|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
|  | Guía de laboratorio 5 |

“*…Para esa época, Melquíades había envejecido con una rapidez asombrosa. En sus primeros viajes parecía tener la misma edad de José Arcadio Buendía. Pero mientras éste conservaba su fuerza descomunal, que le permitía derribar un caballo agarrándolo por las orejas, el gitano parecía estragado por una dolencia tenaz. Era, en realidad, el resultado de múltiples y raras enfermedades contraídas en sus incontables viajes alrededor del mundo. Según él mismo le contó a José Arcadio Buendía mientras lo ayudaba a montar el laboratorio, la muerte lo seguía a todas partes, husmeándole los pantalones, pero sin decidirse a darle el zarpazo final. Era un fugitivo de cuantas plagas y catástrofes habían flagelado al género humano. Sobrevivió a la pelagra en Persia, al escorbuto en el archipiélago de Malasia, a la lepra en Alejandría, al beriberi en el Japón, a la peste bubónica en Madagascar, al terremoto de Sicilia y a un naufragio multitudinario en el estrecho de Magallanes. Aquel ser prodigioso que decía poseer las claves de Nostradamus, era un hombre lúgubre, envuelto en un aura triste, con una mirada asiática que parecía conocer el otro lado de las cosas. Usaba un sombrero grande y negro, como las alas extendidas de un cuervo, y un chaleco de terciopelo patinado por el verdín de los siglos. Pero a pesar de su inmensa sabiduría y de su ámbito misterioso, tenía un peso humano, una condición terrestre que lomantenía enredado en los minúsculos problemas de la vida cotidiana. Se quejaba de dolencias de viejo, sufría por los más insignificantes percances económicos y había dejado de reír desde hacía mucho tiempo, porque el escorbuto le había arrancado los dientes. El sofocante mediodía en que reveló sus secretos, José Arcadio Buendía tuvo la certidumbre de que aquél era el principio de una grande amistad. Los niños se asombraron con sus relatos fantásticos. Aureliano, que no tenía entonces más de cinco años, había de recordarlo por el resto de su vida como lo vio aquella tarde, sentado contra la claridad metálica y reverberante de la ventana, alumbrando con su pro-funda voz de órgano los territorios más oscuros de la imaginación, mientras chorreaba por sus sienes la grasa derretida por el calor. José Arcadio, su hermano mayor, había de transmitir aquella imagen maravillosa, como un recuerdo hereditario, a toda su descendencia. Úrsula, en cambio, conservó un mal recuerdo de aquella visita, porque entró al cuarto en el momento en que Melquíades rompió por distracción un frasco de bicloruro de mercurio.*

*-Es el olor del demonio -dijo ella. -En absoluto -corrigió Melquíades-. Está comprobado que el demonio tiene propiedades sulfúricas, y esto no es más que un poco de solimán. Siempre didáctico, hizo una sabia exposición sobre las virtudes diabólicas del cinabrio, pero Úrsula no le hizo caso, sino que se llevó los niños a rezar. Aquel olor mordiente quedaría para siempre en su memoria, vinculado al recuerdo de Melquíades …”*

***Tomado de Cien Años de Soledad, Gabo.***

**Objetivo:**

En este laboratorio se construye una aplicación de una lista enlazada simple con las operaciones básicas que se realizan en la lista.

**Preparación y creación del proyecto:**

1. Crear las clases, digitar los métodos y la correspondiente documentación, de las clases Lista y Nodo (archivo adjunto “Clase Nodo y Lista.doc”) a continuación estudie que hace cada método.

Clase Lista: Modela las opciones del menú, para crear la lista, anexar y borrar los nodos, buscar elementos en la lista, e impresión de la información almacenada en la lista.

Clase Nodo: Clase que modela el elemento básico de la lista conformado por los campos dato de tipo entero y la auto-referencia denominada enlace.

1. Crear y copiar la clase AplicaciónLista (archivo adjunto “Clase AplicacionLista.doc”) depurar errores y estudiar cada método.

**CLASE AplicaciónLista**

import java.util.Scanner;

public class AplicacionLista {

private Lista lst;

private String menuOpciones;

private String opcion;

private int numero;

private int dato;

private int datoInicio;

private Scanner input;

public static void main(String []args) {

AplicacionLista lista = new AplicacionLista();

lista.leerMenu();

}

/\*\*

\* constructor de la clase ListaApp

\*/

public AplicacionLista() {

this.lst=new Lista ();

this.menuOpciones="Listas Enlazadas Simples\n" +

"=========================\n"+

"1. Crear lista\n2. Imprimir lista" +

"\n3. Anexar nodos Inicio \n4. Anexar nodos final\n5. Buscar nodos\n6. Borrar nodo inicio" +"\n7. Borrar nodo final\n8. Salir";

this.opcion="";

this.dato=0;

this.numero=0;

this.datoInicio=0;

this.input = new Scanner(System.in);

}

/\*\*

\* Metodo que tiene las opciones del menu de la lista

\*/

public void leerMenu(){

do{

listarMenu(menuOpciones);

numero=input.nextInt();

switch (numero){

case 1: crearLista(); break;

case 2: listar(); break;

case 3: anexarNodoInicio(); break;

case 4: anexarNodoFinal(); break;

case 5: buscarNodo(); break;

case 6: borrarDatoInicio(); break;

case 7: borrarDatoFinal(); break;

}

}while(numero !=8);

}

/\*\*

\* Metodo que imprime el menu de opciones de la aplicacion

\* @param menu

\*/

public void listarMenu(String menu){

System.out.println(menu);

}

/\*\*

\* creacion de la lista con 10 nodos

\*/

public void crearLista() {

lst=new Lista ();

for (int dato=1; dato<=10; dato++){

lst.anexarFinalLista(dato);

this.dato=dato;

}

listar();

}

/\*\*

\* Se anexa un nodo al inicio de la lista

\*/

public void anexarNodoInicio() {

lst.anexarInicioLista(datoInicio);

--datoInicio;

listar();

}

/\*\*

\* Se anexa un nodo al final de la lista

\*/

public void anexarNodoFinal() {

dato++;

lst.anexarFinalLista(dato);

listar();

}

/\*\*

\* Se imprime la lista

\*/

public void listar() {

String laLista=lst.imprimirLista();

System.out.println(laLista);

}

/\*\*

\* metodo que a partir de un valor invoca el metodo a buscar

\* este numero en la lista

\*/

public void buscarNodo(){

int n=0;

System.out.println("digite el numero a buscar");

n=input.nextInt();

if(lst.buscarNodo(n)){

System.out.println("valor encontrado");

}else{

System.out.println("valor no encontrado");

}

listar();

}

/\*\*

\* metodo que borra el primer nodo de la lista

\*/

public void borrarDatoInicio(){

if(lst.borrarNodoInicio()){

System.out.println("nodo borrado");

}else{

System.out.println("valor no borrado");

}

listar();

}

/\*\*

\* metodo que borra el nodo al final de la lista

\*/

public void borrarDatoFinal(){

if(lst.borrarNodoFinal()){

System.out.println("nodo borrado");

}else{

System.out.println("valor no borrado");

}

listar();

}

1. Completar el método imprimirLista de la clase Lista.

public String imprimirLista(){

String str = "\n";

if(!esVacia()){

Nodo aux = primero;

System.out.println("\nLISTA");

while(aux.getEnlace()!= null){

System.out.println(" "+aux.getDato());

aux = aux.getEnlace();

}

}else{

str ="Lista vacia";

}

return str;

}public String imprimirLista(){

String str = "\n";

if(!esVacia()){

Nodo aux = primero;

System.out.println("\nLISTA");

while(aux.getEnlace()!= null){

System.out.println(" "+aux.getDato());

aux = aux.getEnlace();

}

}else{

str ="Lista vacia";

}

return str;

}

1. Realizar un informe con los siguientes datos:
2. Relacione los Errores encontrados y cómo los solucionó.
3. Explique con sus palabras los métodos borrarNodoInicio() y anexarFinalLista (int dato)
4. Explique cómo funciona el método esVacia ()

**Errores:**

* El main estaba situado de ultimas y daba error en la ejecución del programa, se solucionaba simplemente colocándolo después de las declaraciones de variables.
* Error al utilizar el Scanner para ingreso de datos por consola, se debía importar la librería java.util.Scanner.
* En el método anexarNodoInicio() en una línea faltaba incluir una ‘a’ al final de la palabra Lista en la línea lst.anexarInicioList(datoInicio).
* En el método borrarDatoInicio() faltaba especificar el objeto que se utiliza, se debia agregar lst. en la línea **if**(borrarNodoInicio()){.

**Método borrarNodoInicio()**

* Si la lista está vacía devuelve un valor false, pero si tiene nodos lo que hace es crear un nodo *temp* que se iguala *primero*, después mueve el nodo *primero* a la posición a la que estaba apuntando para después poder borrar el nodo *temp*.

**Metodo anexarFinalLista(int dato)**

* Crea un nuevo nodo, si la lista se encuentra vacía solo determina el nuevo nodo como el nodo primero y último, pero si ya hay algo en esa lista lo que hace es asignarle al nodo que esta al final la dirección de nuevo para que pueda quedar como ultimo.

**Metodo esVacia()**

* Solo revisa si el nodo primero no es nulo y dependiendo de si es o no retorna un valor booleano true o false.