**UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS**

FACULTAD DE INGIENERÍA DE SISTEMAS E INFORMÁTICA

*E.A.P. INGIENERÍA DE SISTEMAS*



**TITULO:**

ALGORITMOS DE ORDENAMIENTO

**CURSO:**

Análisis y diseño de algoritmos

**PROFESOR:**

Herminio Paucar Curasma

**PRESENTADO POR:**

* Varillas Meléndez José Luis
* Silva Jaramillo Yamile
* Salcedo Patroni Andrés Gabriel
* Barrios Peña Ernesto Zenobio
* Valenzuela Martínez Alessandro Gregory

**LIMA - PERÚ**

**2020**

**INTRODUCCIÓN**

El presente proyecto gira en torno a la implementación de una pagina web que ayude a entender, mediante gráficos animados, los siguientes algoritmos de ordenamiento: Bubble sort, Merge sort y Radix sort. Para realizarlo nos apoyamos en los lenguajes de marcado HTML y CSS, además también utilizamos el lenguaje de programación JavaScript y algunas librerías que nos facilitan el trabajo.

A lo largo de los años los algoritmos de ordenamiento se han explicado de diferentes maneras, cada una de ellas con sus diferentes ventajas y desventajas. Pero la dificultad real radica en explicar un proceso tan dinámico y cambiante sin las herramientas necesarias y que hasta hace algún tiempo no estaban al alcance de todos.

Desde hace un par de décadas tenemos estas herramientas cada vez más cerca y poco a poco han ido evolucionando mediante estándares que las hacen más fáciles de aprender y manejar, tanto que su uso en el desarrollo web es casi obligatorio para crear un buen proyecto, ya que estos se conectan muy bien al tener un fin muy bien definido (HTML para la estructura, CSS para el diseño y JavaScript para el funcionamiento),

Hoy en día los lenguajes antes mencionados nos permiten llevar estos algoritmos de ordenamiento a una pantalla para poder explicar el proceso gráfica y dinámicamente y así conocer su funcionamiento exacto para compararlos y analizarlos e incluso mejorarlos.

**MARCO TEORICO**

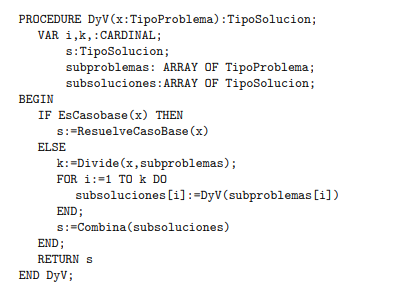
**DIVIDE Y VENCERAS**

Técnica de diseño de algoritmos que consiste en resolver un problema a partir de la solución de subproblemas del mismo tipo, pero de menor tamaño. Si los subproblemas son todavía relativamente grandes se aplicará de nuevo esta técnica hasta alcanzar subproblemas lo suficientemente pequeños para ser solucionados directamente. Ello naturalmente sugiere el uso de la recursión en las implementaciones de estos algoritmos.

La resolución de un problema mediante esta técnica consta fundamentalmente de los siguientes pasos:

1. En primer lugar ha de plantearse el problema de forma que pueda ser descompuesto en k subproblemas del mismo tipo, pero de menor tamaño. Es decir, si el tamaño de la entrada es n, hemos de conseguir dividir el problema en k subproblemas (donde 1 ≤ k ≤ n), cada uno con una entrada de tamaño y donde 0 ≤ < n. A esta tarea se le conoce como división.
2. En segundo lugar han de resolverse independientemente todos los subproblemas, bien directamente si son elementales o bien de forma recursiva. El hecho de que el tamaño de los subproblemas sea estrictamente menor que el tamaño original del problema nos garantiza la convergencia hacia los casos elementales, también denominados casos base.
3. Por último, combinar las soluciones obtenidas en el paso anterior para construir la solución del problema original.

El funcionamiento de los algoritmos que siguen la técnica de Divide y Vencerás descrita anteriormente se refleja en el esquema general que presentamos a continuación:

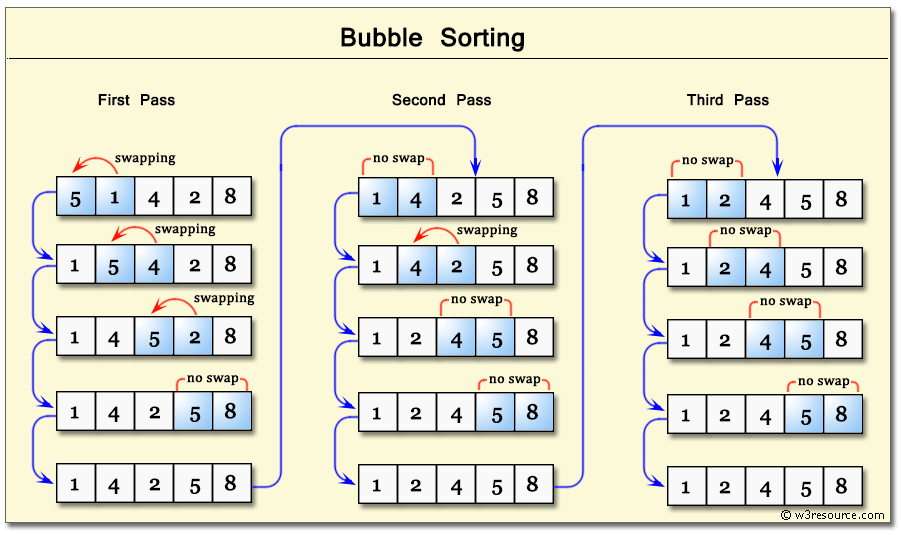


**ALGORITMO DE ORDENAMIENTO**

Un algoritmo de ordenamiento es un algoritmo que pone elementos de una lista o un vector en una secuencia dada por una relación de orden, es decir, el resultado de salida ha de ser una permutación —o reordenamiento— de la entrada que satisfaga la relación de orden dada. Las relaciones de orden más usadas son el orden numérico y el orden lexicográfico. Ordenamientos eficientes son importantes para optimizar el uso de otros algoritmos (como los de búsqueda y fusión) que requieren listas ordenadas para una ejecución rápida. También es útil para poner datos en forma canónica y para generar resultados legibles por humanos.

**Bubble Sort**

Bubble Sort es un algoritmo de ordenamiento simple que recorre repetidamente la lista, compara pares adyacentes y los intercambia si están en el orden incorrecto. El paso a través de la lista se repite hasta que se ordena la lista. El algoritmo, que es un tipo de comparación, recibe su nombre de la forma en que los elementos más pequeños o más grandes "burbujean" en la parte superior de la lista. Aunque el algoritmo es simple, es demasiado lento y poco práctico para la mayoría de los problemas, incluso en comparación con la ordenación por inserción. La clasificación de burbujas puede ser práctica si la entrada está mayoritariamente ordenada con algunos elementos desordenados casi en posición. Tiene un peor caso y una complejidad promedio de [*О*](https://es.qwe.wiki/wiki/Big_o_notation)( ),

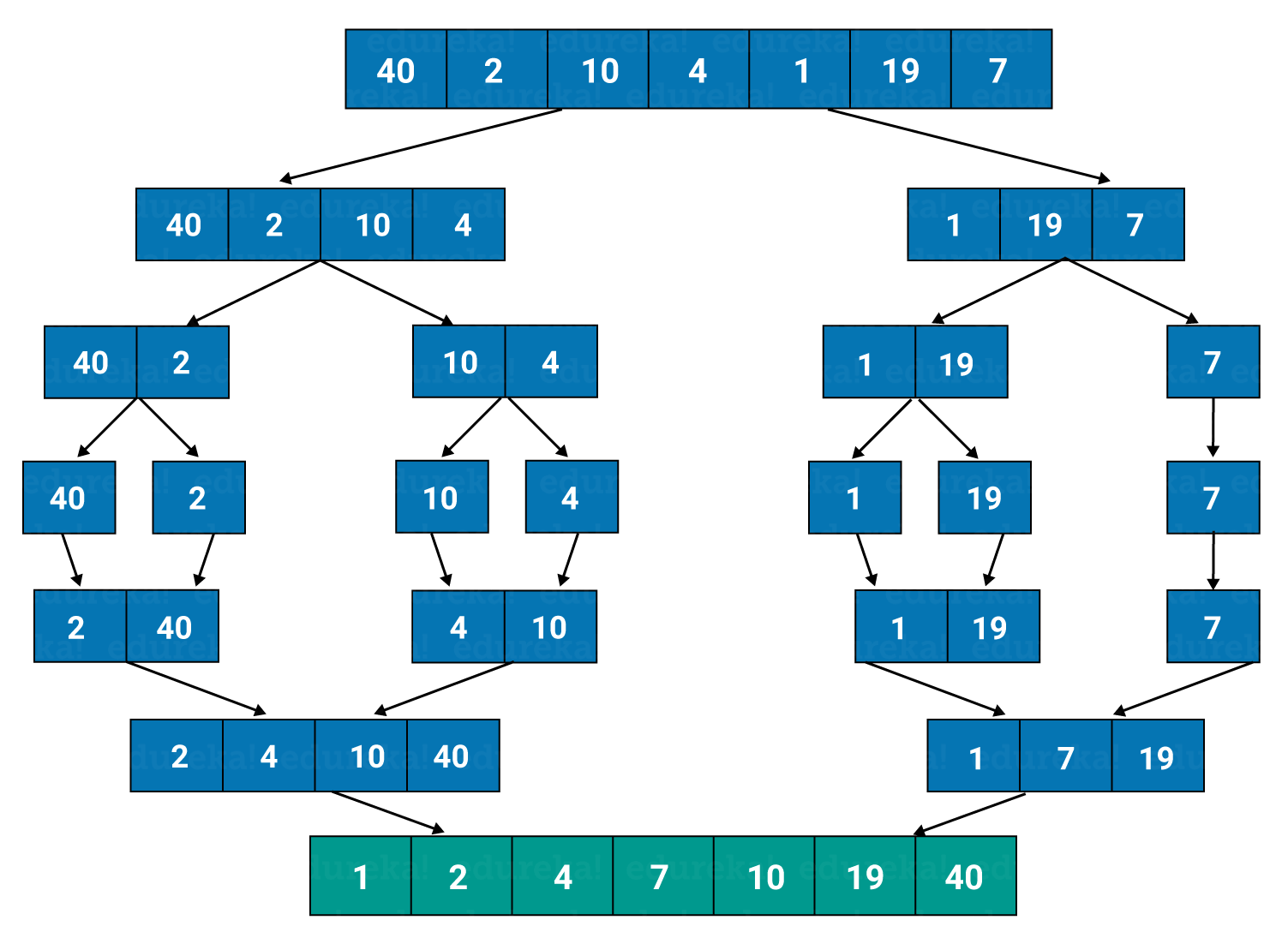


**Merge Sort**

Fue desarrollado en 1945 por John Von Neumann. Es un algoritmo de ordenamiento externo estable basado en la técnica divide y vencerás. Es de complejidad O(n log n).

Conceptualmente, el ordenamiento por mezcla funciona de la siguiente manera:

1. Si la longitud de la lista es 0 ó 1, entonces ya está ordenada. En otro caso:
2. Dividir la lista desordenada en dos sublistas de aproximadamente la mitad del tamaño.
3. Ordenar cada sublista recursivamente aplicando el ordenamiento por mezcla.
4. Mezclar las dos sublistas en una sola lista ordenada.



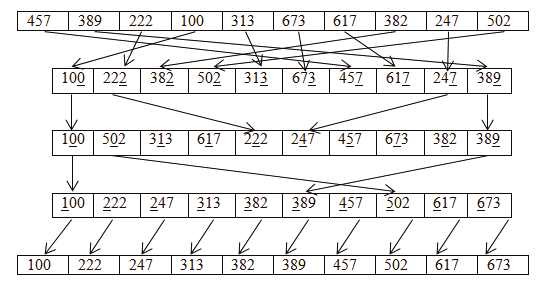
**Radix Sort**

Algoritmo de ordenamiento que ordena enteros procesando sus dígitos de forma individual. Como los enteros pueden representar cadenas de caracteres (por ejemplo, nombres o fechas) y, especialmente, números en punto flotante especialmente formateados, radix sort no está limitado sólo a los enteros. Existen dos clasificaciones de radix sort: el de dígito menos significativo (**LSD**) y el de dígito más significativo (**MSD**). *Radix sort LSD* procesa las representaciones de enteros empezando por el dígito menos significativo y moviéndose hacia el dígito más significativo.

Funcionamiento:

1. Haga lo siguiente para cada dígito i donde i varía desde el dígito menos significativo hasta el dígito más significativo.
   1. Ordene el array de entrada utilizando Counting Sort (o cualquier ordenamiento estable) de acuerdo con el i-ésimo dígito.

Radix Sort toma O (d \* (n + b)) tiempo donde b es la base para representar números y d dígitos en los enteros de entrada. Si k es el valor máximo posible, entonces d sería O (). Entonces, la complejidad general del tiempo es O ((n + b) \* ). Lo que se ve más que la complejidad temporal de los algoritmos de clasificación basados ​​en la comparación para un k grande.Sea k <= nc donde c es una constante. En ese caso, la complejidad se convierte en O (n). ). Pero todavía no supera a los algoritmos de clasificación basados ​​en comparación.  
Si establecemos b como n, obtenemos la complejidad de tiempo como O (n). Podemos ordenar un array de números enteros con un rango de 1 a si los números están representados en base n (o cada dígito toma bits).



**GitHub**

GitHub es una forja (plataforma de desarrollo colaborativo) para alojar proyectos utilizando el sistema de control de versiones Git. Se utiliza principalmente para la creación de código fuente de programas de ordenador. El software que opera GitHub fue escrito en Ruby on Rails. Desde enero de 2010, GitHub opera bajo el nombre de GitHub, Inc. Anteriormente era conocida como Logical Awesome LLC. El código de los proyectos alojados en GitHub se almacena típicamente de forma pública.

El 4 de junio de 2018 Microsoft compró GitHub por la cantidad de 7500 millones de dólares​, un cambio de propietarios que ha provocado la salida de varios proyectos de este repositorio debido la posibilidad de acceso a códigos fuentes por parte de una compañía que basa uno de sus principales negocios en el software.

**Página web :**

La unidad básica del World Wide Web es la principal plataforma en donde se desarrolla nuestro proyecto, cada sitio web está compuesto por un conjunto de páginas web. Es aquí donde toda la codificación e implementación se muestra así como la interfaz gráfica.

**HTML :**

Una de las herramientas para desarrollar el presente proyecto, fue el uso de HTML. Este lenguaje está orientado a indicar como va ordenado el contenido de una página web. Esto se hace mediante el uso de marcas de hipertexto conocidas como etiquetas o tags.

**CSS :**

Para seguir en el desarrollo del proyecto, no basta con conocer HTML. Es imprescindible combinar otras tecnologías como CSS (*Cascading Style Sheets*), ya que esta herramienta nos permite crear animaciones, estilizar elementos de nuestra página web, modificar textos, colores y más.La implementación de CSS nos ayuda a llevar un mejor control de la interfaz y todo el contenido de texto que lleva nuestro sitio web.

**JavaScript :**

Uno de los lenguajes más importantes para el desarrollo web es JavaScript. Esta herramienta ha permitido una diversificación de características interactivas a las páginas web, para el desarrollo de nuestro proyecto esta es la tercera tecnología que se requiere para la implementación de la lógica de los programas de ordenamiento.

**Aplicación web :**

Tipo de software que utiliza como herramientas HTML,CSS o JavaScript y funcionan en todos los sistemas operativos y terminales que cuentan con los navegadores web que las soportan (en general, todos los navegadores actuales).

**METODOLOGIA**

Para la realización y presentación de este proyecto se empleó el uso de herramientas como son el Visual Studio Code, SublimeText y GitHub. Así mismo, usando de guía aplicaciones similares.

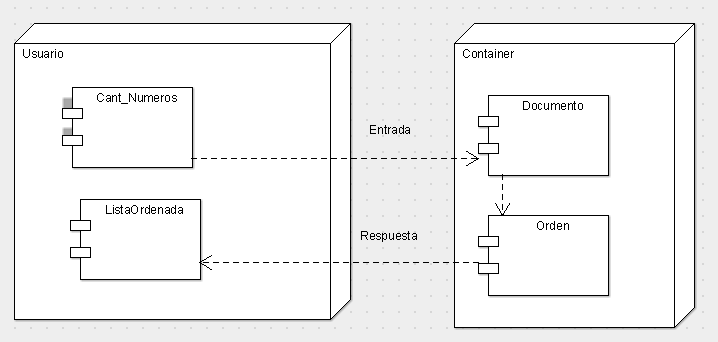
Este proyecto cuyo alcance es realizar una representación de forma animada de los métodos de ordenación Burbuja, Merge Sort y Radix y poder observarlo conllevo al aprendizaje y utilización de los lenguajes de programación JavaScript, HTML 5, CSS y librerías.

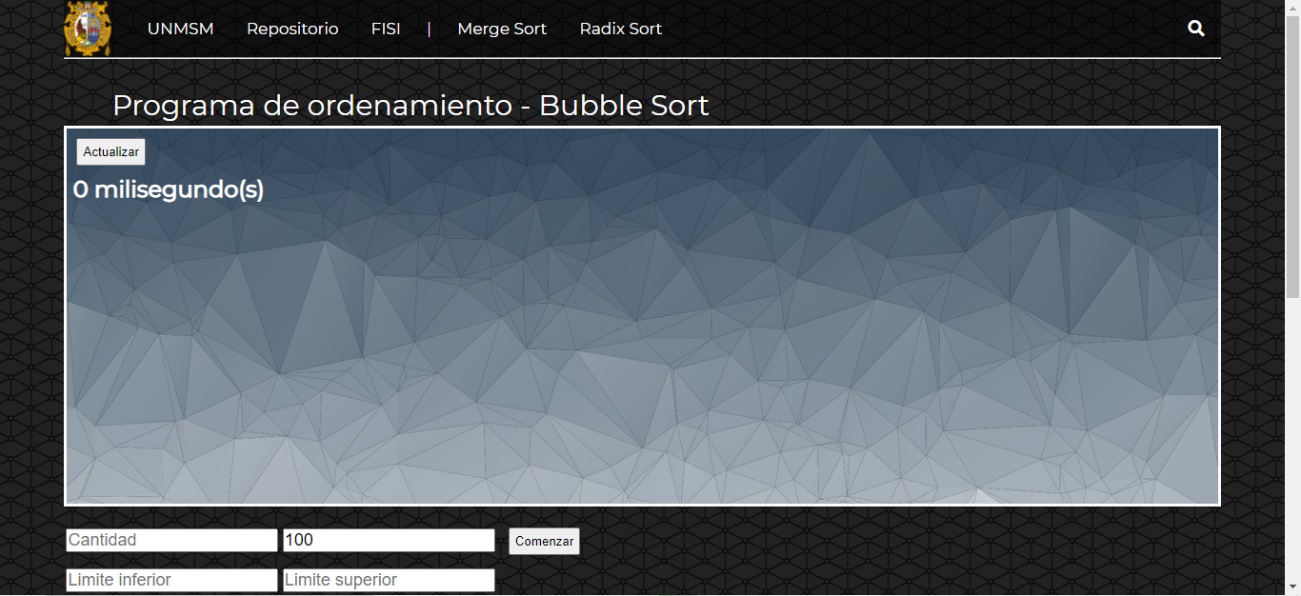
Las tareas para la realización de este proyecto fueron la comprensión y análisis de estos métodos de ordenación como de las herramientas a utilizar y la técnica divide y vencerás, división de tareas en el grupo por cada ordenamiento e implementación de la interfaz.

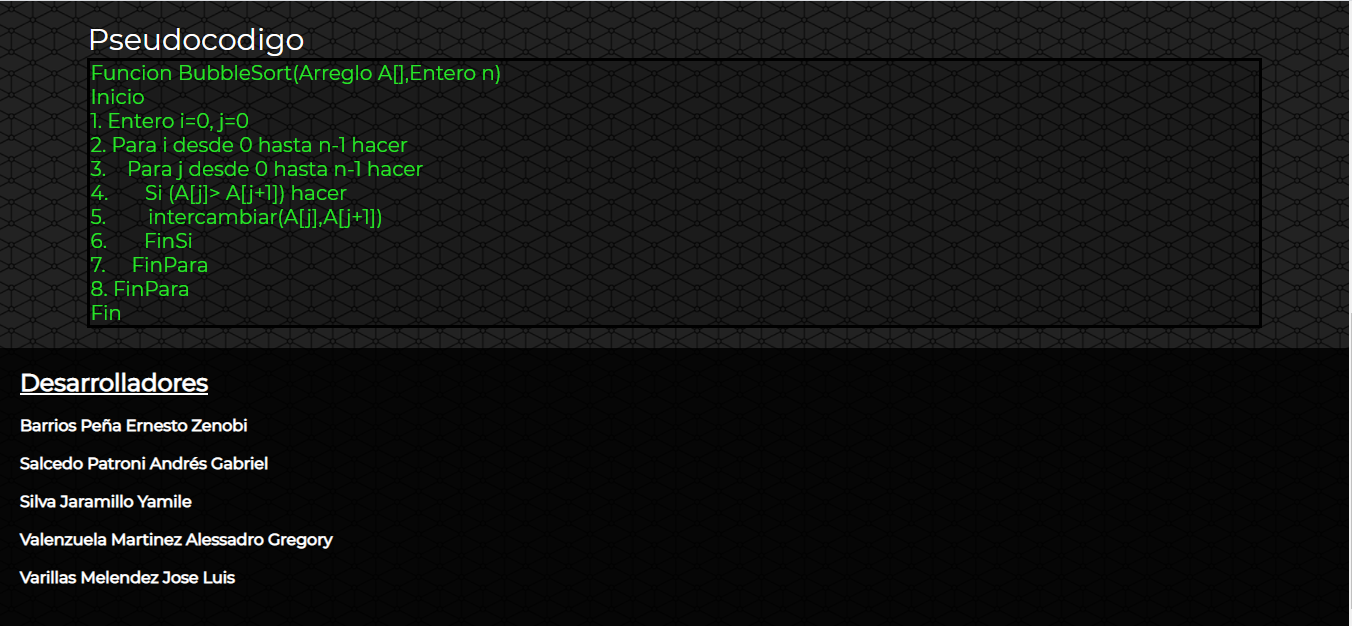
Este proyecto tiene facilidad de aprendizaje, conllevando que los usuarios desarrollen una interacción efectiva con la página web, con una interfaz amigable, sin muchos conocimientos previos sobre los métodos de ordenación o de las herramientas utilizadas. Donde los errores del usuario serian mínimos.

El contenido de esta pagina se da de manera rápida, simple con una navegación constante con una interfaz que resulta entendible y atractiva al usuario.

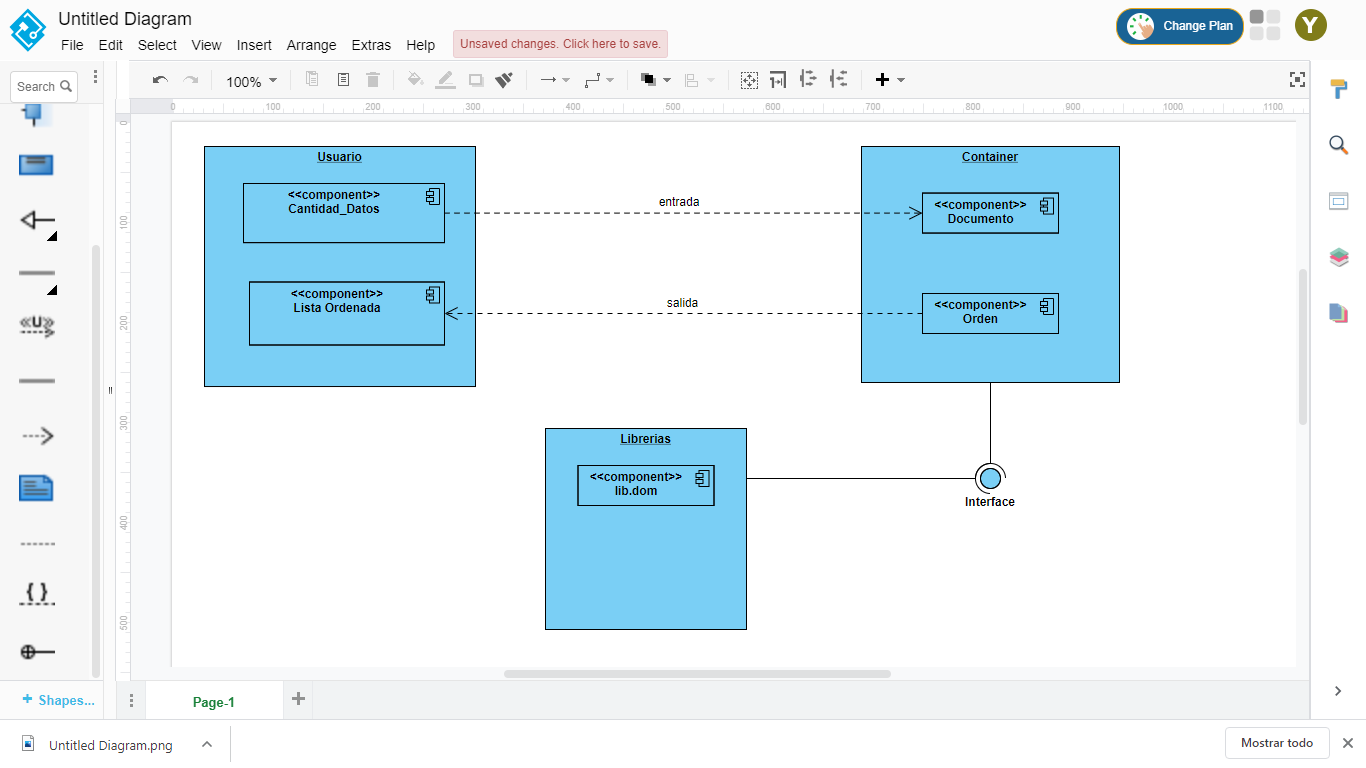
**Buble Sort**

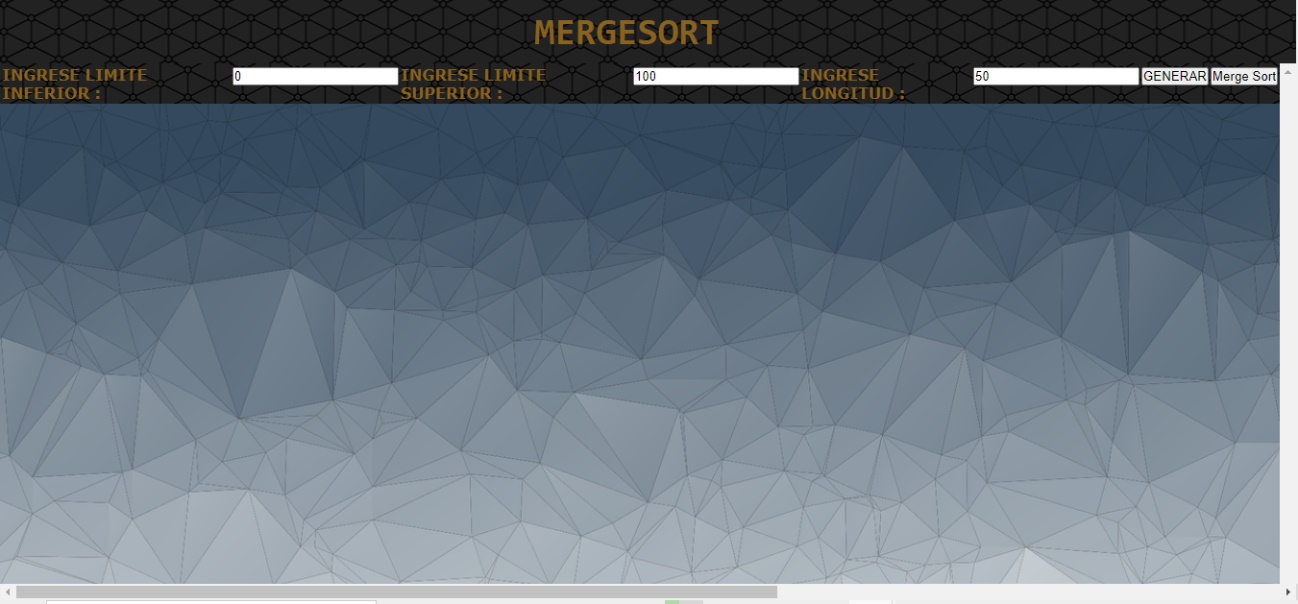
****

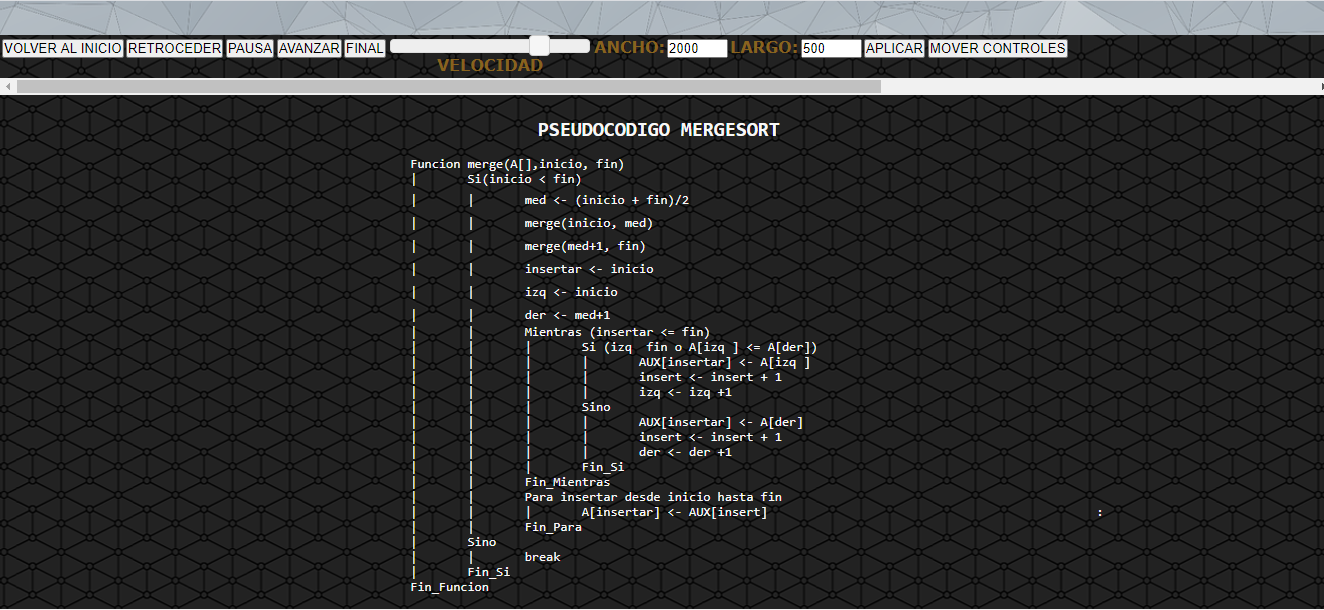




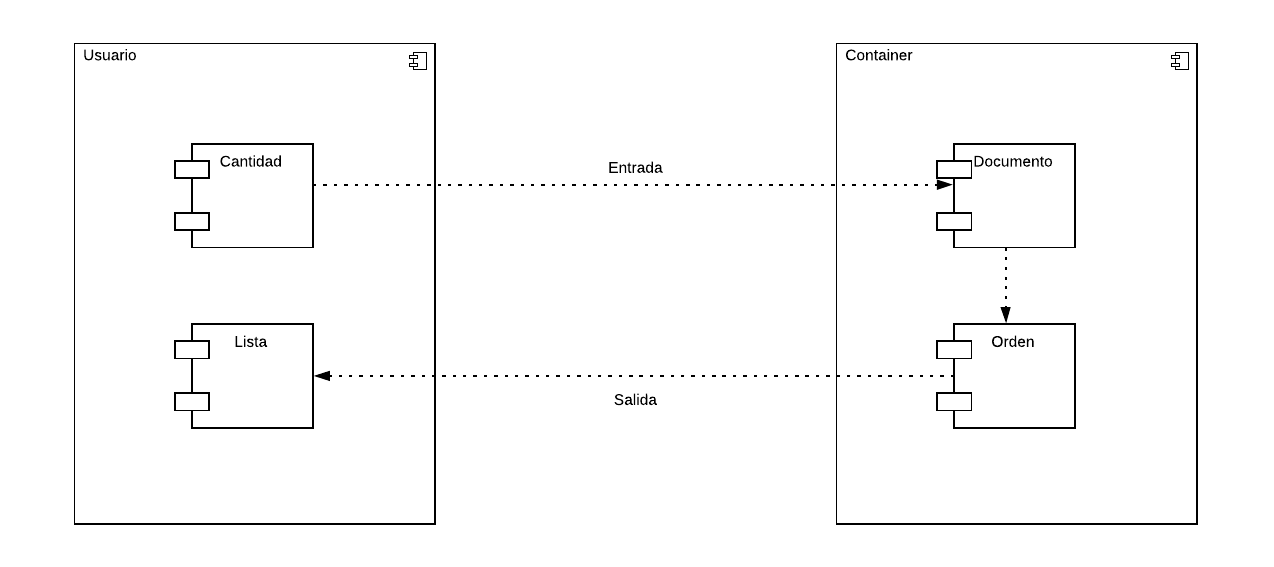
**Merge Sort**

****

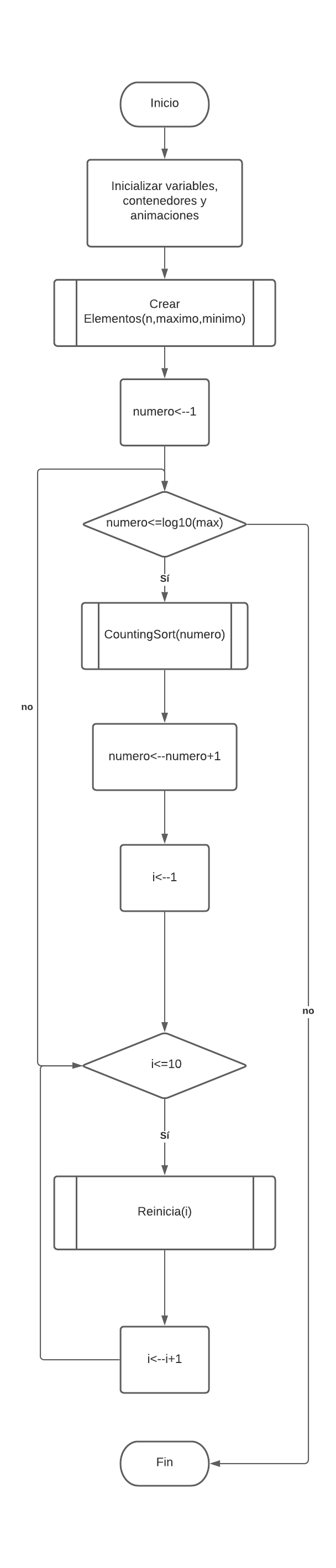




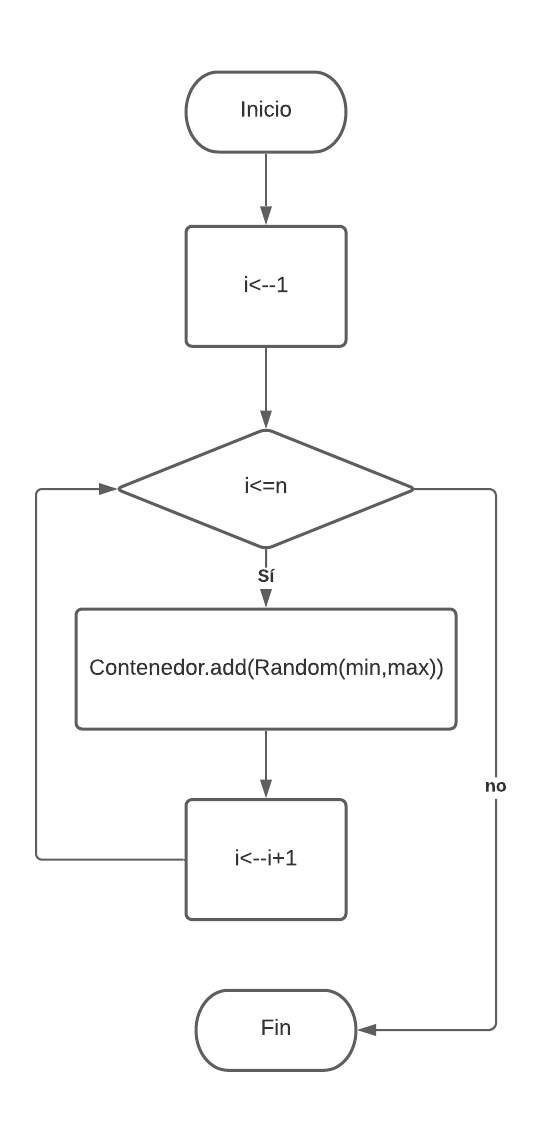
**Radix Sort**

****

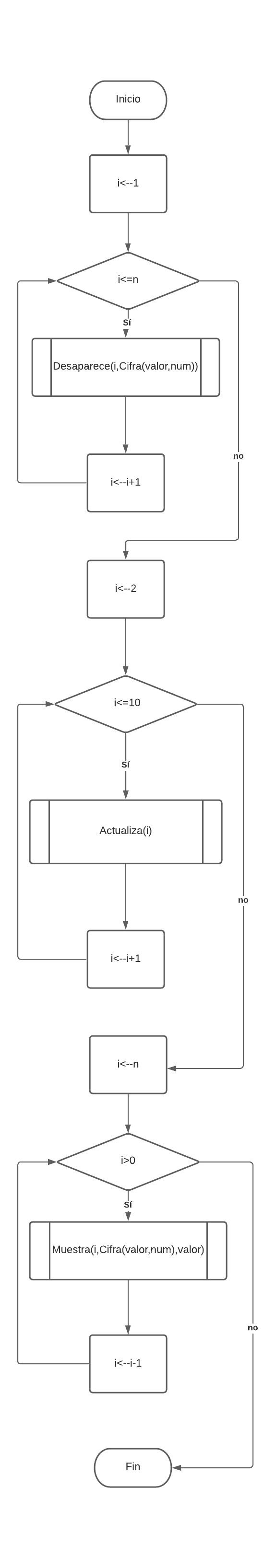
**Principal**

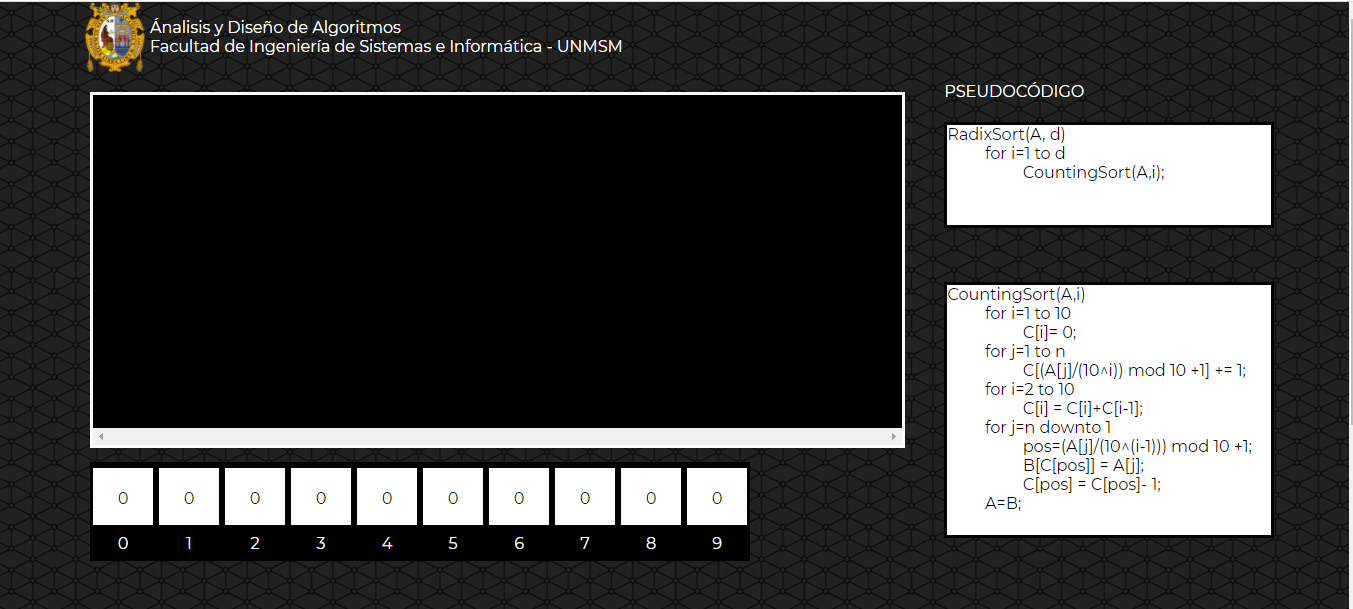


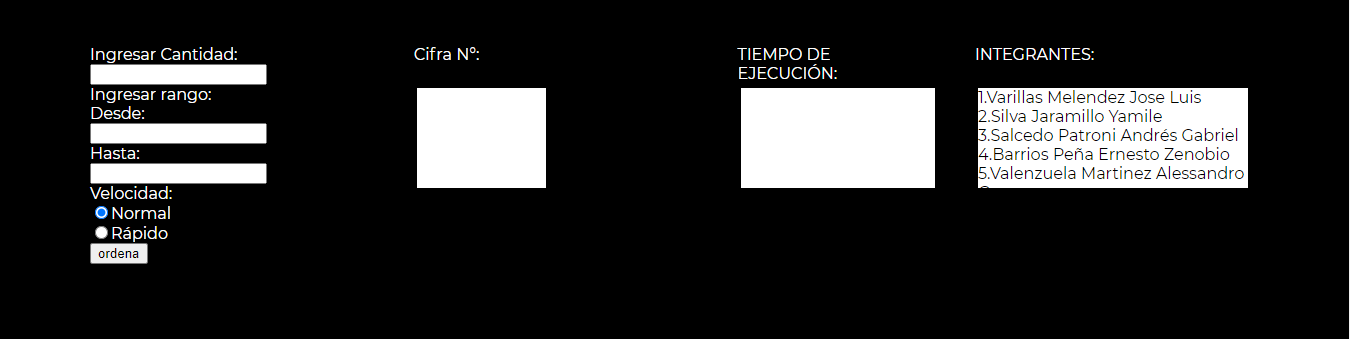
**CrearElementos**



**CountingSort**







**APORTES**

**Bubble Sort**

Para el caso de la implementación del algoritmo de ordenamiento por el método de la burbuja se utilizó como aportes el uso de librerías para dibujar los bloques con tamaño en proporción al número que representan, dando así una mejor vista en el funcionamiento de esta ordenamiento. Luego, se ideó una forma de avanzar con el proyecto dividiendo cada método de ordenamiento en una página cada uno para poder implementar todas las funciones y recursos de manera independiente a los demás programas.

**Merge Sort**

El método de ordenación Merge Sort es el algoritmo de ordenamiento por mezcla, que nos permite manejar gran cantidad de información de manera estable basado en la técnica divide y vencerás.

Para la implementación de este algoritmo se utilizó como aporte el uso de librerías para poder observar los bloques en cierta posición y tamaño, la etiqueta que acompaña a cada bloque que indica el tamaño del bloque y para las animaciones, de esta manera brindar una visualización interactiva y amigable del método para el usuario. Luego se complementó con la visualización de los demás métodos en diferentes páginas, pero con recursos independientes.

**Radix Sort**

La estructura, el diseño y la funcionalidad de la página web que muestra el algoritmo de ordenamiento Radix sort se realizó de manera independiente con respecto a las webs que muestran los otros dos algoritmos. Las animaciones fueron implementadas con CSS mientras que se usó el método setTimeout() para ubicarlas en una línea de tiempo y puedan ser ejecutadas en el momento correcto. También se crearon las funciones crearElementos(), desaparece(), muestra(), actualizar(), reiniciar() como apoyo para realizar las tareas necesarias en el proceso de animación.

**CONCLUSIONES Y RECOMENDACIONES**

El proyecto que realizamos ha contribuido de manera muy importante para identificar y resaltar los puntos que hay que cubrir y considerar para llevar acabo una implementación exitosa de los métodos de ordenación. Nos deja muchas cosas importantes que reflexionar y muchas otras las ha reforzado como puntos angulares para llevar a cabo una buena implementación.

Por otro lado, también vimos que tan necesario es tener un mínimo de conocimiento en general de los lenguajes usados en este proyecto por lo cual al inicio de este mismo se dio una semana para aprenderlos y tener un nivel del cual empezar

Por último luego de haber visto los tres métodos de ordenamientos asignados vimos que el bubble sort de ordenamiento n2 , merge sort con nlogn y el radix sort con nk son estables y cada uno de ellos tiene que usarse dependiendo de la necesidad del usuario.

**REFERENCIAS**

Colaboradores de MDN. (08 de agosto del 2020). *Tecnología para desarrolladores web*. MDN web docs. <https://developer.mozilla.org/es/docs/Web>

W3Schools. *HTML The language for building web pages*. <https://www.w3schools.com/>

jQuery, write les, do more. *jQuery API.* (2020). <https://jquery.com/>

Galles, David. (2011). *Data Structure Visualizations. myUSF.* <https://www.cs.usfca.edu/~galles/visualization/source.html>

**APLICACION**

Pagina web:  
<https://alessandrovm.github.io/Grupo4-ADA/>

Repositorio:  
https://github.com/alessandroVM/Grupo4-ADA