Informe de Laboratorio 04

Tema: Arbol Binario

Nota

Estudiante	Escuela	Asignatura
Misael Marrón Lope mmarronl@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	EDA Semestre: III Código: 20220575

Laboratorio	Tema	Duración
04	Arbol Binario	

Semestre académico	Fecha de inicio	Fecha de entrega
2023 - A	2023	19 junio 2023

1. Tarea

- Implementa una clase de arbol binario de busqueda.
- Implemente una clase Node donde T es un tipo genérico, esta clase debe contener al menos dos propiedades.
- Utilizar Git para evidenciar su trabajo.
- Enviar trabajo al profesor en un repositorio GitHub Privado, dándole permisos como colaborador.

2. Equipos, materiales y temas utilizados

- Sistema Operativo Windows 10 ver. 22H2
- Eclipse IDE, Visual studio
- java 20.0.1
- Git 2.40.1.
- Cuenta en GitHub con el correo institucional.

3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- `https://github.com/MisaelMarron/eda-lab-b-23a.git`
- URL para el laboratorio 04 en el Repositorio GitHub.
- `https://github.com/MisaelMarron/eda-lab-b-23a/tree/main/lab04`

4. Actividades : La creacion de Arbol Binario de Busqueda

- Elabore un informe implementando Arboles Binarios de Busqueda con toda la lista de operaciones search(), getMin(), getMax(), parent(), son(), insert(), remove().

4.1. Commits Importantes:

Listing 1: Mi primer commit importante es cuando agregue el metodo search .

```
commit 42fe67b8679d688fa7f08e11717e9f3bff142b00
Author: Misael Marron <mmarronl@unsa.edu.pe>
Date:  Sun Jun 18 18:38:45 2023 -0500

agregamos el metodo search
```

```java
public class Node<E> {
    private E data;
    private Node<E> left;
    private Node<E> right;

    public Node(E data, Node<E> left,Node<E> right) {
        this.data = data;
        this.left = left;
        this.right = right;
    }
    public Node(E data) {
        this(data, null,null);
    }
    public E getData() {
        return this.data;
    }

    public void setData(E data) {
        this.data = data;
    }
    public Node<E> getRight() {
        return this.right;
    }
    public Node<E> getLeft() {
        return this.left;
    }
    public void setRight(Node<E> right) {
        this.right = right;
    }
    public void setLeft(Node<E> left) {
        this.left = left;
    }

    public String toString() {
        return this.data.toString();
    }
}
```

Listing 2: Mi segundo commit mas importante es cuando agregue los metodos getmin y get max .

```
commit 6cd58bbc5d4576e099b38a36c71515252c058186
```

```
Author: Misael Marron <mmarronl@unsa.edu.pe>
Date:   Sun Jun 18 18:48:54 2023 -0500

Corregimos los metodos getMin() y getMax()
```

```java
import myExceptions.*;
public class BST<E extends Comparable<E>> {
    private Node<E> root;


    public BST() {
        this.root = null;
    }

    public boolean isEmpty() {
        return this.root == null;
    }

    public E search(E x) throws ExceptionNoFound {
        Node<E> res = searchNode(x, root);

        if(res == null)
            throw new ExceptionNoFound ("El dato "+ x + "no se encuentra");
            return res.getData();
        }

    private Node<E> searchNode(E x, Node<E> n){
        if (n == null)
            return null;
        else {
            int resC = n.getData().compareTo(x);

            if (resC < 0)
                return searchNode(x, n.getRight());
            else if (resC > 0)
                return searchNode(x, n.getLeft());
        else
            return n;
        }
    }

    public E getMin() throws ExceptionIsEmpty{
        E minData = findMin().getData();
        return minData;
    }
    private Node<E> findMin() throws ExceptionIsEmpty{
        Node<E> current = this.root;
```

```
96          if (isEmpty())
97              throw new ExceptionIsEmpty("El ARBOL ESTA VACIO");
98
99          return (getSon(data , root)).getData();
100     }
101     private Node<E> getSon(E data, Node<E> current) {
102         current= searchNode(data,current);
103
104         if (current.getRight() == null)
105             return current.getLeft();
106         else
107             return current.getRight();
108     }
109
110     public void insert(E x) throws ItemDuplicated {
111         this.root = insertNode(x, this.root);
112     }
113
114     private Node<E> insertNode(E x, Node<E> actual) throws ItemDuplicated {
115         Node<E> res = actual;
116         if (actual == null) {res = new Node<E>(x);}
117
118         else {
119             int resC = actual.getData().compareTo(x);
120             if (resC == 0 ) throw new ItemDuplicated(x + "esta duplicado!");
121             if (resC < 0)
122                 res.setRight( insertNode(x, actual.getRight() ));
123             else
124                 res.setLeft(insertNode(x, actual.getLeft()));
125
126         }
127         return res;
128     }
129     public void remove(E x) throws ExceptionNoFound {
130         this.root = removeNode(x, this.root);
131         }
132         protected Node<E> removeNode(E x, Node<E> actual) throws ExceptionNoFound {
133         Node<E> res = actual;
134         if (actual == null)
135             throw new ExceptionNoFound(x + "no esta");
136         int resC = actual.getData().compareTo(x);
137
```

Listing 3: Mi tercer commit es cuando termine y agregue la ejecucion.

```
commit 8ef7ab41083248b14c74b79d6fda92a19352eb34
Author: Misael Marron <mmarronl@unsa.edu.pe>
Date:  Sun Jun 18 21:21:14 2023 -0500

Agregamos las pruebas
```

‘

```java
import myExceptions.*;

public class pruebas {

    public static void main(String[] args) {
        try {
            // Crear un árbol BST
            BST<String> bst = new BST<>();

            // Insertar elementos en el árbol
            bst.insert("C");
            bst.insert("A");
            bst.insert("E");
            bst.insert("B");
            bst.insert("D");

            // Mostrar el árbol en orden
            System.out.println("Árbol en orden:");
            bst.displayInOrder();

            // Buscar un elemento en el árbol
            String searchElement = "B";
            try {
                String result = bst.search(searchElement);
                System.out.println("Elemento encontrado: " + result);
            } catch (ExceptionNoFound exception) {
                System.out.println(exception.getMessage());
            }

            // Obtener el valor mínimo del árbol
            try {
                String minValue = bst.getMin();
                System.out.println("Valor mínimo: " + minValue);
            } catch (ExceptionIsEmpty exception) {
                System.out.println(exception.getMessage());
            }

            // Obtener el valor máximo del árbol
            try {
                String maxValue = bst.getMax();
                System.out.println("Valor máximo: " + maxValue);
            } catch (ExceptionIsEmpty exception) {
```

- Ejecución del lab04:

```java
import myExceptions.*;

public class pruebas {

    public static void main(String[] args) {
        try {
            // Crear un árbol BST
            BST<String> bst = new BST<>();

            // Insertar elementos en el árbol
            bst.insert("H");
            bst.insert("O");
            bst.insert("L");
            bst.insert("A");
            bst.insert("G");

            // Mostrar el árbol en orden
            System.out.println("Arbol en orden:");
            bst.displayInOrder();

            // Buscar un elemento en el árbol
            String searchElement = "B";
```

Problems Javadoc Declaration Console Coverage Debug

<terminated> Pruebas [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (18 jun. 2023 21:25:19 – 21:25:19) [

```
Arbol en orden:
A G H L O
El dato Bno se encuentra
Valor minimo: A
Valor maximo: O
El dato no se encuentra
Hijo de A: G
Eno esta
Arbol en orden despuEs de eliminar E:
A G H L O
```

4.2. Estructura de laboratorio 04

- El contenido que se entrega en este laboratorio es el siguiente:

```
lab03/
|--- codigo
|   |--- Node.java
|   |--- pruebas.java
|   |--- BST.java
|--- latex
    |--- img
    |   |--- logo_abet.png
    |   |--- logo_episunsa.png
    |   |--- logo_unsa.jpg
    |   |--- codigo1.jpg
    |   |--- codigo2.jpg
    |   |--- codigo2b.jpg
    |   |--- codigo3.jpg
    |   |--- ejecucion.jpg
    |--- Lab04-MisaelMarron.pdf
```

```
|--- Lab04-MisaelMarron.tex
```

5. Preguntas:

- ¿Explique como es el algoritmo que implemento para obtener el arbol binario de busqueda con la libreria Graph Stream? Recuerde que pueden haber operaciones sobre el BST.
- El algoritmo para mostrar un árbol binario de búsqueda con Graph Stream consta de los siguientes pasos:

  Crear un objeto Graph para representar el grafo. Definir las características visuales del grafo. Agregar nodos y aristas al grafo recursivamente. Mostrar el grafo en una ventana utilizando el visor de Graph Stream.

6. Rúbricas

6.1. Entregable Informe

Tabla 1: Tipo de Informe

Informe	
Latex	El informe está en formato PDF desde Latex, con un formato limpio (buena presentación) y facil de leer.

6.2. Rúbrica para el contenido del Informe y demostración

- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumplio con el ítem correspondiente.
- Si un alumno supera la fecha de entrega, su calificación será sobre la nota mínima aprobada, siempre y cuando cumpla con todos lo items.
- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la siguiente tabla:

Tabla 2: Niveles de desempeño

	Nivel			
Puntos	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

Tabla 3: Rúbrica para contenido del Informe y demostración

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	X	2	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	3	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X	2	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	X	1.5	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	X	1	
6. Fechas	Las fechas de modificación del código fuente estan dentro de los plazos de fecha de entrega establecidos.	2	X	2	
7. Ortografía	El documento no muestra errores ortográficos.	2	X	1.5	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4	X	4	
Total		20		17	

7. Referencias

- https://docs.oracle.com/javase/tutorial/java/generics/types.html
- https://www.eclipse.org/downloads/packages/release/2022-03/r/eclipse-ide-enterprise-java-and-web
- https://www.w3schools.com/java/default.asp