Manual Técnico

Christian Alessander Blanco González
202000173
Facultad de Ingeniería de la Universidad de San Carlos de Guatemala

INTRODUCCIÓN

El presente manual técnico tiene como finalidad describir el diseño de dicho proyecto. Esta página web fue desarrollada con uno de los lenguajes de programación llamado JavaScript (JS) y con el lenguaje de marcado (HTML). Realizado en el entorno de Visual Studio Code. Dicha página web cuenta con una interfaz gráfica que fue realizada a partir de CSS.

También cuenta con un diagrama de clases diseñado por el estudiante donde se podrá visualizar de una mejor forma en la que se desglosó la resolución de dicho programa, con el objetivo de facilitar dicho entendimiento del desarrollo. El motivo de dicha página web es poder leer archivos con la extensión json y la implementación de las estructuras de datos lineales vistas en clase.

OBJETIVOS

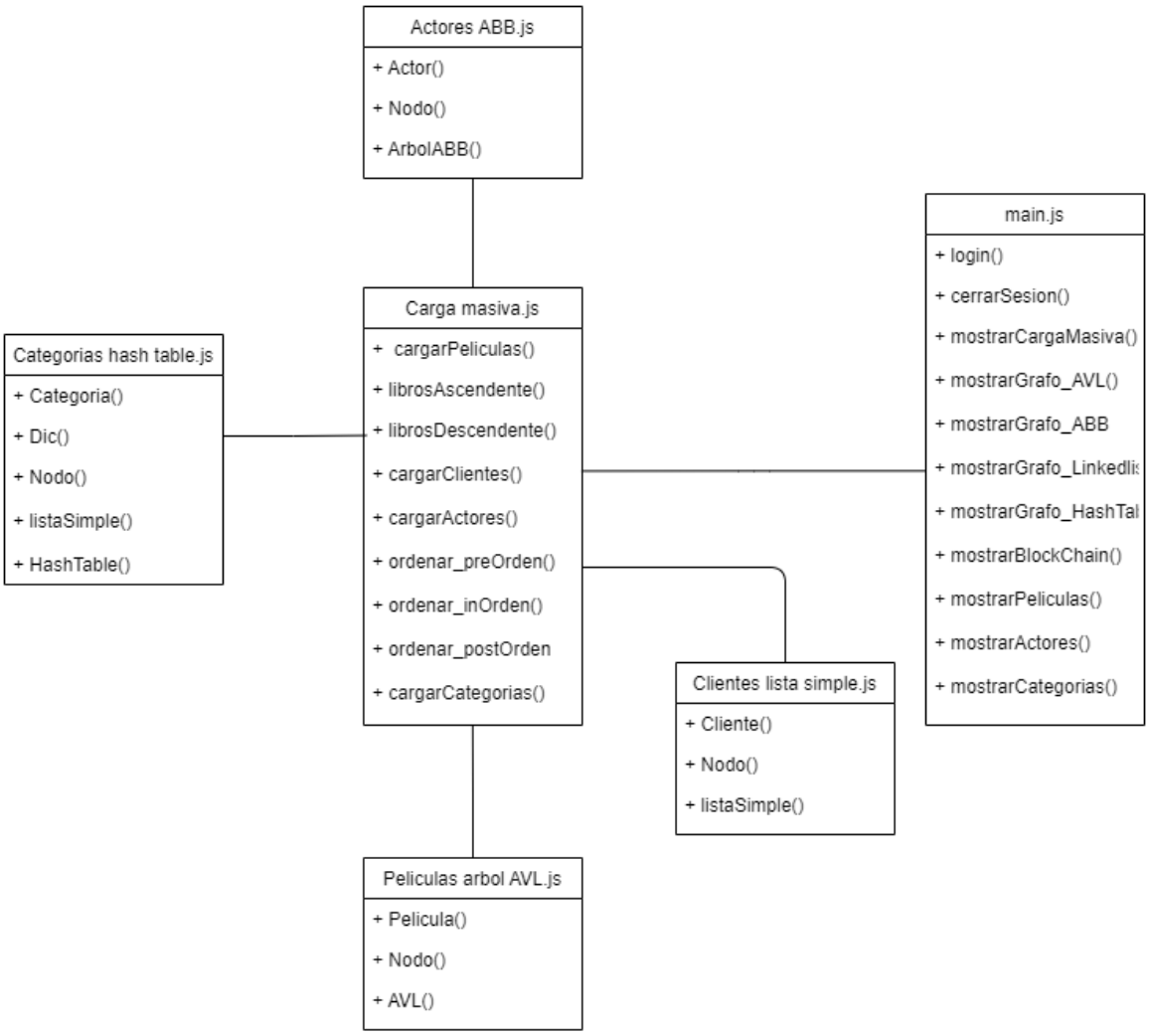
- Demostrar los conocimientos adquiridos sobre estructuras de datos lineales poniéndolos en práctica en una aplicación de simulación.
- Utilizar el lenguaje de JavaScript para implementar estructuras de datos lineales y visualizarlas en una página Web.
- Utilizar una herramienta que permita graficar las estructuras de datos lineales.
- Definir e implementar algoritmos de búsqueda, recorrido y eliminación

DESARROLLO DEL PROGRAMA

La aplicación web esta conformada por una interfaz grafica realizada con un archivo de extensión css para poder obtener una mayor amigabilidad con el usuario que utilizará este programa. Este programa consiste en la venta de libros, y así mismo ayudar a la población con obtener los libros digitales.

ESPECIFICACIONES TECNICAS

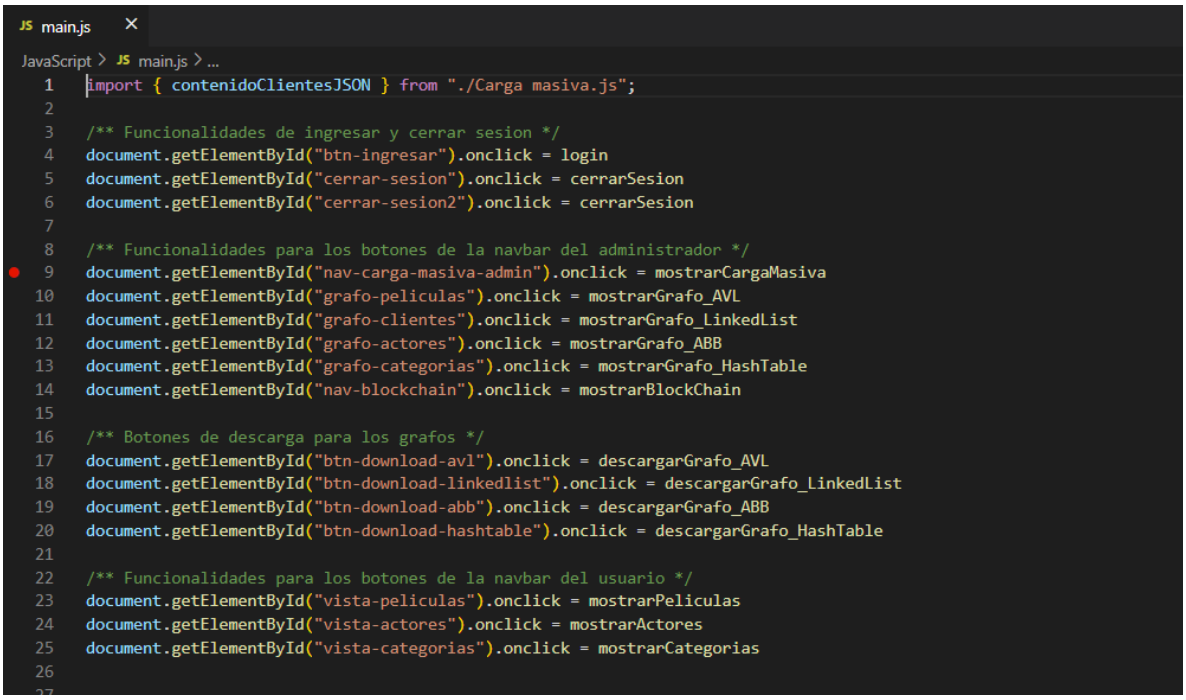
Lenguajes implementados	JavaScript, CSS, HTML
IDE usado	**Ninguno**
Editor de código	**Visual Studio Code**
Sistema Operativo	**Windows 10 (64 bits)**
Diseño web	**CSS**
Librerias implementadas	**Graphviz, d3**



En el diagrama de clases podemos visualizar cada uno de los archivos de JavaScript que fueron trabajados, en la cual podemos ver también cada una de las clases que conforman en cada archivo.

Main.js:

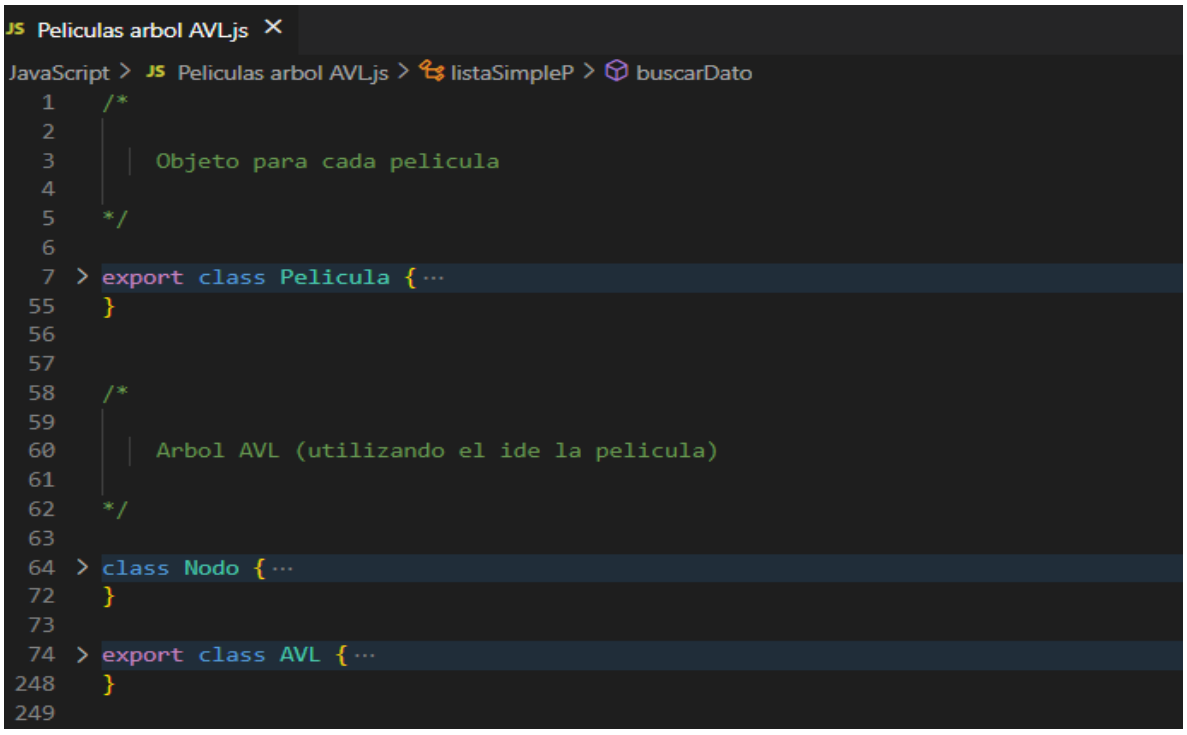
Es el encargado de poder mostrar cada una de las vistas creadas por medio de las funcionalidades de cada botón.



```
import { contenidoClientesJSON } from "./Carga masiva.js";

/** Funcionalidades de ingresar y cerrar sesion */
document.getElementById("btn-ingresar").onclick = login
document.getElementById("cerrar-sesion").onclick = cerrarSesion
document.getElementById("cerrar-sesion2").onclick = cerrarSesion

/** Funcionalidades para los botones de la navbar del administrador */
document.getElementById("nav-carga-masiva-admin").onclick = mostrarCargaMasiva
document.getElementById("grafo-peliculas").onclick = mostrarGrafo_AVL
document.getElementById("grafo-clientes").onclick = mostrarGrafo_LinkedList
document.getElementById("grafo-actores").onclick = mostrarGrafo_ABB
document.getElementById("grafo-categorias").onclick = mostrarGrafo_HashTable
document.getElementById("nav-blockchain").onclick = mostrarBlockChain

/** Botones de descarga para los grafos */
document.getElementById("btn-download-avl").onclick = descargarGrafo_AVL
document.getElementById("btn-download-linkedlist").onclick = descargarGrafo_LinkedList
document.getElementById("btn-download-abb").onclick = descargarGrafo_ABB
document.getElementById("btn-download-hashtable").onclick = descargarGrafo_HashTable

/** Funcionalidades para los botones de la navbar del usuario */
document.getElementById("vista-peliculas").onclick = mostrarPeliculas
document.getElementById("vista-actores").onclick = mostrarActores
document.getElementById("vista-categorias").onclick = mostrarCategorias
```

Películas árbol AVL.js:

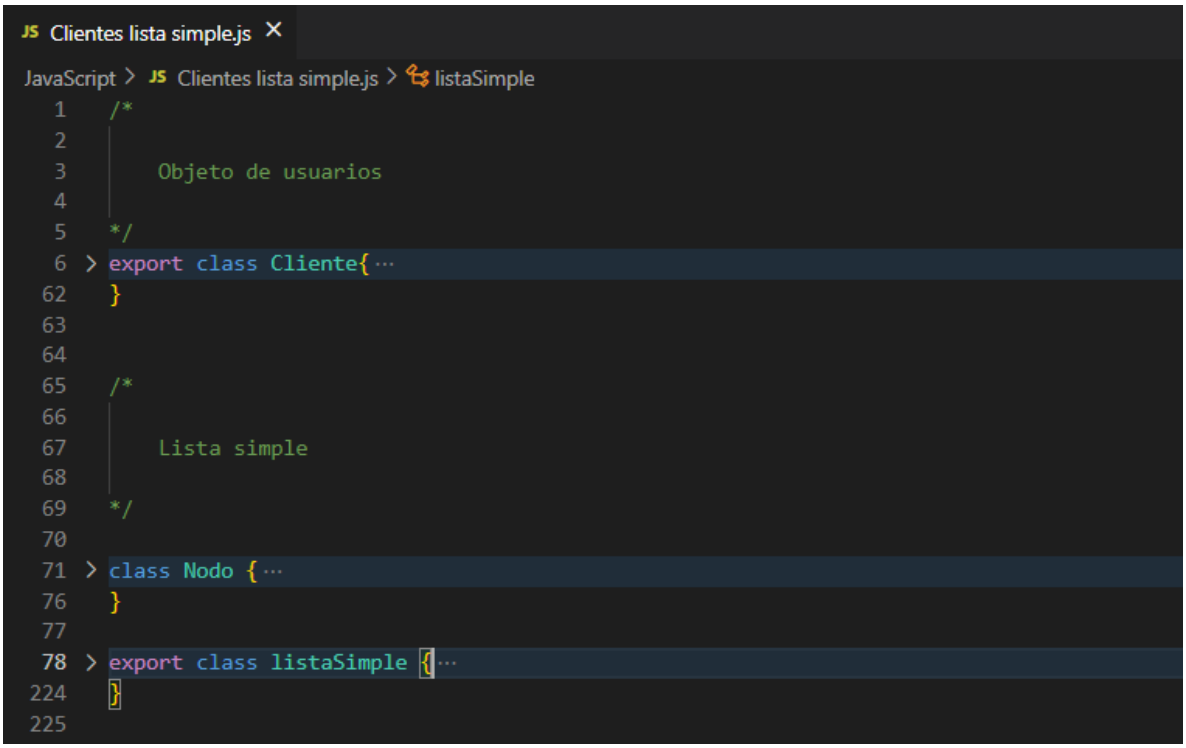
Es el encargado de poder almacenar y crear el objeto Película. Así tambien realiza el grafo de nuestro árbol AVL y la vista de cada una de las películas para el usuario por clases separadas.



```
/*

    Objeto para cada pelicula

*/

export class Pelicula {···
}


/*

    Arbol AVL (utilizando el ide la pelicula)

*/

class Nodo {···
}

export class AVL {···
}
```

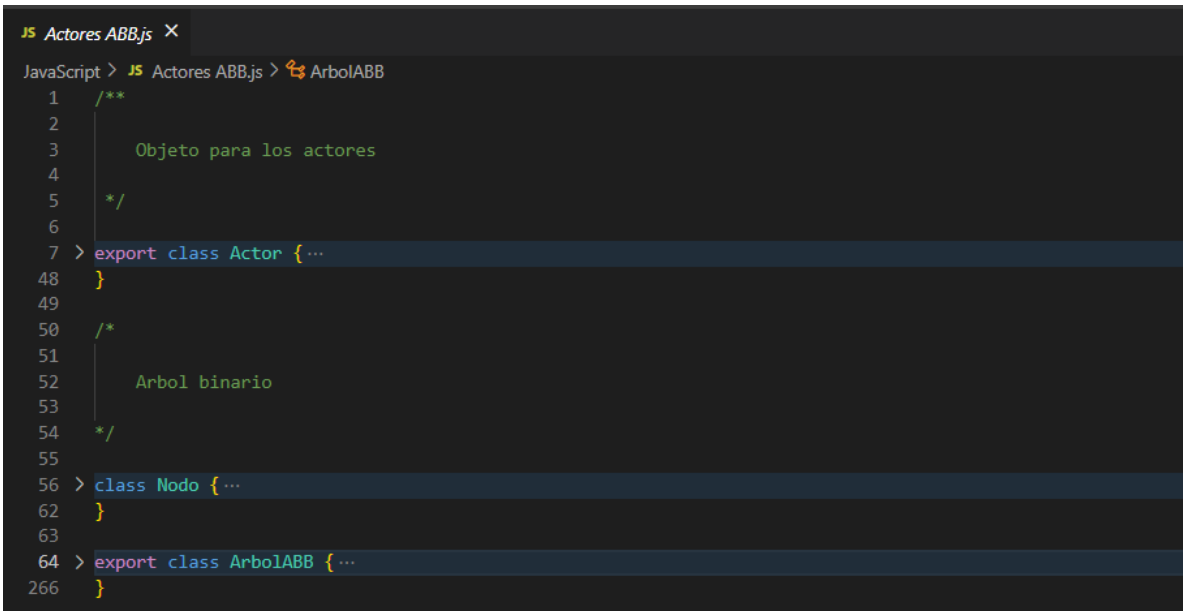
Clientes lista simple.js:

El encargado de crear y guardar cada uno de los objetos Cliente, esta conformada por 3 clases donde la clase Cliente es el objeto, la clase Nodo es la referencia del objeto guardado y la clase listaSimple que es la encargada de almacenar los objetos como una lista como tal, y la encargada de realizar el grafo de clientes.



Actores ABB.js:

Esta compuesta por 3 clases, una clase que es la creación del objeto Actor, otra clase llamada Nodo donde se hacen referencia a cada uno de los objetos almacenados en la estructura del árbol binario, y una tercera clase llamada ArbolABB que es la encargada de poder realizar cada uno de los ordenes (preOrder, inOrder, postOrder) para la vista de actores al usuario.

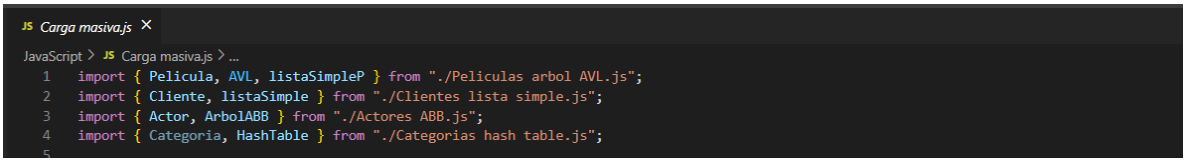


Categorías hash table.js:

Compuesta por 5 clases, la clase Categoria es la encargada de crear y almacenar el objeto para las categorías previamente ingresadas. La calse llamada Dic es la encargada de almacenar la llave y el valor previamente ingresados a nuestra tabla hash table. La clase Nodo es la encargada de hacer referencia al objeto y poder acceder a cada uno de sus atributos. La clase listaSimple es una lista simple como tal. Y la clase HashTable es la encargada de poder realizar el llenado de la tabla, ya que esta cuenta con un constructor donde se pide ingresar el valor máximo de nodos para la tabla, y luego hacer una búsqueda por medio del resultado de la llave % (capacidad de la tabla) y así poder ubicarlo en la posición correspondiente de la tabla hash.

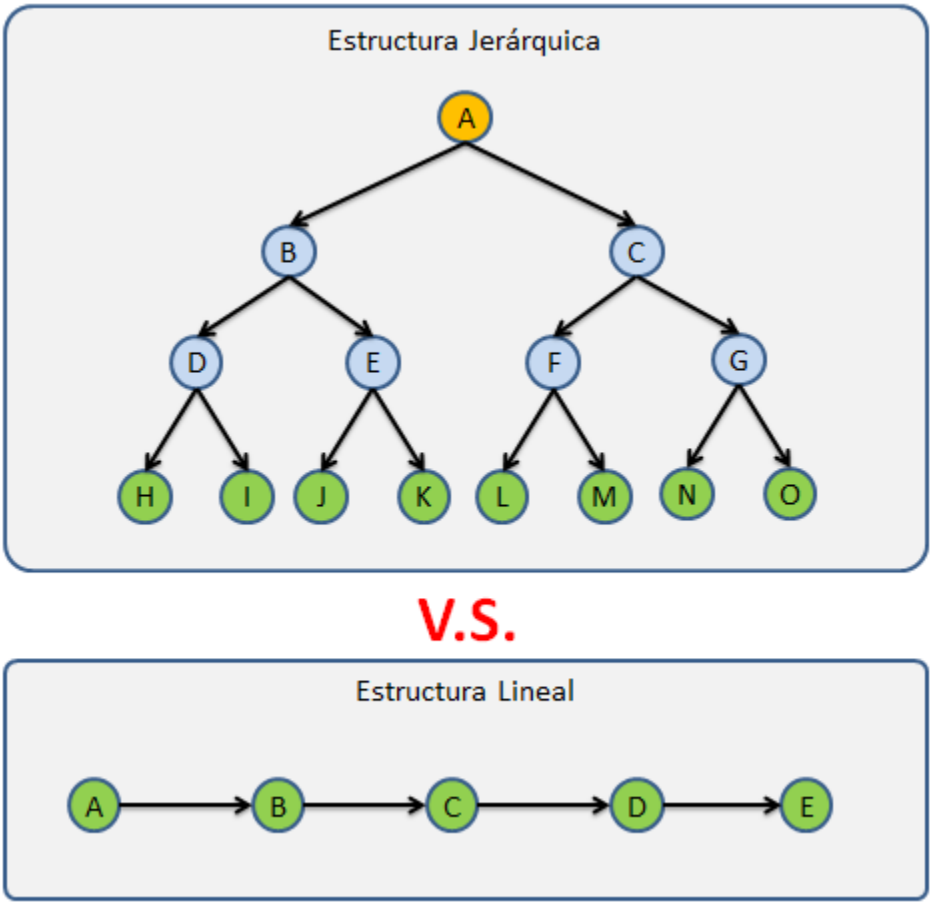
Carga masiva.js:

Es la encargada de poder cargar y leer los archivos json previamente seleccionados. Tambien es la encargada de enviar los datos del archivo json al objeto y a su estructura como tal, para así poder realizar los demás procedimientos solicitados, como las vistas al usuario y los grafos. Tambien es la encargada de realizar cada uno de los ordenamientos para las películas y actores.



ARBOLES

Los árboles son las estructuras de datos más utilizadas, pero también una de las más complejas. Los árboles se caracterizan por almacenar sus nodos en forma jerárquica y no en forma lineal como las Listas ligadas, Colas, pilas, etc, de las cuales ya hemos hablado en días pasados.



Datos importantes de los árboles:

Para comprender de mejor manera que es un árbol comenzaremos explicando cómo está estructurado.

- **Nodos:** Se le llama Nodo a cada elemento que contiene un árbol.
- **Nodo Raíz:** Se refiere al primer nodo de un árbol, solo un nodo del árbol puede ser la raíz.
- **Nodo Padre:** Se utiliza este término para llamar a todos aquellos nodos que tienen al menos un hijo.
- **Nodos hijo:** Los hijos son todos aquellos nodos que tiene un padre.
- **Nodo Hermano:** Los nodos hermanos son aquellos nodos que comparte a un mismo padre en común dentro de la estructura.
- **Nodo Hoja:** Son todos aquellos nodos que no tienen hijos, los cuales siempre se encuentran en los extremos de la estructura.
- **Nodo Rama:** Estos son todos aquellos nodos que no son la raíz y que además tiene al menos un hijo.

ESTRUCTURA DE DATOS

Una estructura de datos es un tipo de dato compuesto que permite almacenar un conjunto de datos de diferente tipo. Los datos que contiene una estructura pueden ser de tipo simple (caracteres, números enteros o de coma flotante, etc.)

A cada uno de los datos o elementos almacenados dentro de una estructura se les denomina miembros de esa estructura y estos pertenecerán a un tipo de dato determinado.

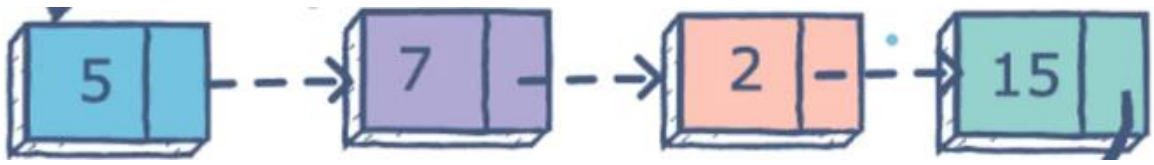
- **Estructuras de datos estáticas:** Son aquellas en las que se asigna una cantidad fija de memoria y no cambia durante la ejecución de un programa, es decir, las variables no pueden crearse ni destruirse durante la ejecución del programa.
- **Estructuras de datos dinámicas:** Son aquellas en las que su ocupación en memoria puede aumentar o disminuir durante el tiempo de ejecución de un programa.

  A su vez las estructuras de datos dinámicas se pueden clasificar en lineales y no lineales:

  - Estructuras lineales
  - Estructuras no lineales

Estructuras lineales:

Son aquellas en las que se definen secuencias como conjuntos de elementos entre los que se establece una relación de predecesor y sucesor. Las diferentes estructuras de datos basadas en este concepto se diferencian por las operaciones de acceso a los elementos y manipulación de las estructuras. Existen tres estructuras lineales especialmente importantes: las pilas, las colas y las listas.



CONCLUSIONES

- Como hemos observado los árboles son estructuras bastante complejas, tiene una gran aplicación en la ciencia y en la programación convencional. En los últimos años este tipo de estructuras ha sido utilizadas con mucha frecuencia en la Inteligencia artificial.

- Mediante el desarrollo de este proyecto se ha logrado desarrollar y entender la utilización de cada uno de las funciones de cada estructura previamente solicitada, para así poder concluir que la estructuración de datos es muy importante para la manipulación de información de objetos como tal.