Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 2do. Semestre 2023.

# PROYECTO 2 – IMPLEMENTACION DE TDA

**201900532 – Juan José Gerardi Hernández**

**Resumen**

Un TDA es un conjunto de datos u objetos al cual se le asocian operaciones. Los TDA's se utilizan para abstraer y definir como se organizan y manipulas los datos en un programa de computadora.

Las listas enlazadas es un TDA que nos permite almacenar información de una manera organizada, estas estructuras se caracterizan por ser dinámicas lo que favorece al momento de guardar los datos ya que no poseen un limite o un espacio fijo en memoria. Son bastante importante ya que se les puede dar una gran variedad de implementaciones.

Como por ejemplo en el uso navegadores web en el área del historial.

La programación orientada a objetos (POO) es un paradigma de programación el cual a ganado una gran popularidad debido a su capacidad de crear aplicaciones mas robustas o flexibles y fácil de mantener. Esta metodología se basa en la idea de que los programas se pueden organizar como una colección de objetos.

**Palabras clave**

- ***POO***
- ***XML***
- ***TDA***
- ***LISTA ENLAZADA***
- ***MEMORIA DINAMICA***

***Abstract***

*A TDA is a set of data or objects to which operations are associated. TDAs are used to abstract and define how data is organized and manipulated in a computer program.*

*Linked lists are a TDA that allows us to store information in an organized way; these structures are characterized by being dynamic, which favors the storage of data since they do not have a limit or a fixed memory space. They are quite important since they can be given a wide variety of implementations. Such as in the use of web browsers in the area of history.*

*Object-oriented programming (OOP) is a programming paradigm that has gained great popularity due to its ability to create more robust, flexible, and easily maintainable applications. This methodology is based on the idea that programs can be organized as a collection of objects.*

***Keywords***

- *OPP*
- *XML*
- *ADT*
- *Linked list*
- *Dynamic memory*

## Introducción

En la creación de distintos programas de informática es esencial comprender el cómo implementar distintas herramientas como lo son las estructuras de datos como las listas enlazadas ya que pueden ser muy eficientes al momento de usarlas en el procesamiento y gestión de los datos. Un ejemplo podría ser el uso que se les dio en el proyecto en las cuales se usaron para el manejo de la información obtenida de los documentos XML. Al igual la implementación de la programación orientada a objetos para poder realizar la aplicación.

## Desarrollo del tema

En el desarrollo de software muchas veces es difícil el saber que herramientas implementar para poder llevar a cabo las distintas tareas deseadas, por ello es importante el conocer distintos temas sobre ello y tener en cuenta que cada proyecto busca satisfacer diferentes necesidades para el usuario por lo que debe tener un ecosistema que sea agradable y fácil de usar para el usuario y así mismo que pueda realizar las tareas para las que fue desarrollado.
En el desarrollo del proyecto se tenia que tener en cuenta que se debían implementar lo que eran listas enlazadas simples las cuales son una colección lineal de elementos llamados nodos. El orden entre ellos se establece mediante punteros, en el caso de las listas simplemente enlazadas solamente se hace uso del puntero 'siguiente' el cual apunta al dato siguiente.
Para que una estructura sea un TDA debe tener operadores asociados que permitan la manipulación de los datos que contiene. Los operadores básicos de una lista son:

- Insertar
- Buscar
- Eliminar
- Buscar
- Localizar
- Vaciar

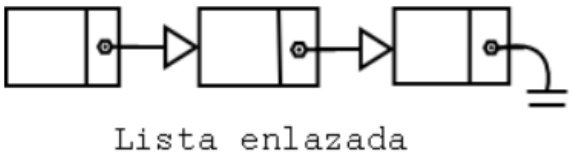


*Ilustración 1: representación gráfica de una lista enlazada*

Las listas se implementaron para el manejo de los datos los cuales se obtuvieron de documentos XML los cuales contenían información de las señales como su amplitud, tiempo y así como el nombre del conjunto de datos. Los archivos XML o lenguaje de marcado extensible el cual es un lenguaje de marcado que define un conjunto de reglas para la codificación de documentos que se puede aplicar en el análisis de datos.

```xml
<?xml version="1.0"?>
<senales>
  <senal nombre="Señal Facilita" t="10" A="3">
    <dato t="1" A="1">3</dato>
    <dato t="1" A="2">5</dato>
    <dato t="1" A="3">4</dato>
    <dato t="2" A="1">1</dato>
    <dato t="2" A="2">2</dato>
    <dato t="2" A="3">3</dato>
    <dato t="3" A="1">0</dato>
    <dato t="3" A="2">0</dato>
    <dato t="3" A="3">0</dato>
    <dato t="4" A="1">0</dato>
    <dato t="4" A="2">0</dato>
    <dato t="4" A="3">0</dato>
    <dato t="5" A="1">0</dato>
    <dato t="5" A="2">0</dato>
    <dato t="5" A="3">0</dato>
    <dato t="6" A="1">5</dato>
    <dato t="6" A="2">0</dato>
    <dato t="6" A="3">7</dato>
    <dato t="7" A="1">2</dato>
    <dato t="7" A="2">0</dato>
    <dato t="7" A="3">1</dato>
    <dato t="8" A="1">5</dato>
    <dato t="8" A="2">5</dato>
    <dato t="8" A="3">0</dato>
    <dato t="9" A="1">4</dato>
    <dato t="9" A="2">4</dato>
    <dato t="9" A="3">0</dato>
    <dato t="10" A="1">5</dato>
    <dato t="10" A="2">9</dato>
    <dato t="10" A="3">1</dato>
  </senal>
</senales>
```

*Ilustración2: Ejemplo de entrada XML*

Al observar el desarrollo de la aplicación y su estructura se puede especificar su entorno y sus funciones. Al arrancar el programa lo primero que se observa es el menú principal que posee 7 opciones.

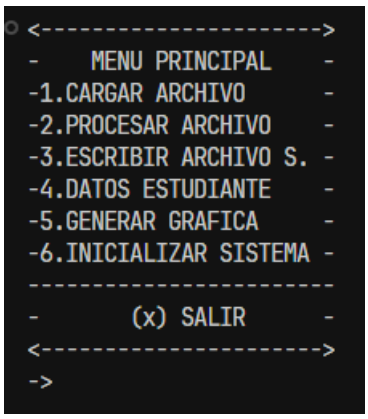


*Ilustración 3: Diseño de menu inicial*

Enfocándonos en la aplicación podemos ver el funcionamiento de cada opción la cual en cada una se hace uso en el manejo de distintas herramientas como

a) Graphviz: es un software de visualización gráfico, el cual es código abierto. La visualización grafica es una forma de representar la información estructural como diagramas de gráficos y redes abstractas.
b) Python: es un lenguaje de programación de alto nivel lo cual significa que es fácil de aprender y eficiente. Es ampliamente utilizado en el desarrollo web, ciencia de datos, machine learning y desarrollo de software.

En las librerías utilizadas en el proyecto se hicieron uso de tres:

- Os: permitir borrar consola cuando se ejecuta en el Código
- Tkinter: uso de la parte grafica para buscar los archivos deseados
- Xml.etree: leer documentos XML

Descripcion de cada opción del menú:

En base al diseño ejemplo del menú el cual se proporciono en el enunciado del proyecto, se especificaron la implementación de 7 opciones que poseería la aplicación:

a) **Cargar archivo:**
Aquí se le da la opción al cliente de poder ingresar al sistema el documento a analizar el cual como se a mencionado anteriormente es en formato XML

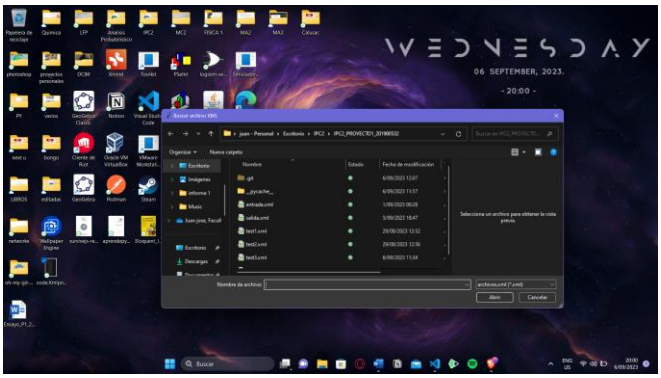


*Ilustración 4: despliegue de la ventana explorador de archivos*

En esta opción se lee el archivo con la biblioteca elementTree la cual por medio de ciclos for permite de una forma muy sencilla lo que es obtener los datos contenidos en el documento y luego se crea los objetos y se guardan en las listas enlazadas las cuales ayudan a manejar de forma muy eficiente los datos. Ya que son estructuras dinámicas. Esta parte del proyecto la mayor parte del planteamiento del desarrollo era el como se podría obtener los datos con ayuda de la librería disponible ya que a simple vista se mira cierto de grado de complejidad y una de las ventajas de esta librería es el fácil manejo de los datos ya que se manejan mediante ciclos ‘for’.
Obtenidos los datos mediante las funciones de inserción y ordenamiento de datos se agrego la información a la lista principal la cual es donde se van guardando en orden según su tiempo (t) y amplitud (A) cada señal que se obtiene así mismo se obtiene el patrón de “0” y “1” de cada señal los cuales ayudan a un mejor análisis de cada dato.

```
def lector_xml(Docs_xml):
    try:
        tree = ET.parse(Docs_xml)
        root = tree.getroot()
        for senales in root.findall('senal'):
            name = senales.get('nombre')
            t = senales.get('t')
            a = senales.get('A')
            for dato in senales.findall('dato'):
                tim = dato.get('t')
                amp = dato.get('A')
                dat = dato.text
                if int(dat)>0:
                    st = '1'
                elif str(dat) == '0':
                    st = '0'
                new_dato = Dato(tim,amp,st,dat)
                list_datos.insert_dato_ordenado(new_dato)
            new_senal = Senal(name,t,a,list_datos)
            list_sen.new(new_senal)
    except:
        print('Archivo no encontrado!')
```

*ıstración 5: función lector XML*

b) **Procesar archivo:**
Aquí ya obtenidos los datos se procesan es decir que se crean las matrices con los datos las cuales se visualizan gráficamente con Graphviz. Se muestran 3 tipos de matrices:

| Matriz | definición |
|---|---|
| Frecuencias | Se muestran los datos según su tiempo y amplitud es decir una matriz t*A |
| Patrones | Se muestra en la posición del dato un 0 o 1 según el dato, donde:<br>D = 0 = 0<br>D > 0 = 1 |
| Reducida | Según el patrón obtenido se comparan las cadenas y se suman según su fila y columna solo y si sus patrones son idénticos. |

*Tabla 1*

Aquí para la obtención de las matrices se manejo las listas con métodos de búsqueda, eliminación e iguales el cual analizaba la cadena de patrones y buscaba cuales eran iguales y los clasificaba por grupo. La mayor dificultad en la creación de estas matrices fue la creación de la matriz reducida donde se tenia que analizar cada celda y por medio de funciones se iba sumando cada digito siempre y cuando el grupo que representaban tuviera un patrón idéntico a otro.

| t\|A | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| G: 1 3 | 5 | 7 | 0 | 6 |
| G: 2 5 | 0 | 0 | 9 | 4 |
| G: 4 | 1 | 0 | 1 | 5 |

*Ilustración 6: Ejemplo de matriz reducida*

c) **Escribir archivo de salida:**
Con ayuda de la librería elementTree se realizó la escritura del documento de salida con formato XML. En donde se registraron los datos obtenidos con la matriz reducida. Al principio se podría pensar que es algo muy laborioso lo que es crear el documento lo cual al contrario es muy sencillo ya que se hacen uso de algunos comandos para realizarlos como:

| Import | As |
|---|---|
| Xml.etree.ElementTree | ET |

*Tabla 2*

| Comando | Ejemplo |
|---|---|
| ET.Element() | ET.Element(‘senalReducida’) |
| ET.SubElement() | ET.SubElement(root,’senal’) |

*Tabla 3*

d) **Datos del estudiante:**
En esta sección se muestran en pantalla los datos del estudiante como su nombre, carnet, curso y sección en la que se encuentra asignado, así como carrera y semestre.

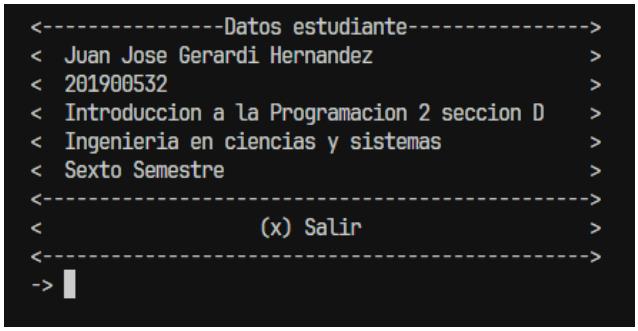


*Ilustración 7 tabla de datos del estudiante*

e) **Generar graficas:**
En esta sección con ayuda de Graphviz se crean las graficas de los datos obtenidos del archivo ingresado anteriormente donde se muestran la grafica de la matriz de frecuencias y grafica de señales reducidas.
El uso de Graphviz lleva cierta complejidad ya que su sintaxis de entrada es muy específica para definir los grafos. ya que requiere describir nodos, aristas, atributos y varios elementos del grafo de una manera precisa y estructurada. Un punto donde se puede apoyar bastante para su comprensión y que su implementación sea mas agradable es que posee una documentación bastante detallada desde cómo crear grafos simples a más complejos.

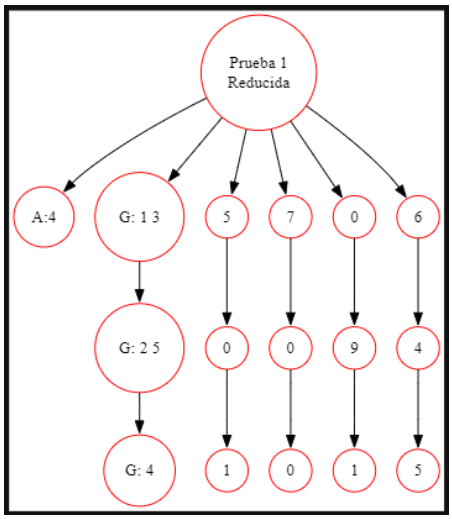


*Ilustración 8 grafica de frecuencias reducidas*

f) **Inicializar sistema:**
Ofrece al usuario la opción de borrar datos ingresados anteriormente y así poder registrar nuevos documentos

g) **Salir:**
Cierra el programa

**Importancia de las listas enlazadas**

Las listas enlazadas son estructuras de datos fundamentales en la programación debido a su versatilidad y utilidad en una amplia gama de aplicaciones. Algunas de las principales razones por que las listas enlazadas son importantes:

- Estructura Dinámica
- Inserción y eliminación eficientes
- Uso eficiente de la memoria
- Facilitan la gestión de memoria
- Aprendizaje y educación

**Conclusiones**

→ Hoy en día las opciones de herramientas para implementar en proyectos son variadas y presentan distintos niveles de complejidad en su uso.

→ Las listas enlazadas son estructuras que presentan grandes ventajas como puede ser que son estructuras de datos que no presentan un espacio fijo en memoria.

→ Cada problema que se plantee en programación se puede solucionar de diversas maneras por lo que no hay una sola solución.

→ Los archivos XML se usan para guardar distintos tipos de datos. ¿Pero hasta donde podría llegar su importancia de implementación en sistemas actuales?

## Referencias bibliográficas

a) Canelo, M. M. (2023). ¿Qué es la programación orientada a objetos? *Profile Software Services*. https://profile.es/blog/que-es-la-programacion-orientada-a-objetos/#Beneficios_de_Programacion_Orientada_a_Objetos

b) *Graphviz*. (s. f.). Graphviz. https://graphviz.org/

c) De Souza, I. (2021). XML: ¿Qué es y para qué sirve este lenguaje de marcado? *Rock Content - ES*. https://rockcontent.com/es/blog/que-es-xml/

d) *Estructuras de datos: listas enlazadas, pilas y colas.* (s. f.). https://calcifer.org/documentos/librognome/glib-lists-queues.html

## Anexos

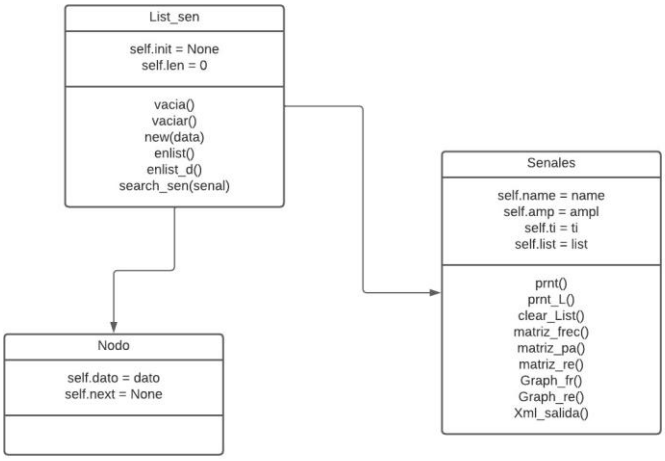


*Ilustración 2: Diagrama de clases*