Pestaña 1

Estructura de Datos y Recursividad

## Tp: Segunda entrega

## Fecha: 16/10

## Integrantes: Ariel Aguilar, Ignacio Nicolás Heredia, Katherine Avendaño

# Introducción:

El presente trabajo tiene como objetivo implementar y analizar la gestión de carpetas y subcarpetas utilizando estructuras de datos recursivas, particularmente un **Árbol General**. A Través de este modelo, se busca representar de manera jerárquica un sistema de almacenamiento de mensajes, donde cada carpeta puede contener tanto subcarpetas como mensajes individuales.

La recursividad permite recorrer, buscar y modificar elementos de esta estructura de forma eficiente y natural, replicando el funcionamiento de sistemas reales de organización como el de los gestores de correo electrónico o los exploradores de archivos. Además, se pretende analizar la eficiencia de las operaciones implementadas y complementar el desarrollo con material gráfico explicativo sobre el árbol de carpetas.

# 1) Descripción del Problema:

# En los sistemas modernos de información, es común la necesidad de organizar grandes volúmenes de datos o mensajes en estructuras jerárquicas que permitan acceder a ellos de forma ordenada y flexible. Un ejemplo típico son los servicios de correo electrónico, donde los usuarios crean carpetas y subcarpetas para clasificar sus mensajes según criterios personales o temáticos.

# El desafío radica en poder navegar y manipular esta jerarquía de manera eficiente. Para lograrlo, se requiere una estructura de datos que permita representar relaciones de padre-hijo entre elementos y que facilite operaciones como búsqueda, inserción, eliminación o movimiento de elementos entre carpetas.

# La estructura de árbol general resulta ideal para esta finalidad, ya que cada nodo puede tener múltiples hijos, lo que permite reflejar fielmente la naturaleza jerárquica y ramificada del sistema de carpetas y subcarpetas..

# 2) Estructura de Datos Utilizada:

La estructura seleccionada para esta implementación es el árbol general, en el cual cada nodo puede representar una carpeta o un mensaje.

## Nodo raíz: representa la carpeta principal o base del sistema, desde la cual se derivan todas las demás.

## Nodos intermedios: representan las subcarpetas, que pueden contener a su vez otras subcarpetas o mensajes.

## Nodos hoja: representan los mensajes, que son los elementos finales del árbol

## Cada modo puede contener :

## Un nombre o identificador (por ejemplo, “Bandeja de entrada”, “Trabajo”, “Mensaje 1”).

## Una lista de hijos (otras carpetas o mensajes).

## Un puntero al nodo padre (opcional, para facilitar movimientos o retornos).

El uso de listas o arreglos dinámicos permite almacenar una cantidad variable de hijos por carpeta. De esta forma, la estructura mantiene su flexibilidad y puede crecer o reducirse según sea necesario.

3) Implementación Paso a Paso:

3-1 ) Creación del árbol de carpetas:

El proceso comienza con la creación de la carpeta raíz, a partir de la cual se generan las demás carpetas y subcarpetas. Cada vez que se crea una nueva subcarpeta, se la añade como hijo del nodo correspondiente.

Este procedimiento puede realizarse de forma recursiva: cada función que agrega una carpeta puede volver a llamarse a sí misma si se desea insertar dentro de una subcarpeta existente

Por ejemplo:

Si el usuario desea crear la subcarpeta “2025” dentro de “Proyectos”, la función buscará recursivamente el nodo “Proyectos” y luego insertará “2025” como su hijo.

3-2) Inserción de mensajes:

Los mensajes se agregan dentro de las carpetas como nodos hoja. Para ello, la función de inserción recorre el árbol de forma recursiva hasta localizar la carpeta deseada. Una vez encontrada, se crea un nuevo nodo que representa el mensaje, con atributos como asunto, remitente y contenido.

Este mecanismo recursivo permite que la inserción funcione sin importar el nivel de profundidad del árbol, ya que la función se llama a sí misma hasta alcanzar el punto exacto donde debe añadirse el nuevo elemento.

3-3) Búsqueda recursiva de mensajes:

La búsqueda recursiva es una de las operaciones más importantes del sistema. Dado un criterio (por ejemplo, el asunto o el remitente), la función recorre todo el árbol verificando en cada nodo si el mensaje coincide con el valor buscado.

Si el nodo actual no cumple la condición, la función se vuelve a llamar para cada hijo, hasta encontrar el resultado o recorrer todo el árbol.

Este método garantiza que se exploren todas las posibles ubicaciones del mensaje, sin importar la profundidad de la estructura

3-4) Movimiento de mensajes entre carpetas:

Para mover un mensaje de una carpeta a otra, se requiere primero localizarlo dentro del árbol, lo cual se realiza mediante una búsqueda recursiva. Una vez encontrado, se elimina del conjunto de hijos de su carpeta original y se inserta como hijo de la carpeta destino.

Ambas operaciones (buscar y agregar) son recursivas, lo que demuestra cómo la recursividad facilita manipular estructuras jerárquicas sin necesidad de bucles complejos.

3-5) Eliminación o renombrado de carpetas (opcional):

Estas operaciones también pueden aplicarse de forma recursiva.

Por ejemplo, eliminar una carpeta implica eliminar todos los elementos que contiene, lo que se logra fácilmente recorriendo los nodos hijos y aplicando la misma función a cada uno.

Del mismo modo, cambiar el nombre de una carpeta o mensaje requiere primero localizar el nodo correspondiente de forma recursiva y luego modificar su atributo de nombre

## 4) Análisis de Eficiencia:

El análisis de eficiencia depende de las operaciones ejecutadas:

* Búsqueda recursiva: en el peor caso, se visitan todos los nodos del árbol, por lo que su complejidad temporal es O(n), siendo n el número total de nodos (carpetas y mensajes).
* Inserción y eliminación: si se conoce la ubicación exacta, la operación puede realizarse en tiempo constante O(1); de lo contrario, se requiere una búsqueda previa, resultando también O(n).
* Movimiento de mensajes: combina las operaciones anteriores, por lo tanto, su eficiencia es aproximadamente O(2n), que en la práctica se simplifica a O(n).

Aunque la recursividad puede aumentar el uso de memoria en la pila de llamadas, su claridad conceptual y la facilidad para recorrer estructuras jerárquicas justifican ampliamente su utilización.

## 5) Material Adicional: Infografía

## La infografía complementaria ilustrará la estructura del árbol de carpetas mediante un diagrama visual. En ella se representarán los siguientes elementos:

* **Carpeta raíz** en la parte superior del esquema.
* **Subcarpetas** que descienden en distintos niveles jerárquicos.
* Mensajes ubicados como nodos finales (hojas).
* Ejemplo de **recorrido recursivo** con flechas que indican cómo se va descendiendo y retornando en el árbol.
* Explicaciones breves sobre las funciones de búsqueda, inserción y movimiento.

## 6) Conclusión:

## El uso de estructuras de datos recursivas, como los árboles generales, constituye una herramienta fundamental para la organización jerárquica de información. En este trabajo, se demostró cómo la recursividad simplifica la gestión de carpetas y mensajes, facilitando operaciones de búsqueda, inserción y movimiento sin necesidad de recorrer manualmente cada nivel.

## Además, se analizó la eficiencia de las operaciones y se evidenció que, a pesar de las posibles limitaciones en el consumo de memoria, la claridad y la potencia del enfoque recursivo lo vuelven una solución altamente efectiva para este tipo de sistemas.

## Finalmente, la implementación propuesta no solo permite comprender los principios teóricos de la recursividad y las estructuras de datos, sino también su aplicación práctica en entornos reales de gestión y organización digital.