Laboratorio Nro. 3
Listas Enlazadas y Vectores Dinámicos

Juan Sebastián Guerra Hernández
Universidad Eafit
Medellín, Colombia
jsguerrah@eafit.edu.co

Jacobo Rave Londoño
Universidad Eafit
Medellín, Colombia
jravel@eafit.edu.co

3) Simulacro de preguntas de sustentación de Proyectos

3.1 Complejidad del problema 1.1

Leer los datos de los ficheros toma:

O(n*m) para la lectura de los vértices y O(n*m) para lectura de los arcos. En ambos casos la n representa el número de filas, mientras que m el número de columnas. Por multiplicidad, podemos afirmar que T(n, m) es O(n*m).

Construir un objeto MapaCiudad toma:

O(n) para generar un objeto de ese tipo. Esto sucede porque en el constructor se hace llamado del método asignarElementos, quien tiene complejidad O(n).

- En el caso de usar listas enlazadas o vectores dinámicos, buscar en una lista requiere de O(n) en ambos. Esto haría que al final la complejidad fuera de $O(n^2)$.

Operation	LinkedList time complexity	ArrayList time complexity
Insert at last index	O(1)	O(1) (If array copy operation is Considered then O(N))
Insert at given index	O(N)	O(N)
Search by value	O(N)	O(N)
Get by index	O(N)	O(1)
Remove by value	O(N)	O(N)
Remove by index	O(N)	O(N)

- La ventaja de usar HashMap es que para comprobar que dentro de la estructura existe una key, lo puede hacer en O(1) con el método containsKey() si está bien implementado.

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

3.2 Explicación del método mostrarEnPantalla():

Para comenzar creamos una lista enlazada de String y un objeto StringBuilder. El primero es donde iremos almacenando las cadenas de texto, mientras que el StringBuilder nos sirve para ir generando un String adicionándole un caracter a lo largo de las iteraciones.

También generamos un boolean botonInicio, quien va a determinar si fue presionado (al empezar a escribir se hace desde el inicio, así que esta inicializado con true).

Dentro del ciclo, quien tiene la función de recorrer cada caracter de una palabra, preguntamos si se trata de un corchete: si no se trata de ningún corchete, adicionamos ese caracter a nuestra cadena; si se cumple la condición, agregamos nuestro String (los caracteres que recorrimos hasta el momento) a la lista enlazada – de acuerdo a la condición de botonInicio se hace en el final o en el inicio – y luego, dependiendo del tipo de corchete se cambia el valor de botonInicio, interpretándolo como que volvió a presionar Inicio o, en su defecto, Fin. Al terminar, reestablecemos el valor de la cadena y continuamos las iteraciones hasta que se haya recorrido toda la palabra. Sin embargo, si estamos en la última posición, simplemente agregaremos ese caracter como de costumbre y de acuerdo con la condición lo agregaremos a nuestra lista.

Al final, como nuestra intención es devolver un String de lo que se ve en pantalla, utilizamos un ciclo para recorrer la lista sacando su primer elemento con pollFirst() y concatenándolo a nuestro String hasta que la lista quede vacía. Finalmente, devolver nuestro String de como se vería el texto en pantalla con base en lo ingresado en el teclado.

3.3 Complejidad del algoritmo de teclado roto:

Su peor caso es cuando hay solo un carácter entre cada corchete lo que hace que el tamaño de la lista sea la mitad de la cantidad de caracteres. Sin embargo, en el primer ciclo se recorre toda la palabra, caracter por caracter.

$$T(n) = c_1 n + c_2 + (c_3 n)/2$$
$$T(n) \text{ es } O(n)$$

3.4. *n* es el número de caracteres que contiene el escrito a través del teclado roto

4) Simulacro de Parcial

4.1.1. [Opc] b
4.1.2. [Opc] b

4.2. [Opc] c

4.4.1. token
4.4.2. b

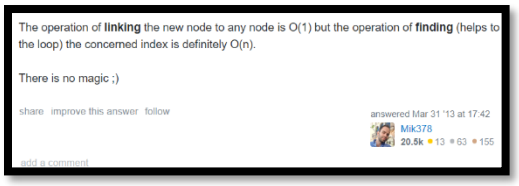
4.5. [Opc] a

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

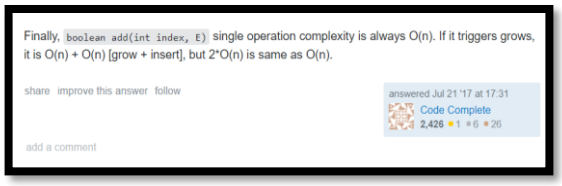
4.6. [Opc] a

4.8 O(n) y O(n) (Ninguna de las anteriores)

Listas enlazadas



Vectores dinámicos (ArrayList en Java)



En las listas enlazadas debemos de ir hasta al nodo anterior para ahí si poder insertar, por lo que es O(n); el add(E, index) de ArrayList implica ir hasta la posición en la que queremos agregar para luego si poder hacerlo.

4.8.1. [Opc] a
4.8.2. [Opc] c
4.8.3. [Opc] c

4.10.1. b
4.10.2. b

4.11.1. while(s1.size() > s2.size())
4.11.2. s2.push(s1.get(s2.size() + 1));
4.11.3. return s1.head;

4.12.1. [Opc] iv
4.12.2. [Opc] i

4.13.1. iii
4.13.2. iii

4.14. [Opc] iii

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473