Laboratorio Nro. 3: Utilización de Listas Enlazadas y Listas Hechas con Arreglos

Alejandro Murillo González
Universidad Eafit
Medellín, Colombia
amurillog@eafit.edu.co

Juan Pablo Vidal Correa
Universidad Eafit
Medellín, Colombia
Jpvidalc@ eafit.edu.co

3) *Ejercicios en línea sin documentación HTML en GitHub:*

3.1 Completar la siguiente tabla con la complejidad:

	ArrayList	LinkedList
Ejercicio 1.1	O(n ^2)	O(n)
Ejercicio 1.2	O(1)	O(1)
Ejercicio 1.3	O(n ^2)	O(n)
Ejercicio 1.4	O(n ^2)	O(n)

3.2 Expliquen con sus propias palabras cómo funciona la implementación del ejercicio 2.1:

En el ejercicio 2.1 se debe tomar el string, y se debe inspeccionar cada letra, para se debe tener en cuenta cada vez que haya un carácter "[" o "]", para añadirlos en orden a la lista, es decir cada vez que se encuentre un "[" , se inserta en la lista este carácter junto con todas las palabras que se encuentran a la derecha , hasta que se encuentre otro carácter "[" o "]", y este se inserta al inicio, para que siempre salga al inicio, lo mismo ocurre con el otro carácter pero este se inserta siempre al final, para que salga al final; los demás caracteres se insertar en después de que se hayan añadido los caracteres con "[".

3.3 Calculen la complejidad del ejercicio realizado en el numeral 2.1

La complejidad de este ejercicio es O(n), ya que eligiendo el peor de los casos solo tiene un ciclo.

Nota: no hay ciclos anidados.

Código:

```
def excercise_2():

    def process_token(string):

        if string == "":
            return ""

        newString = LinkedList.LinkedList()

        placeBefore = False
        substr = ""

        for character in string: // C + 1

            if character == "[":
                newString.add_at_start(substr) if placeBefore else newString.add(substr)
                substr = ""  // C
                placeBefore = True

            elif character == "]":
                newString.add_at_start(substr) if placeBefore else newString.add(substr)
                substr = ""  // C
                placeBefore = False

            else:  // C
                substr += character

        newString.add_at_start(substr) if placeBefore else newString.add(substr)

        finalStr = ""
        current = newString.head
```

```
    while current != None:  // C + n
        finalStr += current.data
        current = current.next

    return finalStr

proccessed_tokens = []
tokens = []
token = input()

while token != "EOF":  // C + n
    proccessed_tokens.append(process_token(token))
    tokens.append(token)
    token = input()

for p in proccessed_tokens: // (C + 1)
    print(p)
```

T(n) = O(C + n + C + n + C + 1 + C + 1 == T(n) es O(n)

3.4 Expliquen con sus palabras las variables (qué es 'n', qué es 'm', etc.) del cálculo de complejidad del numeral 3.3

Las variables "n", "m" y/o "y" son las variables de entrada, esto con el objetivo de facilitar los cálculos de complejidad.

4) Simulacro de Parcial

1. A
2. C
3.
   - while (q.size() > 1)
   - for(int i = 1; i <= num; i++)
   - q.add(q.remove());
   - return q.remove()

DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co

5) Lectura recomendada (opcional)

a) Resumen:

Lista Enlazada:

La lista enlazada es una estructura de datos que se utiliza para almacenar información. Esta cumple con las siguientes propiedades:

- Los elementos sucesivos están conectados por apuntadores o punteros
- El último elemento es nulo
- Puede variar en tamaño durante su ejecución
- Puede ser tan grande como se requiera
- No gasta innecesariamente espacio de memoria, solo gasta un poco más para los apuntadores

Ventajas:
La mayor ventaja es que se puede expandir constantemente.

Desventajas:
El tiempo de acceso para elementos individuales es muy largo comprado con otro tipo de estructuras como los arreglos. En el peor de los casos le toma O(n), mientras que en el arreglo es O (1).

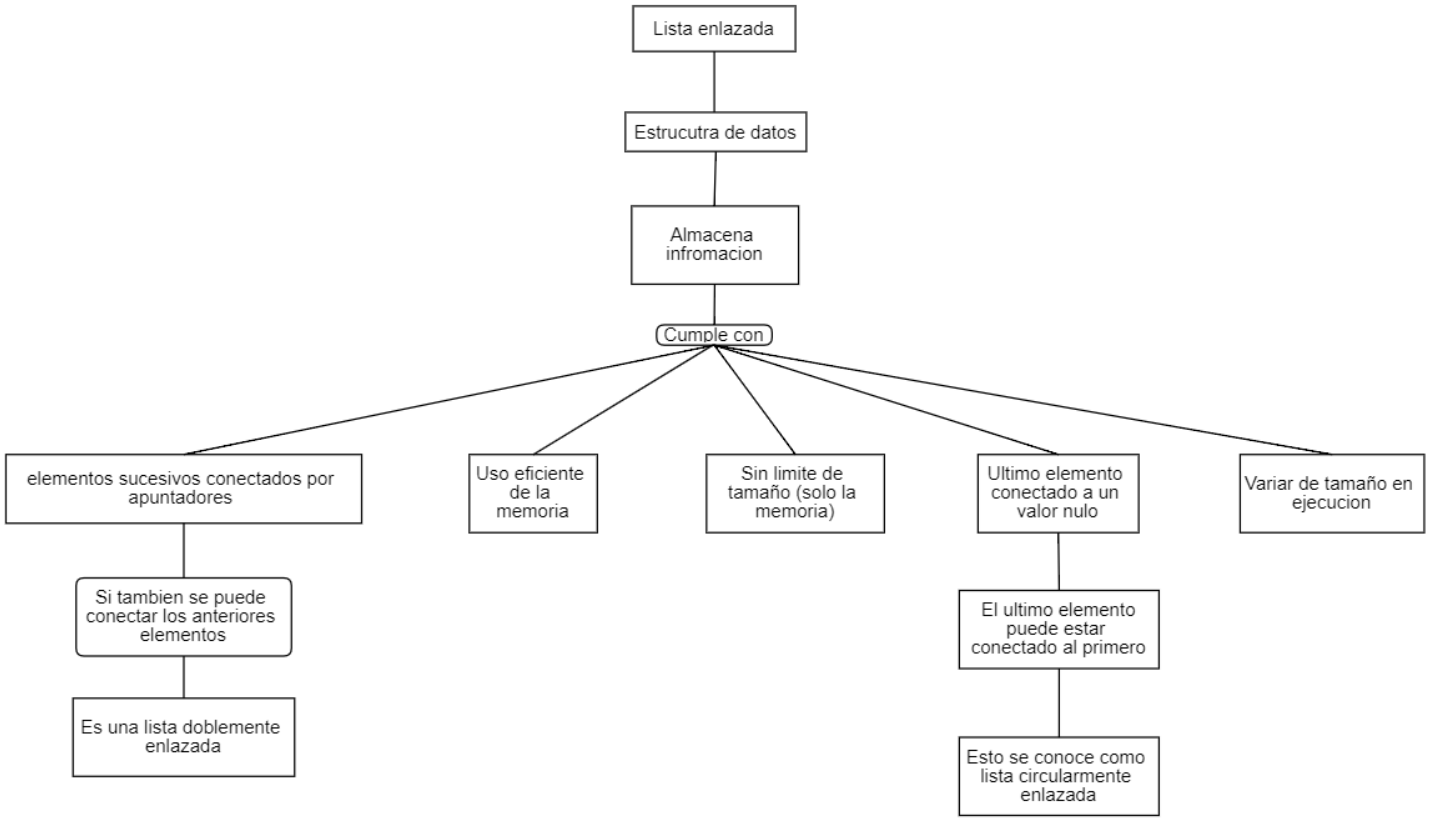
Tipos de listas enlazadas:

Lista enlazada simple:
Cada nodo tiene un puntero para llegar al siguiente elemento, el enlace del último nodo es nulo, lo cual indica el final.

Lista enlazada doble:
Es parecida a la lista enlazada simple, pero además los nodos tienen un puntero para retroceder, y llegar al elemento anterior.

Lista enlazada circular:
Es parecido a una lista simple, pero el final de la lista no es nulo, si no que continúa apuntando al primer nodo.

b) Mapa Conceptual:



6) Trabajo en Equipo y Progreso Gradual (Opcional)

a) Actas de reunión:

Integrante	Fecha	Hecho	Haciendo	Por Hacer
Murillo	20/09/2017	Implemento códigos del numeral 1	Calcular complejidad	
Vidal	22/09/2017	Implemento código en línea (2.1)	Calculando complejidad	leer lectura opcional y resumen

DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co

Murillo	22/09/2017	responder preguntas	Organizar texto-informe de laboratorio	Agregar partes opcionales de vidal
Vidal	23/09/2017	implemento resumen	mapa conceptual de lectura	Simulacro del parcial

DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co