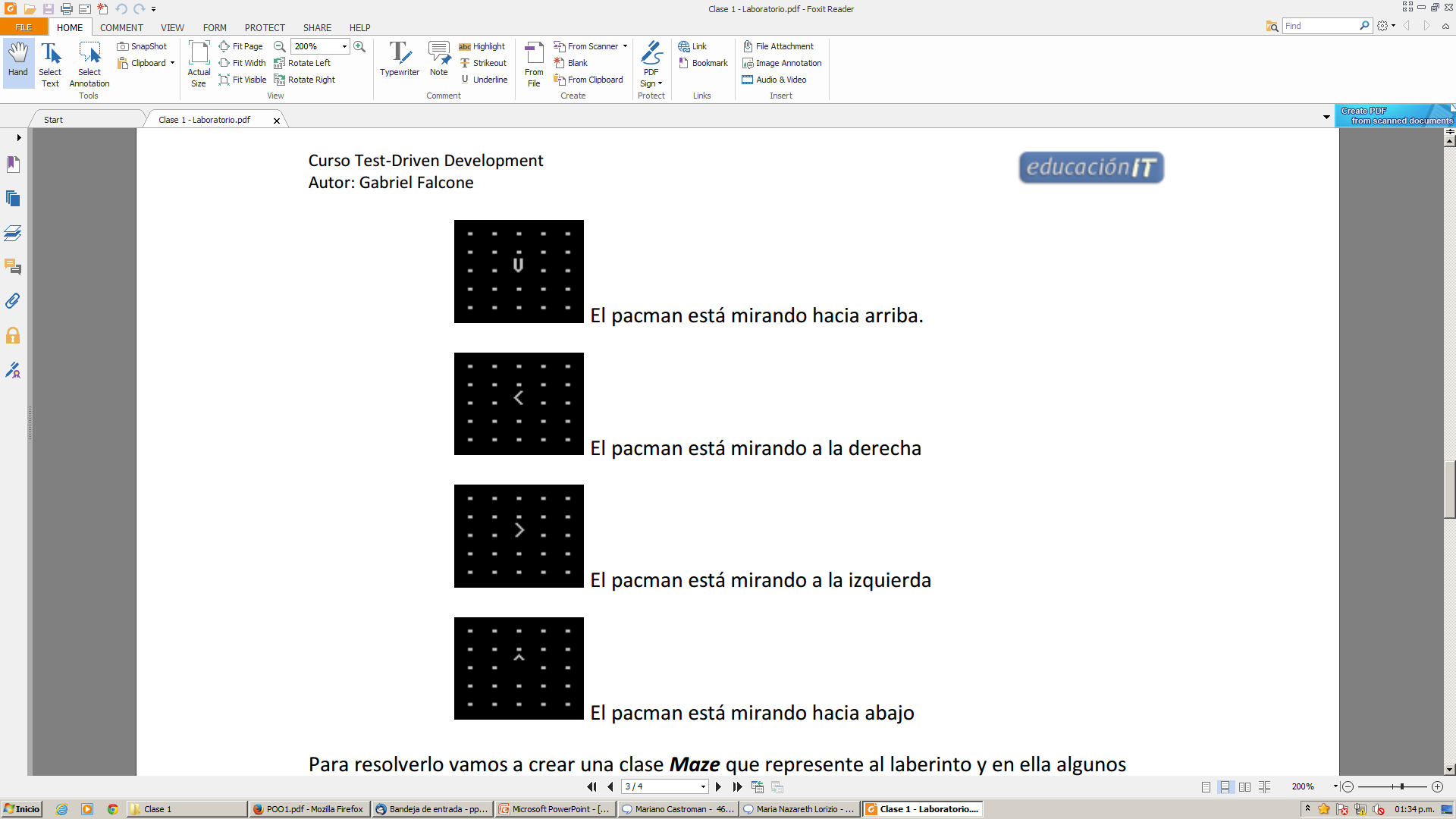


**Ejercicio 19:** Crear una página web y pedir al usuario que adivine un número en 3 intentos.

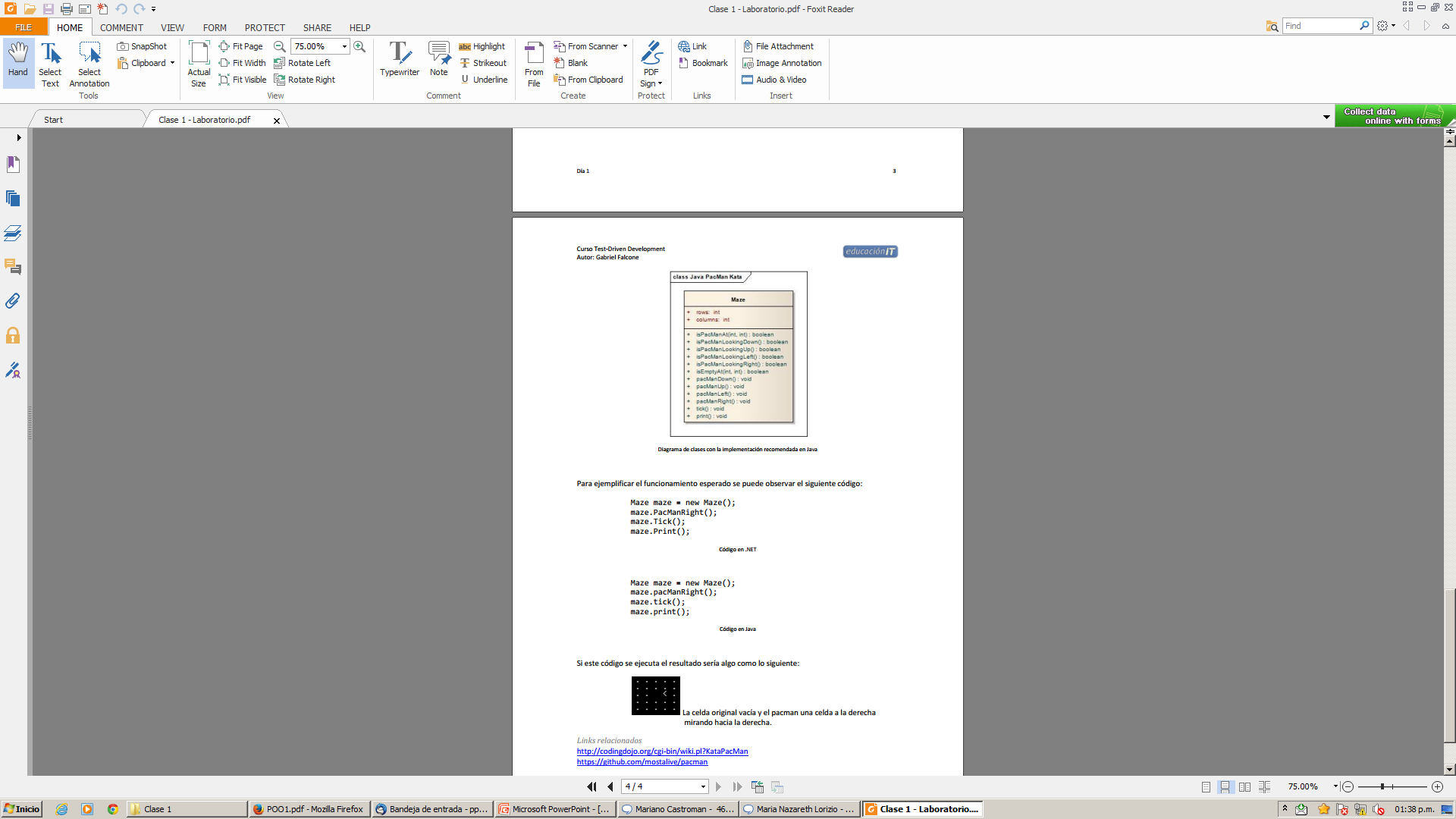
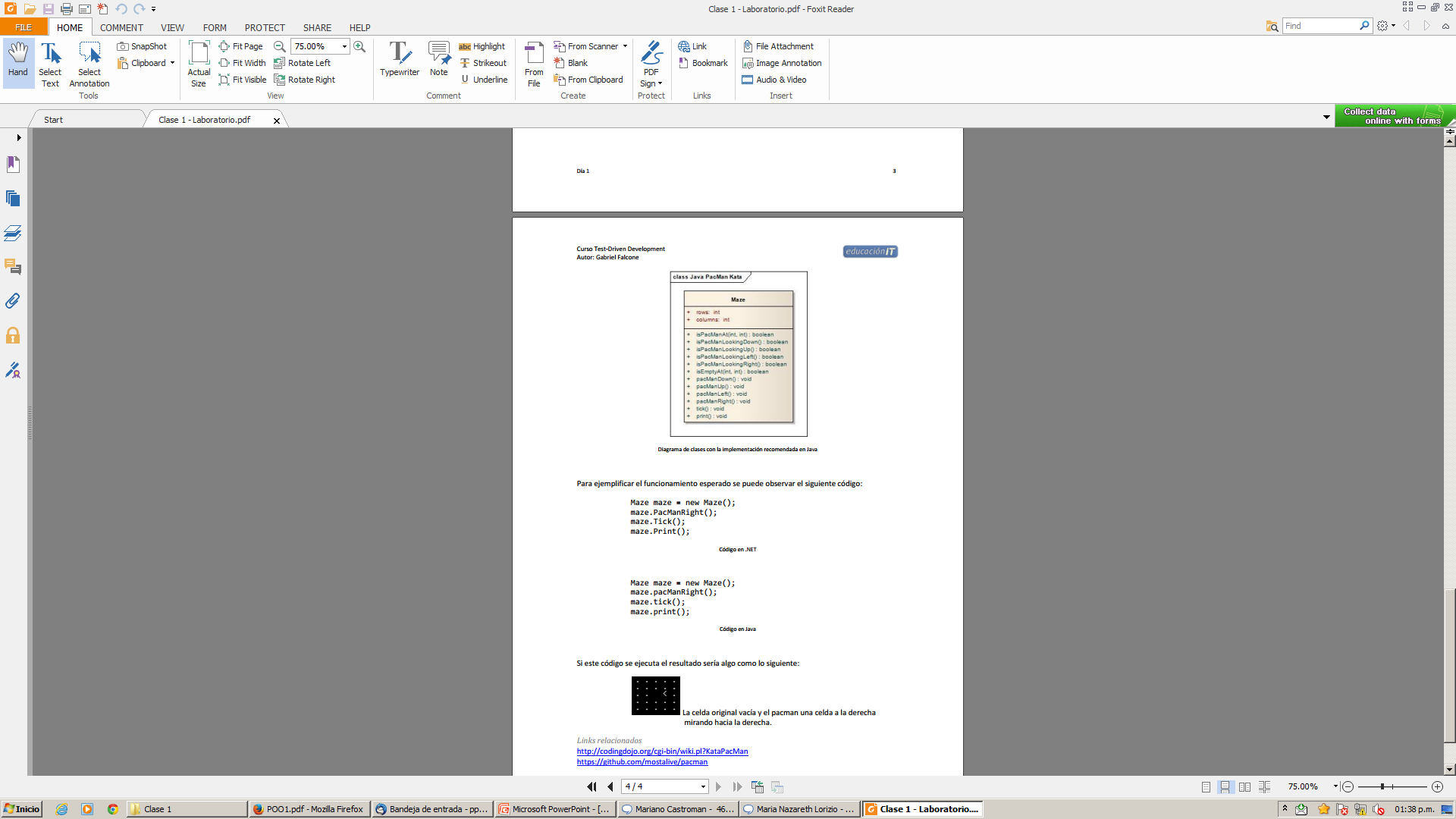
**Ejercicio 20:** FizzBuzz: Imprimir por pantalla los números del 1 al N pero considerando lo siguiente:

* N entra como dato en el constructor.
* Si el número es divisible por 3 se debe imprimir “Fizz”.
* Si el número es divisible por 5 se debe imprimir “Buzz”.
* Si el número es divisible por 3 y por 5 se debe imprimir “FizzBuzz”.
* Sino entonces imprimir el número.

**Ejercicio 21:** Defina una clase 'Monedero' que permita gestionar la cantidad de dinero de que una persona dispone en un momento dado. La clase deberá tener un constructor que permitirá crear un monedero con una cantidad de dinero inicial y deberá definir un método para meter dinero en el monedero, otro para sacarlo y finalmente, otro para consultar el disponible; solo podrá conocerse la cantidad de dinero del monedero a través de este último método. Por supuesto, no se podrá sacar más dinero del que haya en un momento dado en el monedero. Probar el funcionamiento de la clase.

**Ejercicio 22:** Se plantea una simplificación del laberinto del juego del pacman representado por una secuencia de caracteres como la siguiente:

Para resolverlo se va a crear una clase Maze que represente al laberinto y en ella algunos métodos que permitan mover al pacman en distintas direcciones, métodos para saber si el pacman está en cierta posición o con cierta dirección, un método que permite simular el paso de un instante de tiempo y un método que muestre en la consola el estado del laberinto. Una posible solución está representada por el siguiente diagrama:

****

**Ejercicio 23:** Ingresar un color por el prompt y cambiar el color del documento.

**Ejercicio 24**: Ingresar un número por el prompt y si es menor que 100 mostrar imagen ok.png caso contrario error.png. Las imágenes mostrarlas dentro de un div, modificando con innerHTML.

**Ejercicio 25:** Mostrar en un div el dia de hoy con el siguiente formato:

Hoy es Viernes 19 de octubre de 2018.

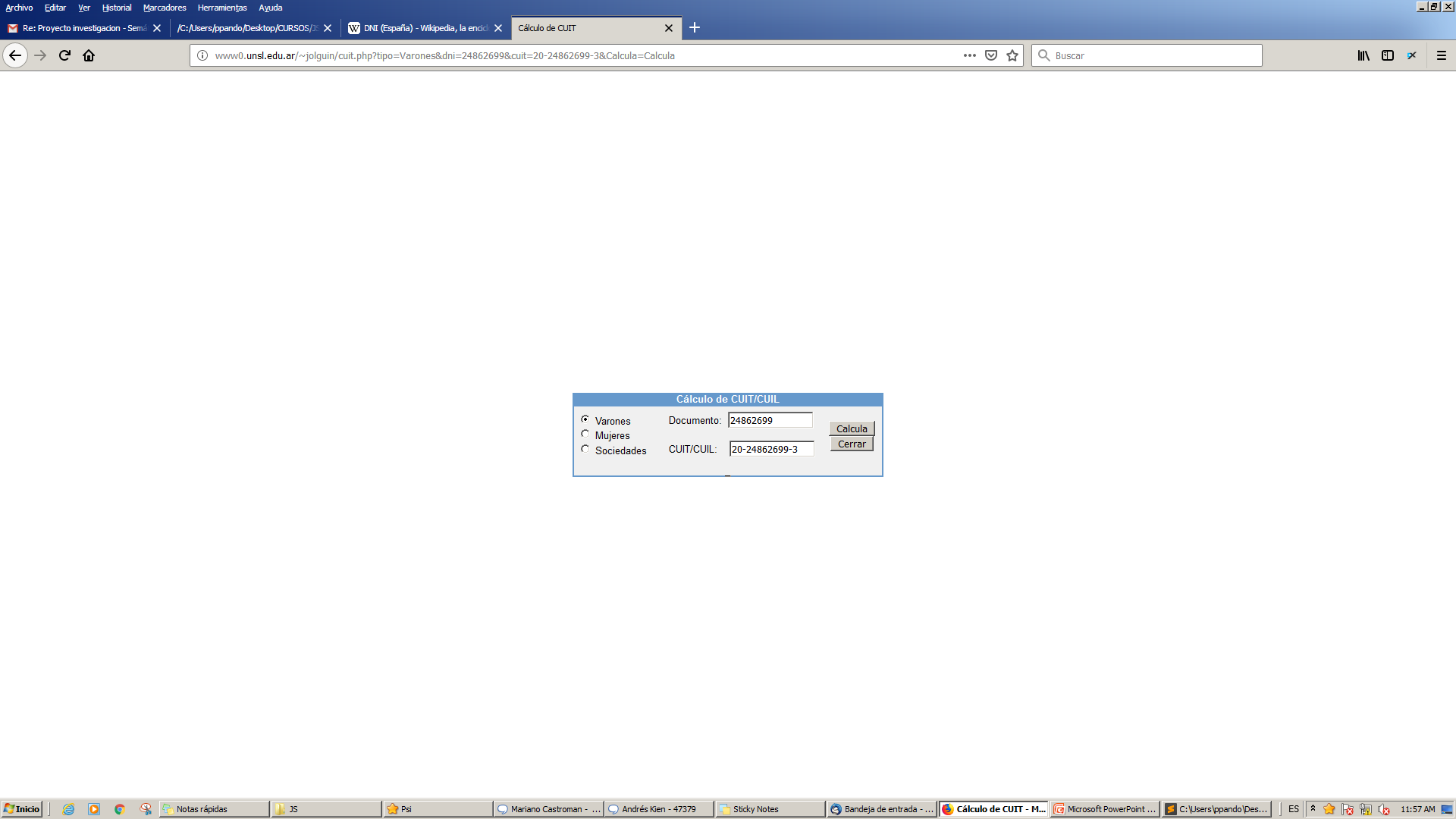
**Ejercicio 26:** Armar una página HTML con dos elementos <p>, y un hipervínculo. Las características adicionales del ejercicio son las siguientes:

* Debe tener un título h1.
* Debe contener dos elementos párrafos.
* Debe contener un hipervínculo.
* Debe existir una función JS dentro de la misma página.
* Entonces al presionar el hipervínculo, se deberá ejecutar la función JS. Dentro de la función JS, se declara una variable, la cual debe almacenar el texto especificado en el primer elemento <p>. Luego de capturar el texto de dicho elemento, este se mostrará en el segundo elemento <p>.

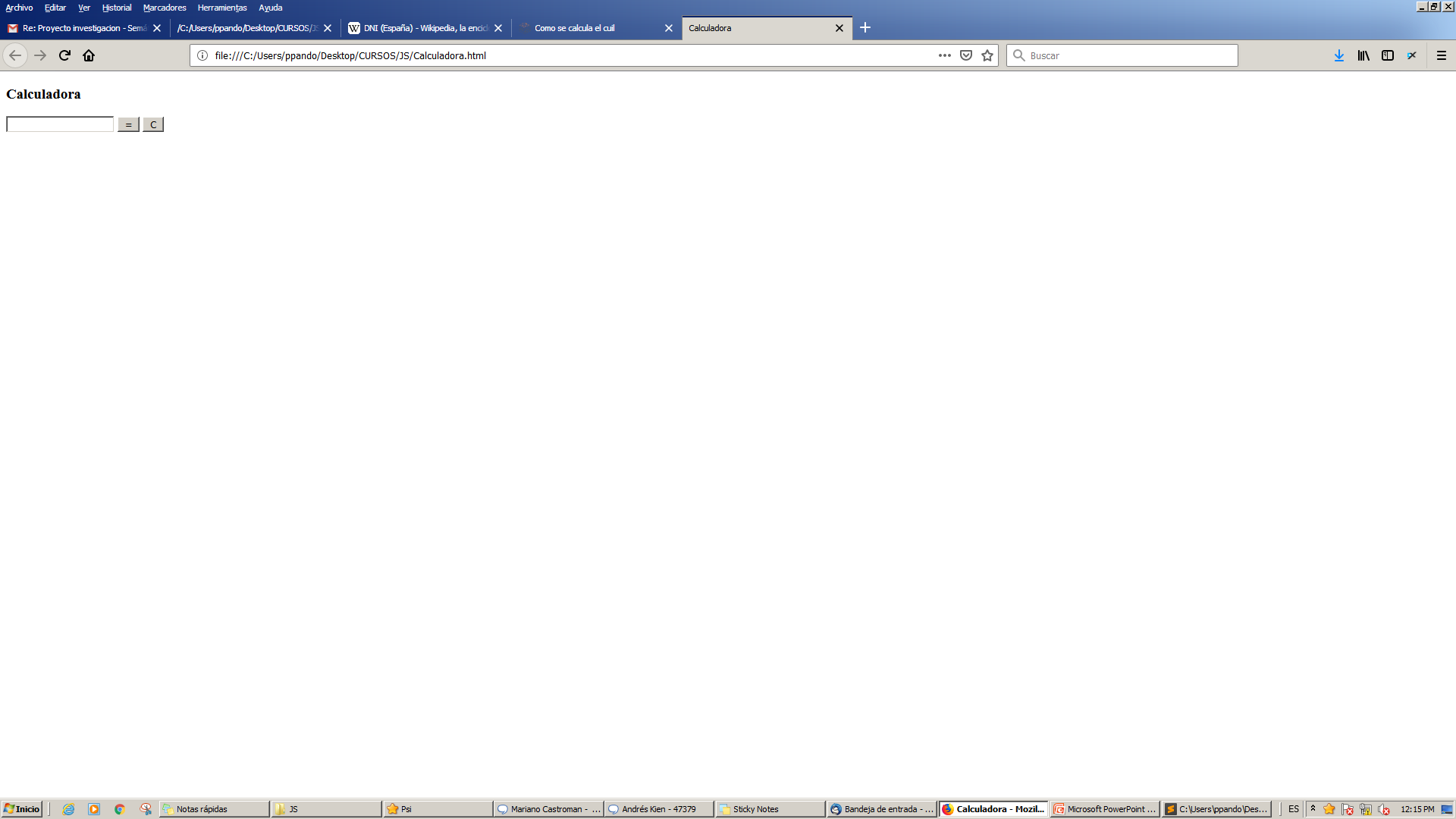
**Ejercicio 27:** Mostrar un texto rotativo: simular el efecto de desplazar un texto horizontalmente hacia la izquierda a modo de letrero informativo. Los elementos a usar son:

* Una capa HTML (<div>) que contendrá el mensaje.
* Una función JavaScript que hará rotar el texto.
* El efecto cesará cuando se ponga el puntero del mouse encima de la capa, para poder leer el texto más cómodamente, y se reanudará cuando se lo quite.

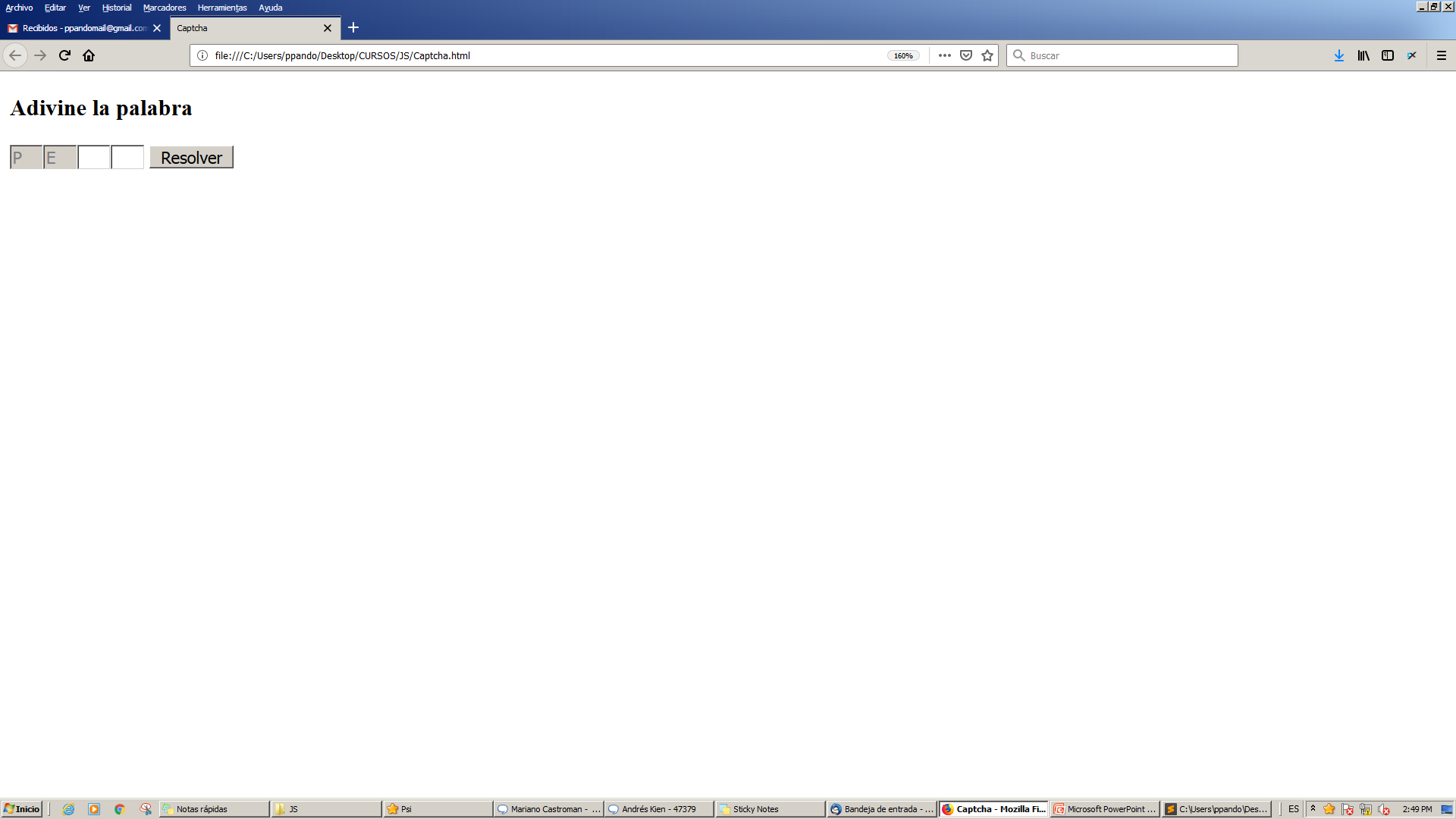
**Ejercicio 28:** Calcular CUIT/CUIL: calcular automáticamente el número de CUIT/CUIL luego que de que el usuario haya introducido su DNI.



**Ejercicio 29:** Calculadora básica: para realizar operaciones sencillas, como sumas o divisiones. Los elementos a usar son: Un formulario HTML (<form>) que constará de una caja de texto, que simulará la pantalla, y dos botones: uno para realizar el cálculo de la operación que haya en la pantalla, y otro para borrar el resultado y poder introducir una nueva operación.

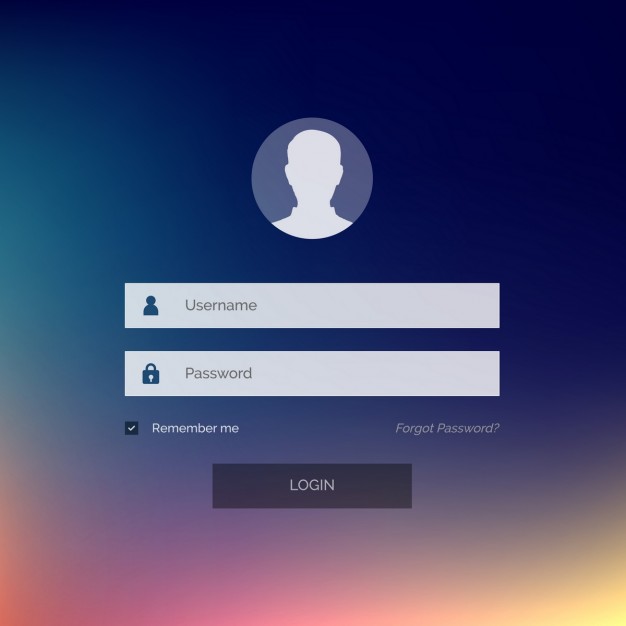


**Ejercicio 30:** Captcha: acertar una palabra antes de tener acceso al resto de las páginas. Los elementos a usar son: Un formulario HTML (<form>) que constará de varias cajas de texto con las letras, tanto las visibles como las que deberá rellenar el usuario, y un botón para comprobar si ha acertado o no. Como se momento no se sabe qué palabra se va a mostrar, no se pondrá ningún campo <input> y se dejará que el código se encargue de hacerlo.

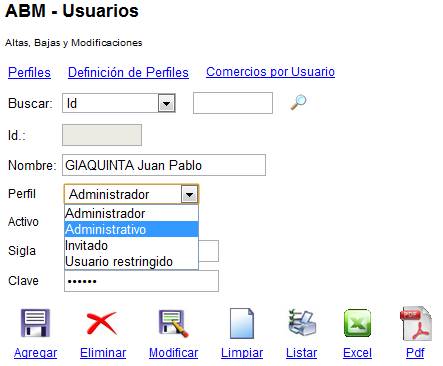


|  |  |
| --- | --- |
| **Asignatura** | **Programación 3** |
| **Carrera** | **Tec. en Programación de Computadoras** |
| **Plan** | **2014** |
| **Ciclo** | **2021** |
| **Cuatrimestre** | **1ero** |
| **Trabajo Práctico** | **2** |
| **Tema** | **Desarrollo Backend** |
| **Tipo de Práctica** | **Formación Experimental (P1) – ejercicios 1 a 5** |

**Ejercicio 1:** Login



**Ejercicio 2:** ABM de Usuarios



**Ejercicio 3:** Búsqueda y listado de elementos a definir con los alumnos.

**Ejercicio 4:** Validadores de objeto a definir con los alumnos.

**Ejercicio 5:** Gestión de objeto a definir con los alumnos.