

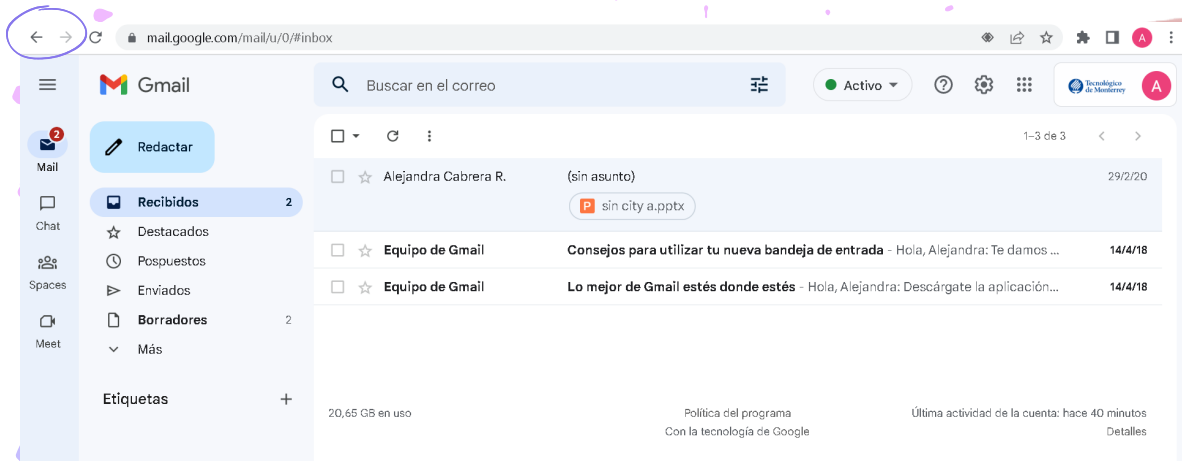
reflexión 2.3

listas doblemente ligadas


Una Double Link List se constituye por un conjunto de nodos alineados uno después del otro que tienen dos referencias. Dichas referencias apuntan al sucesor (next) y al predecesor (previous). El nodo que se encuentra al inicio de la lista no tiene un predecesor por lo que este tiene el valor de NULL, y a este nodo se le conoce como head, el primer elemento. Cuando el nodo su sucesor tiene un valor de NULL entonces sabemos que es el último nodo de nuestra lista puesto que no apunta a nada después de él. Cuando se diseñan algoritmos que definen el comportamiento de una Double Link List se deben considerar dos casos para las operaciones principales (buscar, insertar y eliminar):

- Estructura vacía (caso extremo).
- Estructura con elemento(s) (caso base).

Algunos ejemplos de las aplicaciones es las páginas web, cuando uno selecciona el botón de next y previous podemos regresar y avanzar entre las páginas web visitadas. Esto es una lista doblemente enlazada donde cada url de la página web se vuelve un nuevo nodo y este apunta a la dirección previa y a la posterior. Habilitando así esta función del navegador. También dentro de un manejador de correos web no podemos ver todos los correos en una sola página, es necesario avanzar y retroceder con las flechas en la parte posterior de la página. Esta función del manejador de correos constituye una lista doblemente ligada.



Página web con Double Link Lists



En el caso de la Actividad 2.3 el tener los elementos una vez separados y seleccionados en una lista doble, nos permite hacer funciones como el push back que optimizar el código y editar directamente la función de `.toString()` para que solo nos muestre el barco de las líneas que introduce el archivo de texto. De esta forma se puede reutilizar la lista al hacer un clear y así poder utilizarlas para cada mes, logrando que todo el código se implemente a través de ciclos for. En el desarrollo de la actividad utilizamos las operaciones principales de estas listas como eliminar, agregar con el push back y saber la longitud de esta lista para el correcto funcionamiento de los ciclos con `.length()`. Sin el uso de las listas el código estaría menos optimizado y no podríamos agregar los elementos a solamente dos listas doblemente ligadas que en este caso son Mlist y Rlist, facilitando la escritura de los archivos de texto y la solución de esta situación problema en menor cantidad de líneas sin la necesidad de tener muchos condicionales anidados.

Referencias

García, A y Solano, J. (2016). Guía práctica de estudio 08: Estructuras de datos lineales: Lista doblemente ligada y doblemente ligada circular. Facultad de Ingeniería Campus Puebla. http://odin.fi-b.unam.mx/salac/practicassEDAA/eda1_p8.pdf