Laboratorio Nro. 3
Listas Enlazadas, Vectores Dinámicos y tablas de Hash

Luis Fernando Vargas Agudelo
Universidad Eafit
Medellín, Colombia
lvarga12@eafit.edu.co

Tomás Bedoya Henao
Universidad Eafit
Medellín, Colombia
tbedoyah@eafit.edu.co

3) Simulacro de preguntas de sustentación de Proyectos

3.2

La implementación de este ejercicio consta principalmente de dos partes, una es la construcción de la clase linkedlist la cual se compone por los métodos push, y append, estos métodos insertan un nodo al inicio y al final de la lista respectivamente, por último, el método printlist, imprime cada uno de sus elementos de forma secuencial, es decir no como una sola variable que contenga todos los valores, si no que muestra cada uno de sus nodos de forma independiente.
La otra parte principal es la función text, esta función recibe el texto en desorden con el objetivo de procesarlo y guardarlo en la lista anteriormente mencionada, por último, imprime los nuevos valores de la lista.

La función text funciona de la siguiente manera, primero inicializa una lista vacía en cuál se guardaran los datos, posteriormente crea una variable auxiliar donde se reemplazan los valores de “[” y “]” por “,”, el objetivo de este reemplazo es poder hacer un Split por cada “,” y guardarlo en la misma variable aux como tupla, es decir que esta variable aux contiene los bloques de caracteres que deben ser enviados al inicio o al final de la lista.

Una vez esta construida esta variable la función le indica a la lista que guarde el valor inicial de la tupla aux, este sirve como punto de referencia par las siguientes operaciones. Posteriormente, con la ayuda de un contador la función comienza a recorrer el texto de entrada carácter por carácter, buscando si en algún punto se encuentra “[” o “]”, si esto llega a suceder la función le indica a la lista que guarde el bloque de caracteres que se encuentran en la variable aux correspondientes al carácter encontrado.
Es decir, si la función encuentra un “[” quiere decir que el valor de aux[1] debe ir al inicio de la lista, si continua y encuentra otro valor que sea “]” quiere decir que el valor de aux[2] debe ir al final de la lista, esto realiza hasta se recorran todos los caracteres de la variable text. Por último, la función imprime los valores de la lista, mostrando de esta manera el orden correcto del texto.

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

3.3

La complejidad del método push() es O(1) debido a que siempre realiza una cantidad constante de trabajo

La complejidad del método append() es O(n), donde n hace referencia a la cantidad de nodos en la lista

La complejidad del método printList() es O(n), ya que debe imprimir la cantidad de nodos que se encuentren en la lista

La complejidad de la función texto es de O(n), donde n hace referencia a la cantidad de caracteres que tiene el texto inicial

3.4

Las variables n y m son ambos requerimientos y están directamente relacionados con la complejidad de los ejercicios, esto debido a que según el caso que se realice en sus llamadas esta tendrá que recorrer en el mejor de los casos solo las filas de su estructura de datos con complejidad O(n) o inclusive ir directamente a un valor específico o cuál tendría una complejidad de O (1).
Pero puede llegar a existir casos en los cuales es necesario que la estructura de datos sea recorrida en su totalidad de filas y además en es fila recorre la totalidad de sus columnas, esto sucede cuando se desea buscar un valor especifico y no se conoce su ubicación.
Debido a situaciones como la anteriormente mencionada hace que las funciones puedan llegar a tener una complejidad tipo O(n*m) donde n hace referencia a la cantidad de filas que hay en una estructura de datos y m a la cantidad de columnas de esta. Por situaciones como esta el tiempo de ejecución para estructuras muy grandes se puede ver linealmente afectado.

4) Simulacro de Parcial

Nota: Debido a que la numeración se encuentra mal realizada en el laboratorio, la numeración en el informe se encuentra igual a como se encuentra descrita en el laboratorio de github, esto con el fin de hacer referencia a la pregunta indicada.

4.1

4.1.1 – A
4.1.2 – A

4.2 – *C*

4.3

4.3.1 – D
4.3.2 – D

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

4.4 – A

4.5 – B

4.6 – C

4.7

4.8.1 - B
4.8.2 - C
4.8.3 - C

4.10

4.9.1 – D
4.9.2 – A
4.9.3 – B

4.11

4.11.1 – D
4.11.2 – D

4.12

4.11.1 – s1.isNotEmpty
4.11.2 – s1.pop()
4.11.3 – s2.pop()

4.13

4.12.1 – IV
4.12.2 – I

4.13

4.13.1 – II
4.13.2 – I

4.14 – 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