Trabajo Práctico
tp2- TDA Lista y Exttras

[7541/9515] Algoritmos y Programación
II Primer cuatrimestre de 2022

Alumno:	Duque, Karla
Número de padrón:	108406
Email:	kduque@fi.uba.ar

Índice

1. Introducción

Se pide implementar un TDA Lista utilizando nodos simplemente enlazados. Para ello se brindan las firmas de las

funciones públicas a implementar y se deja a criterio del alumno la creación de las funciones privadas del TDA para

el correcto funcionamiento de la Lista cumpliendo con las buenas prácticas de programación. Adicionalmente se pide

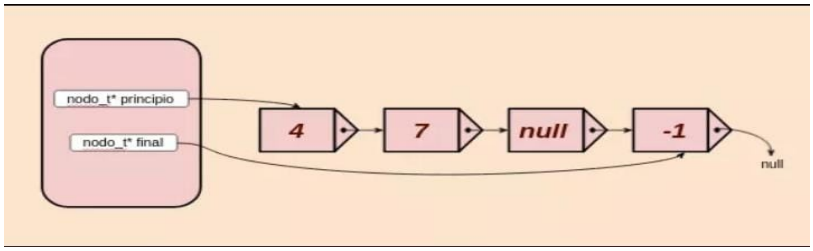
la creación de un iterador interno y uno externo para la lista, como así también reutilizar la implementación de lista

simplemente enlazada para implementar los TDAS pila y cola.

2. TDA LISTA

La idea del este TDA es simple la lista tiene un puntero a un nodo inicial y a uno final y se trata de que un nodo al poder conocer su siguiente puede crear una lista de nodos, esta lista termina cuando el ultimo nodo apunta a NULL. Básicamente una lista compuesta por este tipo de nodos se denomina lista enlazada. La Lista Simplemente Enlazada es que cada nodo conoce al nodo siguiente y tiene un campo para el elemento . Las operaciones básicas de una lista simplemente enlazada son:

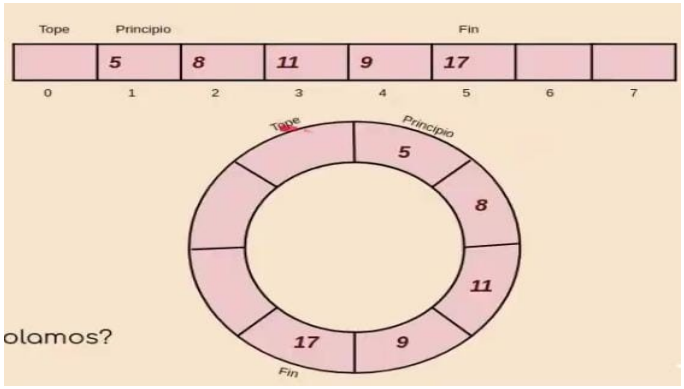
insertar, insertar en posicion, borrar, borrar de posicion,elemento en posicion, ultimo elemento, vacía, elementos, destruir.



3. TDA COLA

Una cola es una estructura que posee dos extremos por los que se realizan operaciones. Un extremo es el inicio o frente de la cola y el otro extremo es el final de la cola. Este tipo de estructura sirve para modelar procesos como la cola de un colectivo, la cola de un supermercado, el despacho de combustible, entre muchos otros.
Cola con vector estático(circular) consiste en ir agregando al final de la cola (hasta la ultima posicion) que es el caso de encolar , caso contrario es desencolar que se fija en el frente de la cola y elimina elementos, este proceso se le denomina FIFO .

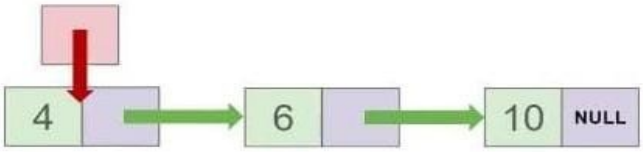
en el caso de querer encolar un elemento y ya se ha alcanzado el máximo del vector se fija en donde esta el tope y si donde esta el tope esta vació se encola en esa posicion.
El conjunto básico de operaciones para el caso de una cola es:
crear(complejidad O(1)), encolar(complejidad O(1)), desencolar(complejidad O(1)), primero ,esta vacia,destruir



4. TDA PILA

La idea del este TDA es simple la lista tiene un puntero a un nodo inicial y a uno final y se trata de que un nodo al poder conocer su siguiente puede crear una pila de nodos, esta lista termina cuando el ultimo nodo apunta a NULL. La pila Simplemente Enlazada es que cada nodo conoce al nodo siguiente y tiene un campo para el elemento .

Una pila es una colección ordenada de elementos en la que pueden insertarse y eliminarse por un extremo,denominado tope. Normalmente existe un conjunto de operaciones que se puede realizar sobre un tda:crear, apilar ,desapilar, tope esta vaciá ,destruir.
Complejidad de desencolar es O(N) porque necesito encontrar al nodo anterior al que quiero eliminar, encolar es O(1) ya que tengo un putero al primer nodo, destruir es de complejidad O(N) ya que tendría que recorrer e ir liberando memoria.



5. Detalles de implementación

la implementación cuenta con varios archivos :

1) archivo principal(pruebas.c).

2) bibliotecas

a) lista.h

b) pila.h

c) cola.h

1. **El archivo principal:** desarrolla la secuencia base del trabajo practico

2. **bibliotecas**

a)lista :contiene la funciones para el TDA lista: insertar elementos,quitar elementos, una función que devuelve la cantidad de elementos , otro elemento de primer nodo, el elemento del ultimo nodo,buscar elementos , funciones para iterador externo, función para liberar memoria etc.

b) cola : contiene la funciones para el TDA cola: insertar elementos,quitar elementos, una función que devuelve la cantidad de elementos , función que el elemento del primer nodo, función para liberar memoria etc.

c) pila : contiene la funciones para el TDA pila: insertar elementos,quitar elementos, una función que devuelve la cantidad de elementos , una función que devuelve el elemento del ultimo nodo, función para liberar memoria etc.

5.1 Detalles de alguna función

lista_insertar_en_posicion_arbitraria: Inserta un elemento en la posición indicada(a partir de 1), donde 1 es insertar como segundo elemento y 2 es insertar luego del segundo elemento, es decir en posiciones del medio(ni en el inicio ni en el final) de la lista. Devuelve la lista. Se utiliza memoria dinámica y es liberada antes de finalizar el programa

5.2 Detalle de otra función

lista_quitar_de_posicion_arbitraria: Quita de la lista el elemento que se encuentra en la posición indicada(a partir de 1), donde 1 es el segundo elemento de la lista, es decir en posiciones del medio(ni en el inicio ni en el final) de la lista. Devuelve el elemento removido de la lista. Antes de devolver el elemento libera el nodo que se queria eliminar

5.3 Complejidad de implementación

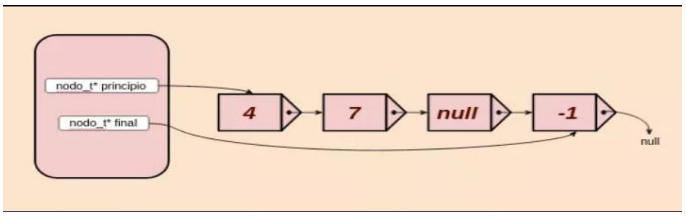
lista_insertar es O(1) porque solo hago intercambio de punteros
lista_insertar_en_posicion:
Caso 1: la posicion es mayor o igual a la cantidad de elementos
Complejidad es O(1) porque llamo a una función y hago intercambio de punteros
Caso 2: posicion del medio
Complejidad es O(N) ya que tengo que recorrer la lista
Caso 3 : posicion 0
Complejidad es O(1) porque solo se intercambian punteros

lista_quitar es O(N) ya que tengo que recorrer la lista
lista_quitar_en_posicion:
Caso 1: la posicion es mayor o igual a la ultima posicion llena
Complejidad es es O(N) ya que tengo que recorrer la lista
Caso 2: posicion del medio
Complejidad es O(N) ya que tengo que recorrer la lista
Caso 3 : posicion 0
Complejidad es O(1) porque solo se intercambian punteros

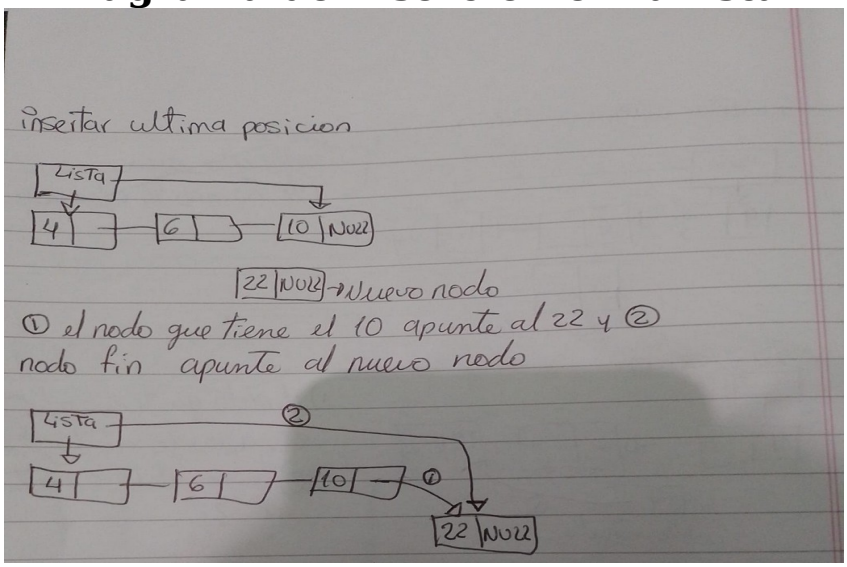
lista_con_cada_elemento,lista_buscar_elemento,lista_elemento_en_posicion y las funciones de destrucción son O(N)
Las demás funciones publicas son O(1)

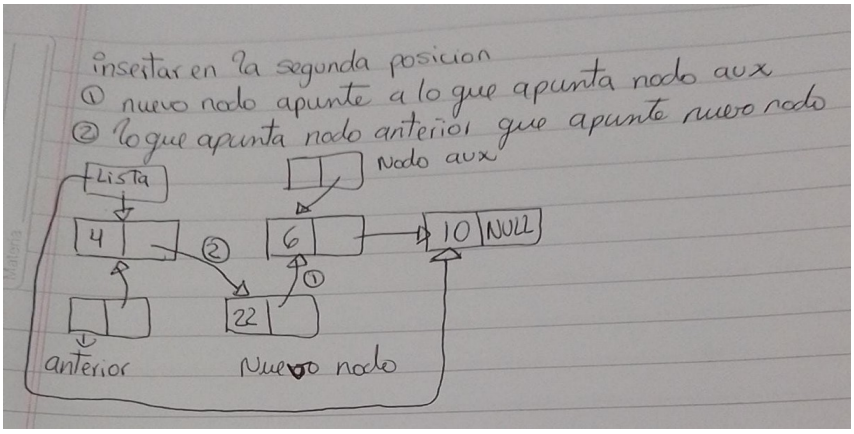
6. Diagramas

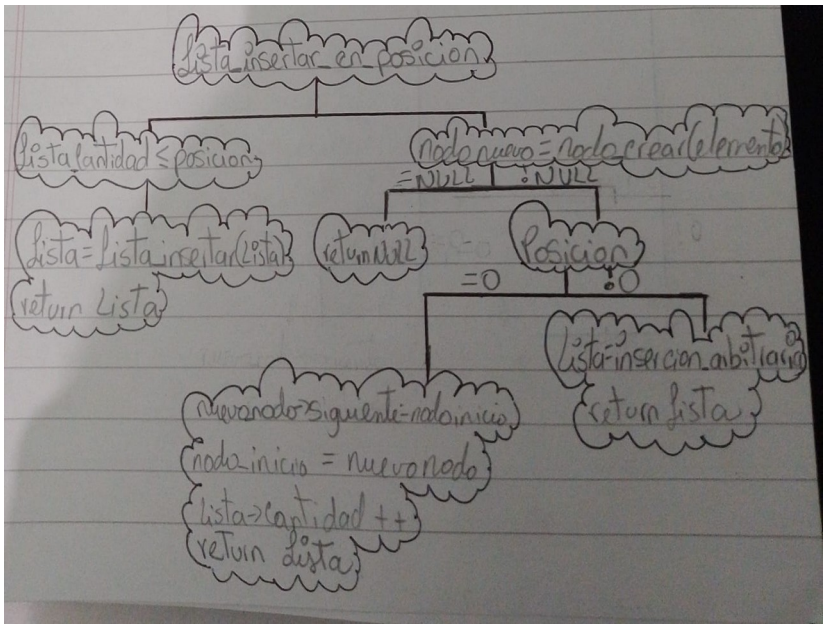
6.1 Diagrama de LISTA



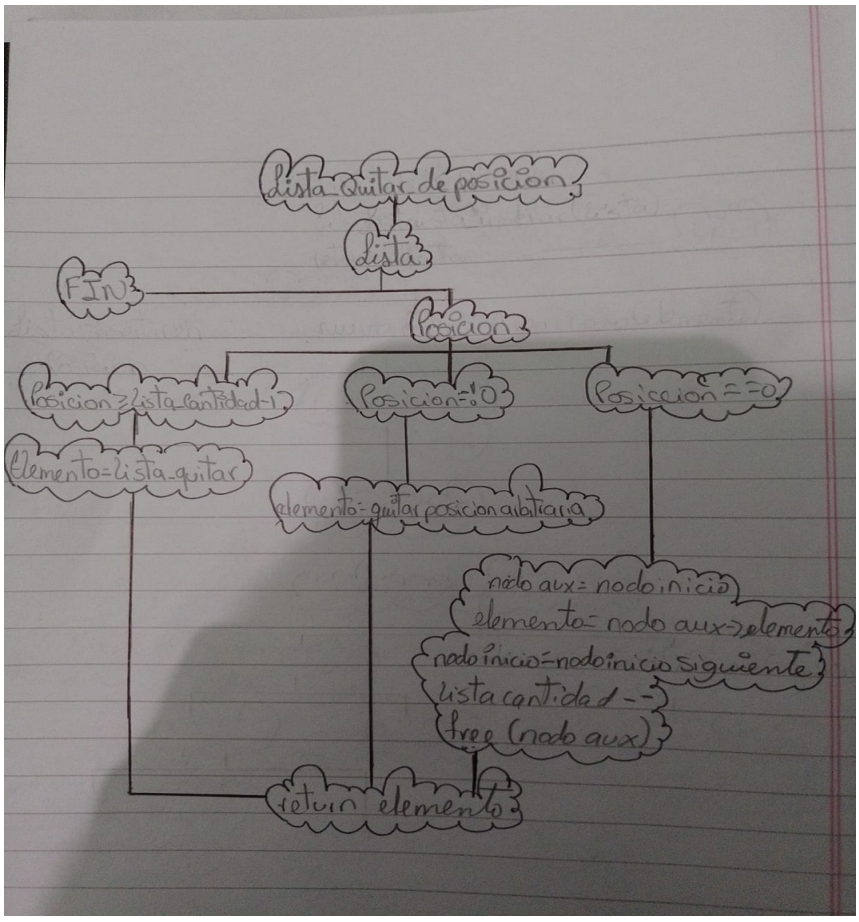
6.1 Diagrama de inserción en la lista







6.2 Diagrama quitar en posicion de la lista



6.1 Diagrama de lista destruir