*2DA ENTREGA DEL PROYECTO FINAL*

**Componentes:**

1. Estructura HTML del simulador
2. Estilos iniciales del simulador
3. Script del simulador

*Estructura inicial del Simulador*

* **Formato:** Archivos HTML
* **Objetivos del desafío:**
  + El estudiante deberá utilizar el tag correspondiente para asociar un script JS a una página HTML.
  + Construir la interfaz inicial del simulador, definiendo elementos destinados a la captura de entradas (formularios, inputs y/o botones) y secciones en el HTML para la notificación de salidas .

*Estilos iniciales del Simulador*

* **Formato:** Archivos CSS
* **Objetivos del desafío:**
  + El estudiante deberá declarar reglas CSS de estilos con el objetivo de otorgar una correcta visualización de entradas y salidas, guardando coherencia con el contexto del simulador escogido.En caso de no contar con conocimientos previos en HTML y CSS es posible emplear un framework para estilar la interfaz del simulador.

*Script del Simulador*

* **Formato:** Archivo JS
* **Objetivo del desafío:**
  + El estudiante deberá codificar las funcionalidades mínimas del simulador. Identificando el flujo de trabajo en el script en términos de captura de entradas y eventos de usuario, funciones de procesos esenciales y notificación de resultados en forma de salida por HTML, modificando el DOM.
  + En caso de requerir un volumen de información estàtica significante, el estudiante deberá emplear objetos literales y/o parseo JSON, tanto para obtener como para almacenar datos.

*RÚBRICA DE EVALUACIÓN*

*2DA ENTREGA DEL PROYECTO FINAL*

| **RÚBRICA: CRITERIOS PARA LA 2DA ENTREGA DEL PROYECTO FINAL** | | | |
| --- | --- | --- | --- |
| **Criterios** | **Bajo**  Falta más profundización. Es confuso. | **Correcto**  Acorde pero con errores puntuales. | **Óptimo**  Es claro y pertinente. |
| Funcionalidad | * No se simula un flujo de trabajo en términos de entrada, proceso y salida. * Entendemos a un flujo de trabajo como una secuencia ordenada de tareas dedicadas a solventar un problema. * Entradas: Uno o más valores ingresados por el usuario. * Salidas: Uno o más mensajes mostrados al usuario. * Proceso: Algoritmo cuyas instrucciones transforman las entradas en salidas esperadas. | * Se simula un flujo de trabajo en términos de entrada,proceso y salida. * La funcionalidad es apropiada al contexto del simulador. | * Se simula uno o más flujos de trabajo en términos de entrada,proceso y salida. * La funcionalidad es apropiada es apropiada al contexto del simulador. * No se advierten errores de cómputo durante el procesamiento. |
| Interactividad | * No se capturan entradas ingresadas por el usuario mediante eventos. * No se efectúan salidas por HTML modificando el DOM. | * Se capturan entradas ingresadas por el usuario mediante eventos. * Se efectúan una o más salidas por HTML modificando el DOM. | * Se capturan entradas ingresadas por el usuario mediante eventos. * Se efectúan una o más salidas por HTML modificando el DOM. * Existe un control de ingreso de entradas y las salidas son coherentes en relación a los datos ingresados. |
| Escalabilidad | * No se emplean funciones. * No se utiliza Array para para agrupar datos símiles. * No se utilizan objetos para estructuras de datos con un mismo comportamiento. * No se establece un criterio homogèneo para la detecciòn de eventos. | * Se declaran funciones con parámetros para definir instrucciones con una tarea específica. * Se emplean arrays para agrupar valores relacionados. * Se definen objetos con propiedades y métodos relevantes al contexto. * Se establece un criterio homogèneo para la detecciòn de eventos | * Se declaran funciones con parámetros para definir instrucciones con una tarea específica. * Se emplean arrays para agrupar valores relacionados. * Se definen objetos con propiedades y métodos relevantes al contexto. * Se establece un criterio homogèneo para la detecciòn de eventos. * Se almacena en storage datos relevantes generados durante la simulaciòn. |
| Integridad | * Se define el código JavaScript dentro del documento HTML. | * Se define el código JavaScript en un archivo .js, referenciándolo correctamente desde el HTML. * No se emplean métodos prompt() y alert() para evitar interrupciones durante el procesamiento y actualización del DOM. | * Se define el código JavaScript en un archivo .js, referenciándolo correctamente desde el HTML. * No se emplean métodos prompt() y alert() para evitar interrupciones durante el procesamiento y actualización del DOM. * La información estática del proyecto se emplea adecuadamente. |
| Legibilidad | * No se establece un criterio de claridad en la definición de variables, funciones y objetos. | * Los nombres de variables. funciones y objetos son significativos para el contexto. * Las instrucciones se escriben de forma legible y se emplean comentarios oportunos. * Entendemos al código legible como aquel que está correctamente estructurado, preferentemente alineado e indentado. | * Los nombres de variables. funciones y objetos son significativos para el contexto . * Las instrucciones se escriben de forma legible y se emplean comentarios oportunos. * El código fuente es ordenado en términos de declaración y secuencia. |