Universidad Nacional Autónoma
de México

Facultad de Estudios Superiores
Aragón

Tarea 5: DoubleLinkedList

Estructura de Datos

Prof. Hernández Cabrera Jesús

Grupo: 1306

Angeles Mermejo Octavio Emiliano

Código Fuente Nodo Doble:

```java
package fes.aragon.clases;

25 usages
public class NodoDoble <T>{
    5 usages
    private T dato;
    3 usages
    private NodoDoble<T> siguiente;
    3 usages
    private NodoDoble<T> anterior;

    no usages
    public NodoDoble() {
    }

    3 usages
    public NodoDoble(T dato) {
        this.dato = dato;
    }

    no usages
    public NodoDoble(T dato, NodoDoble<T> siguiente, NodoDoble<T> anterior) {
        this.dato = dato;
        this.siguiente = siguiente;
        this.anterior = anterior;
    }

    6 usages
    public T getDato() {
        return dato;
    }

    1 usage
    public void setDato(T dato) {
        this.dato = dato;
    }

    13 usages
    public NodoDoble<T> getSiguiente() {
        return siguiente;
    }

    6 usages
    public void setSiguiente(NodoDoble<T> siguiente) {
        this.siguiente = siguiente;
    }

    4 usages
    public NodoDoble<T> getAnterior() {
        return anterior;
    }

    6 usages
    public void setAnterior(NodoDoble<T> anterior) {
        this.anterior = anterior;
    }

    @Override
    public String toString() {
        return dato + "| -> |";
    }
}
```

Código Fuente DoubleLinkedList:

```java
package fes.aragon.clases;

3 usages
public class DoubleLinkedList<T> {
    21 usages
    private NodoDoble<T> head ;
    14 usages
    private NodoDoble<T> tail;
    no usages
    private int direccion;
    10 usages
    private int tamanio;

    1 usage
    public DoubleLinkedList() {
    }

    no usages
    public boolean estaVacio() {
        boolean resultado = false;
        if(this.head == null && this.tail == null){
            System.out.println("La lista esta vacia");
            return true;
        }
        System.out.println("La lista no esta vacia");
        return resultado;
    }

    no usages
    public int getTamanio(){
        return tamanio;
    }

    1 usage
    public void agregarAlInicio (T valor){
        NodoDoble<T> nuevo = new NodoDoble<>(valor);
        NodoDoble<T> aux = this.head;

        if(aux == null){
            this.head = nuevo;
            this.tail = nuevo;
        }else {
            this.head.setAnterior(nuevo);
            nuevo.setSiguiente(this.head);
            this.head = nuevo;
        }
        tamanio++;
    }

    5 usages
    public void agregarAlFinal (T valor){
        NodoDoble<T> nuevo = new NodoDoble<>(valor);
        NodoDoble<T> aux = this.head;

        if(aux == null){
            this.head = nuevo;
            this.tail = nuevo;
        }else {
            this.tail.setSiguiente(nuevo);
            nuevo.setAnterior(this.tail);
            this.tail = nuevo;
        }
        tamanio++;
```

```java
    }

    no usages
    public void agregarDespuesDe (T referencia , T valor){
        NodoDoble<T> aux = this.head;
        while (aux != null){
            if(aux.getDato().equals(referencia)){
                NodoDoble<T> nuevo = new NodoDoble<>(valor);
                nuevo.setSiguiente(nuevo.getSiguiente());
                nuevo.setAnterior(aux);
                if(aux.getSiguiente() != null){
                    aux.getSiguiente().setAnterior(nuevo);
                }else{
                    this.tail = nuevo;
                }
                aux.setSiguiente(nuevo);
                tamanio++;
                return;
            }
            aux = aux.getSiguiente();
        }
        System.out.println("No se encontro la referencia");
    }

    no usages
    public T obtener (int posicion){
        if (posicion < 0 || posicion >= tamanio){
            System.out.println("fuera del rango");
        }

        NodoDoble<T> aux = this.head;
        for (int i = 0; i < posicion; i++) {
            aux = aux.getSiguiente();
        }
        return aux.getDato();
    }

    1 usage
    public void eliminarElPrimero() {
        if (this.head == null) {
            System.out.println("No hay elementos por eliminar");
            return;
        }

        this.head = head.getSiguiente();
        if (this.head != null) {
            this.head.setAnterior(null);
        } else {
            this.tail = null;
        }
        tamanio--;
    }

    1 usage
    public void eliminarElFinal() {
        if (this.tail == null) {
            System.out.println("No hay elementos por eliminar");
            return;
        }

        this.tail = tail.getAnterior();
        if (this.tail != null) {
            this.tail.setSiguiente(null);
```

```
        } else {
            this.head = null;
        }
        tamanio--;
    }

    1 usage
    public void eliminar(int posicion) {
        if (posicion < 0 || posicion >= tamanio) {
            System.out.println("Elemento fuera del rango");
            return;
        }

        if (this.head == null) {
            System.out.println("No hay elementos por eliminar");
            return;
        }

        if (posicion == 0) {
            eliminarElPrimero();
        } else if (posicion == tamanio - 1) {
            eliminarElFinal();
        } else {
            NodoDoble<T> aux = this.head;
            for (int i = 0; i < posicion; i++) {
                aux = aux.getSiguiente();
            }
            aux.getAnterior().setSiguiente(aux.getSiguiente());
            if (aux.getSiguiente() != null) {
                aux.getSiguiente().setAnterior(aux.getAnterior());
            }
        }
        tamanio--;
    }

    1 usage
    public int buscar(T valor){
        NodoDoble<T> aux = this.head;
        int indice = 1;
        while (aux != null){
            if(aux.getDato().equals(valor)){
                return indice;
            }
            aux=aux.getSiguiente();
            indice++;
        }
        System.out.println("No se encontro el elemento");
        return -1;
    }

    1 usage
    public void actualizar (T aBuscar , T valor){
        NodoDoble<T> aux = this.head;
        while (aux != null) {
            if (aux.getDato().equals(aBuscar)) {
                aux.setDato(valor);
                return;
            }
            aux = aux.getSiguiente();
        }
        System.out.println("No se encontro el elemento");

    }
```

```java
    3 usages
    public void transversal(int direccion) {
        if (direccion == 0) {
            NodoDoble<T> aux = this.head;
            while (aux != null) {
                System.out.println(aux.getDato());
                aux = aux.getSiguiente();
            }
        } else {
            NodoDoble<T> aux = this.tail;
            while (aux != null) {
                System.out.println(aux.getDato());
                aux = aux.getAnterior();
            }
        }
    }
}
```

Código Fuente Main:

```java
package fes.aragon.inicio;

import fes.aragon.clases.DoubleLinkedList;

public class Main {
    public static void main(String[] args) {

        DoubleLinkedList<Integer> listaObjects =new DoubleLinkedList<>();
        listaObjects.agregarAlInicio( valor: 50);
        listaObjects.agregarAlFinal( valor: 60);
        listaObjects.agregarAlFinal( valor: 65);
        listaObjects.agregarAlFinal( valor: 70);
        listaObjects.agregarAlFinal( valor: 80);
        listaObjects.agregarAlFinal( valor: 90);

        System.out.println("Lista Principal");
        listaObjects.transversal( direccion: 0);
        System.out.println(" ");

        System.out.println("Lista actualizada a : Eliminar elemento de la posicion 2");
        listaObjects.eliminar( posicion: 1);
        listaObjects.transversal( direccion: 0);

        System.out.println();
        System.out.println("Lista Actualizada a: Actualizar el cuarto elemento a 88");
        listaObjects.actualizar( aBuscar: 80, valor: 88);
        listaObjects.transversal( direccion: 0);

        System.out.println();
        System.out.println("Busca el valor de 80 dentro de la lista: ");
        listaObjects.buscar( valor: 80);
    }
}
```

Ejecución del Programa:

```
/opt/java/jdk-17.0.2/bin/java -javaagent:/opt/java/idea-IC-233.14015.106/lib/idea_rt.jar=43203:/opt/java/idea-IC-233.14015.106/bin -Dfile.encoding=UTF-8
Lista Principal
50
60
65
70
80
90

Lista actualizada a : Eliminar elemento de la posicion 2
50
65
70
80
90

Lista Actualizada a: Actualizar el cuarto elemento a 88
50
65
70
88
90

Busca el valor de 80 dentro de la lista:
No se encontro el elemento

Process finished with exit code 0
```