

Informe de Laboratorio 06

Tema: Tries

Nota

Estudiante	CUI	Correo
Condorios Yllapuma Jorge	20222076	jcondorios@unsa.edu.pe
Cusilayme García José Luis	20220598	jcusilaymeg@unsa.edu.pe
Mamani Mamani Alexis	20222066	almamanima@unsa.edu.pe
Valdivia Luna Carlo Joaquín	20220567	cvaldivialu@unsa.edu.pe

Facultad	Asignatura	Docente
Escuela Profesional de Ingeniería de Sistemas	Estructura de Datos Semestre: III	Richart Smith Escobedo Quispe rescobedoq@unsa.edu.pe

Laboratorio	Tema	Duración
06	Tries	02 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2023 - A	Del 19 Julio 2023	Al 23 Julio 2023

1. Tarea

- Elabore un informe paso a paso de la implementación una Trie para insertar, buscar y reemplazar palabras en un texto.
- Utilice interfaces gráficas de usuario para su implementación.
- Utilice todas las recomendaciones dadas por el docente.
- Utilice la siguiente GUI como referencia:

2. Equipos, materiales y temas utilizados

- Sistema Operativo Windows 10 Home 64 bits (10.0, compilation 19045)
- VIM 9.0.
- OpenJDK 64-Bits 17.0.7.
- Git 2.39.2.
- Cuenta en GitHub con el correo institucional.
- Java

3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- https://github.com/JorgeCY21/EDA_LAB_D
- URL para el laboratorio 06 en el Repositorio GitHub.
- https://github.com/JorgeCY21/EDA_LAB_D/tree/main/Lab_06

4. Actividades con el repositorio GitHub

4.1. Creando e inicializando repositorio GitHub

- Se realizaron los siguientes comandos en la computadora:

Listing 1: Dirigiéndonos al directorio de trabajo

```
D:\
```

Listing 2: Clonando repositorio GitHub

```
D:\ git clone https://github.com/JorgeCY21/EDA_LAB_D
```

Listing 3: Inicializando directorio para laboratorio 06

```
D:\ cd EDA_LAB_D  
D:\ EDA_LAB_D> mkdir lab06  
D:\ EDA_LAB_D> cd lab06
```

4.2. Commits

Listing 4: Commit: Creamos el trie principal

```
D:\ EDA_LAB_D\Lab_06> git add .  
D:\ EDA_LAB_D\Lab_06> git commit -m "Creando entorno principal de trie"  
D:\ EDA_LAB_D\Lab_06> git push -u origin main
```

5. Códigos Fuente

TrieNode:

Listing 5: TrieNode

```
class TrieNode {
    private TrieNode[] children;
    private boolean isEndWord;

    public TrieNode() {
        this.children = new TrieNode[255];
        this.isEndWord = false;
    }

    public boolean isEndWord() {
        return this.isEndWord;
    }

    public void setIsEndWord(boolean isEndWord) {
        this.isEndWord = isEndWord;
    }

    public TrieNode getChild(char c) {
        return this.children[c];
    }

    public void setChild(char c, TrieNode node) {
        this.children[c] = node;
    }

    public int getChildrenLength() {
        return this.children.length;
    }
}
```

En el primer fragmento de código se representa la clase TrieNode, el cual contiene un arreglo de 255 elementos(estos representan el código ACCII). Observamos dos atributos. El primero nos ayudará para conocer a al nodo siguiente, es decir, ayudara a la conexión con otros nodos para representar palabras completas o prefijos. Y el segundo atributo nos ayuda a saber si es el final de la palabra.

Trie:

Listing 6: Trie-insertar

```
class Trie {
    private TrieNode root;

    public Trie() {
        this.root = new TrieNode();
    }

    public Trie(String text) {
        this.root = new TrieNode();

        String[] palabras = text.split("\\s+");
        for (String palabra : palabras) {
            if (!palabra.isEmpty())
                insert(palabra);
        }
    }

    public void insert(String word) {
        TrieNode current = root;
        for (char c : word.toCharArray()) {
            if (current.getChild(c) == null) {
                current.setChild(c, new TrieNode());
            }
            current = current.getChild(c);
        }
        current.setIsEndWord(true);
    }
}
```

En la primera sección del código Trie observamos Nodo raíz de nuestro Trie, después dos constructores, el segundo constructor pide como parámetro texto el cual primero al texto lo dividiremos en palabras, esto haremos mediante la expresión regular "\\s+" para después con un foreach insertar las palabras en nuestro trie

El método para insertar una palabra en el Trie. Comienza desde el nodo raíz y recorre cada carácter de la palabra. Si el carácter no tiene un nodo hijo, se crea un nuevo nodo. Luego, se mueve al siguiente nodo hijo correspondiente al siguiente carácter. Al final de la palabra, marca el último nodo como un nodo de fin de palabra *setIsEndWord(true)* para indicar que se ha insertado una palabra completa.

Listing 7: Trie-eliminación

```
public void delete(String word) {
    TrieNode current = root;
    TrieNode[] parents = new TrieNode[word.length()];
    char[] chars = word.toCharArray();

    for (int i = 0; i < chars.length; i++) {
        parents[i] = current;
        current = current.getChild(chars[i]);
        if (current == null) {
            return;
        }
    }

    if (current.isEndWord()) {
        current.setIsEndWord(false);
    }

    for (int i = chars.length - 1; i >= 0; i--) {
        if (current.getChildrenLength() > 0 || current.isEndWord()) {
            break;
        }
        parents[i].setChild(chars[i], null);
        current = parents[i];
    }
}
```

El método recorre el Trie buscando el camino hacia la palabra a eliminar, almacenando los nodos visitados en el camino en un arreglo llamado `parents`. Si la palabra no existe en el Trie (es decir, no se encuentra el último nodo correspondiente al último carácter de la palabra), el método simplemente retorna sin hacer cambios. Si se encuentra la palabra, se marca el último nodo como no final de palabra, indicando que la palabra ha sido eliminada. A continuación, el método revisa los nodos en el camino hacia la palabra eliminada. Si un nodo no tiene hijos y no representa el final de otra palabra, se elimina del Trie.

Listing 8: Trie-reemplazar, buscar y startswith

```
public void replace(String palabraVieja, String palabraNueva) {
    if (search(palabraVieja)) {
        delete(palabraVieja);
        insert(palabraNueva);
    }
}

public boolean search(String word) {
    TrieNode current = root;
    for (char c : word.toCharArray()) {
        if (current.getChild(c) == null) {
            return false;
        }
        current = current.getChild(c);
    }
    return current.isEndWord();
}
```

```
public boolean startsWith(String prefix) {
    TrieNode current = root;
    for (char c : prefix.toCharArray()) {
        current = current.getChild(c);
        if (current == null) {
            return false;
        }
    }
    return true;
}
```

El método `replace` como su nombre lo dice reemplaza una palabra vieja en el Trie con una palabra nueva. Busca la palabra vieja, si la encuentra la elimina con `delete`, luego inserta la palabra nueva con `insert`. Este método solo se logra aplicando los otros métodos. El método `search` comienza desde el nodo raíz y recorre cada carácter de la palabra (word) buscando el camino en el Trie. Si en algún punto no encuentra un nodo hijo correspondiente a un carácter, se detiene y devuelve `false`, lo que indica que la palabra no está presente en el Trie. Si encuentra todos los caracteres y llega a un nodo de fin de palabra, devuelve `true`, lo que indica que la palabra está presente en el Trie. El método `startswith` se utiliza para verificar si alguna palabra en el Trie comienza con un cierto prefijo dado (prefix). Comienza desde el nodo raíz y recorre cada carácter del prefijo, buscando el camino en el Trie. Si en algún punto no encuentra un nodo hijo correspondiente a un carácter, se detiene y devuelve `false`, lo que indica que no hay palabras en el Trie que comiencen con ese prefijo.

Listing 9: Trie-reemplazar, buscar y `startswith`

```
public boolean isEmpty() {
    for (int i = 0; i < this.root.getChildrenLength(); i++) {
        if (root.getChild((char) i) != null) {
            return false;
        }
    }

    return true;
}

@Override
public String toString() {
    StringBuilder result = new StringBuilder();
    collectWords(root, new StringBuilder(), result);
    return result.toString();
}

private void collectWords(TrieNode node, StringBuilder currentWord, StringBuilder result) {
    if (node == null) {
        return;
    }

    if (node.isEndWord()) {
        result.append(currentWord).append(" ");
    }

    for (char c = 0; c < node.getChildrenLength(); c++) {
        TrieNode child = node.getChild(c);
        if (child != null) {
            currentWord.append(c);
            collectWords(child, currentWord, result);
        }
    }
}
```

```
        currentWord.deleteCharAt(currentWord.length() - 1);
    }
}
}
```

Ya solo nos queda el método para verificar que este vacío el trie, el cual solo recorre todas las referencias hasta llegar a null. Y el método toString que nos ayudara a obtener una representación de todas las palabras del trie, esto gracias a setIsEndWord para saber que palabras deberían ir o no. **GUI: Todo el código realizado para la interfaz gráfica para el usuario**

Listing 10: Trie-eliminación

```
import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Cursor;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.Timer;
import javax.swing.border.EmptyBorder;
import javax.swing.border.LineBorder;
import javax.swing.event.DocumentEvent;
import javax.swing.event.DocumentListener;

public class GUI extends JFrame {

    private final int WIDTH = 450;
    private final int HEIGHT = 250;

    private final int WIDTH_CONTENT_PANEL = WIDTH - 40;
    private final int HEIGHT_CONTENT_PANEL = 133;
    private static final int DELAY = 1000;

    private JPanel bodyPanel;
    private Trie trie;

    private PageBuscar pageBuscar = new PageBuscar(this, trie);
    private PageReemplazar pageReemplazar = new PageReemplazar(this, trie);
    private PageIrA pageIrA = new PageIrA(this, trie);

    private JButton btnBuscar;
    private JButton btnReemplazar;
    private JButton btnIrA;

    private JTextArea textArea;
    private Timer timer;

    private JPanel contentPanel;
```

```
private boolean estaExpandido = false;

public static void main(String[] args) {
    new GUI();
}

public GUI() {
    setResizable(false);
    setSize(WIDTH, HEIGHT);
    setTitle("Editor de Texto");
    setLocationRelativeTo(null);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    bodyPanel = new JPanel();
    bodyPanel.setBorder(new EmptyBorder(5, 5, 5, 5));

    setContentPane(bodyPanel);
    bodyPanel.setLayout(null);

    content();

    setVisible(true);
}

private void content() {
    JLabel label = new JLabel("Texto:");
    label.setFont(new Font("Roboto Black", Font.PLAIN, 12));
    label.setBounds(10, 11, 83, 23);
    bodyPanel.add(label);

    btnBuscar = new JButton("Buscar", 10, 170, 70, 23, null, null);
    btnBuscar.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            showPanel(pageBuscar);
        }
    });
    bodyPanel.add(btnBuscar);

    btnReemplazar = new JButton("Reemplazar", 85, 170, 95, 23, null, null);
    btnReemplazar.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            showPanel(pageReemplazar);
        }
    });
    bodyPanel.add(btnReemplazar);

    btnIrA = new JButton("Ir a", 185, 170, 45, 23, null, null);
    btnIrA.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            showPanel(pageIrA);
        }
    });
    bodyPanel.add(btnIrA);

    contentPanel = new JPanel();
    contentPanel.setBounds(10, 200, WIDTH_CONTENT_PANEL, HEIGHT_CONTENT_PANEL);
```



```
bodyPanel.add(contentPanel);

textArea = new JTextArea(5, 20);
textArea.setBackground(new Color(255, 255, 255));
textArea.setLineWrap(true);
textArea.setWrapStyleWord(true);

textArea.getDocument().addDocumentListener(new DocumentListener() {
    @Override
    public void insertUpdate(DocumentEvent e) {
        restartTimer();
    }

    @Override
    public void removeUpdate(DocumentEvent e) {
        restartTimer();
    }

    @Override
    public void changedUpdate(DocumentEvent e) {
    }
});

timer = new Timer(Delay, new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        timer.stop();
        updateTrie();
    }
});

JScrollPane scrollPane = new JScrollPane(textArea);
scrollPane.setBounds(20, 45, WIDTH_CONTENT_PANEL - 10, 100);
bodyPanel.add(scrollPane);
}

private void restartTimer() {
    timer.restart();
}

private void updateTrie() {
    String text = textArea.getText();
    trie = new Trie(text);
    System.out.println(trie);
}

private void showPanel(JPanel panel) {
    if (!estaExpandido) {
        setSize(WIDTH, HEIGHT + 130);
        setLocationRelativeTo(null);
    }

    panel.setSize(WIDTH_CONTENT_PANEL, HEIGHT_CONTENT_PANEL);
    panel.setLocation(0, 0);

    contentPanel.removeAll();
```

```
        contentPanel.setLayout(new BorderLayout());
        contentPanel.add(panel);
        contentPanel.revalidate();
        contentPanel.repaint();
    }

    private JButton newJButton(String text, int x, int y, int width, int height, Color border,
        Color background) {
        JButton btn = new JButton(text);

        btn.setFocusPainted(false);
        btn.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
        btn.setBorder(new LineBorder((border != null) ? border : new Color(0, 0, 0)));
        btn.setBackground((background != null) ? background : new Color(215, 215, 215));
        btn.setBounds(x, y, width, height);

        return btn;
    }

    public void ocultar() {
        setSize(WIDTH, HEIGHT);
        setLocationRelativeTo(null);
        estaExpandido = false;
        contentPanel.removeAll();
    }

    public JTextArea getTextArea() {
        return textArea;
    }
}
```

Listing 11: Trie-eliminación

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.JButton;
import javax.swing.JOptionPane;
import javax.swing.JPanel;

public class Page extends JPanel {
    protected Trie trie;
    protected GUI gui;

    public Page(GUI gui, Trie trie) {
        this.gui = gui;
        this.trie = trie;

        setSize(410, 133);
        setLayout(null);

        JButton btnCancelar = new JButton("Cancelar");
        btnCancelar.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                gui.ocultar();
            }
        });
    }
}
```

```
});  
btnCancelar.setBounds(300, 100, 89, 23);  
add(btnCancelar);  
}  
  
public void mensajeError(String alert) {  
    JOptionPane.showMessageDialog(this.gui, alert, "MENSAJE DE ERROR",  
        JOptionPane.ERROR_MESSAGE);  
}  
  
public void mensaje(String alert) {  
    JOptionPane.showMessageDialog(this.gui, alert);  
}  
  
public void actualizarTrie(String text) {  
    this.trie = new Trie(text);  
}  
}
```

Listing 12: Trie-eliminación

```
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
  
import javax.swing.JButton;  
import javax.swing.JLabel;  
import javax.swing.JTextField;  
  
public class PageBuscar extends Page {  
    private JTextField buscarField;  
  
    public PageBuscar(GUI gui, Trie trie) {  
        super(gui, trie);  
        contenido();  
    }  
  
    public void contenido() {  
        JLabel lblBuscar = new JLabel("Buscar:");  
        lblBuscar.setBounds(10, 22, 100, 14);  
        add(lblBuscar);  
  
        buscarField = new JTextField();  
        buscarField.setBounds(61, 19, 247, 20);  
        add(buscarField);  
        buscarField.setColumns(10);  
  
        JButton btnBuscar = new JButton("Buscar");  
        btnBuscar.addActionListener(new ActionListener() {  
            public void actionPerformed(ActionEvent e) {  
                actualizarTrie(gui.getTextArea().getText());  
  
                if (gui.getTextArea().getText().isEmpty()) {  
                    mensajeError("Ingrese un texto");  
                } else if (buscarField.getText().equals("")) {  
                    mensajeError("Ingrese una palabra a buscar");  
                }  
            }  
        });  
    }  
}
```

```
        } else {
            String palabraBuscar = buscarField.getText().trim();
            if (trie.search(palabraBuscar)) {
                mensaje("La palabra buscada se encuentra en el texto");
            } else {
                mensaje("La palabra buscada no se encuentra en el texto");
            }
        }
    }
}
});
btnBuscar.setBounds(10, 100, 89, 23);
add(btnBuscar);
}
}
```

Listing 13: Trie-eliminación

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.JButton;
import javax.swing.JLabel;
import javax.swing.JTextField;
import javax.swing.text.DefaultCaret;

public class PageIrA extends Page {

    private JTextField irAField;

    public PageIrA(GUI gui, Trie trie) {
        super(gui, trie);
        contenido();
    }

    public void contenido() {
        JLabel lblIrA = new JLabel("Ir a:");
        lblIrA.setBounds(20, 25, 46, 14);
        add(lblIrA);

        irAField = new JTextField();
        irAField.setBounds(53, 22, 158, 20);
        add(irAField);
        irAField.setColumns(10);

        JButton btnIrA = new JButton("Ir a");
        btnIrA.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                String palabraBuscada = irAField.getText();
                String texto = gui.getTextArea().getText();

                if (!trie.search(palabraBuscada)) {
                    mensajeError("Palabra no encontrada");
                    return;
                }

                int posicion = texto.indexOf(palabraBuscada);
```

```
        if (posicion >= 0) {
            DefaultCaret caret = (DefaultCaret) gui.getTextArea().getCaret();

            caret.setDot(posicion);

            gui.getTextArea().requestFocusInWindow();
        }
    }
});
btnIrA.setBounds(10, 100, 89, 23);
add(btnIrA);
}
}
```

Listing 14: Trie-eliminación

```
import javax.swing.JLabel;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;

public class PageReemplazar extends Page {

    private JTextField buscarField;
    private JTextField reemplazarField;
    private JButton btnReemplazar;

    public PageReemplazar(GUI gui, Trie trie) {
        super(gui, trie);
        contenido();
    }

    public void contenido() {
        JLabel lblBuscar = new JLabel("Buscar:");
        lblBuscar.setBounds(10, 22, 100, 14);
        add(lblBuscar);

        buscarField = new JTextField();
        buscarField.setBounds(61, 19, 247, 20);
        add(buscarField);
        buscarField.setColumns(10);

        JLabel lblReemplazar = new JLabel("Reemplazar con:");
        lblReemplazar.setBounds(10, 60, 100, 14);
        add(lblReemplazar);

        reemplazarField = new JTextField();
        reemplazarField.setBounds(115, 57, 193, 20);
        add(reemplazarField);
        reemplazarField.setColumns(10);

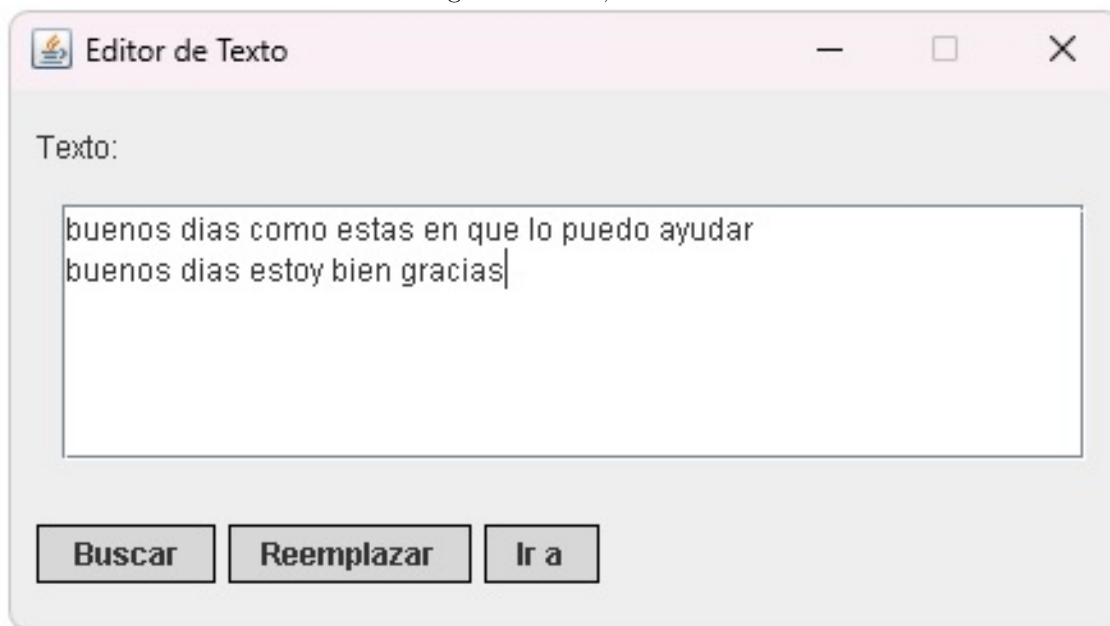
        btnReemplazar = new JButton("Reemplazar");
        btnReemplazar.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
```

```
        actualizarTrie(gui.getTextArea().getText());

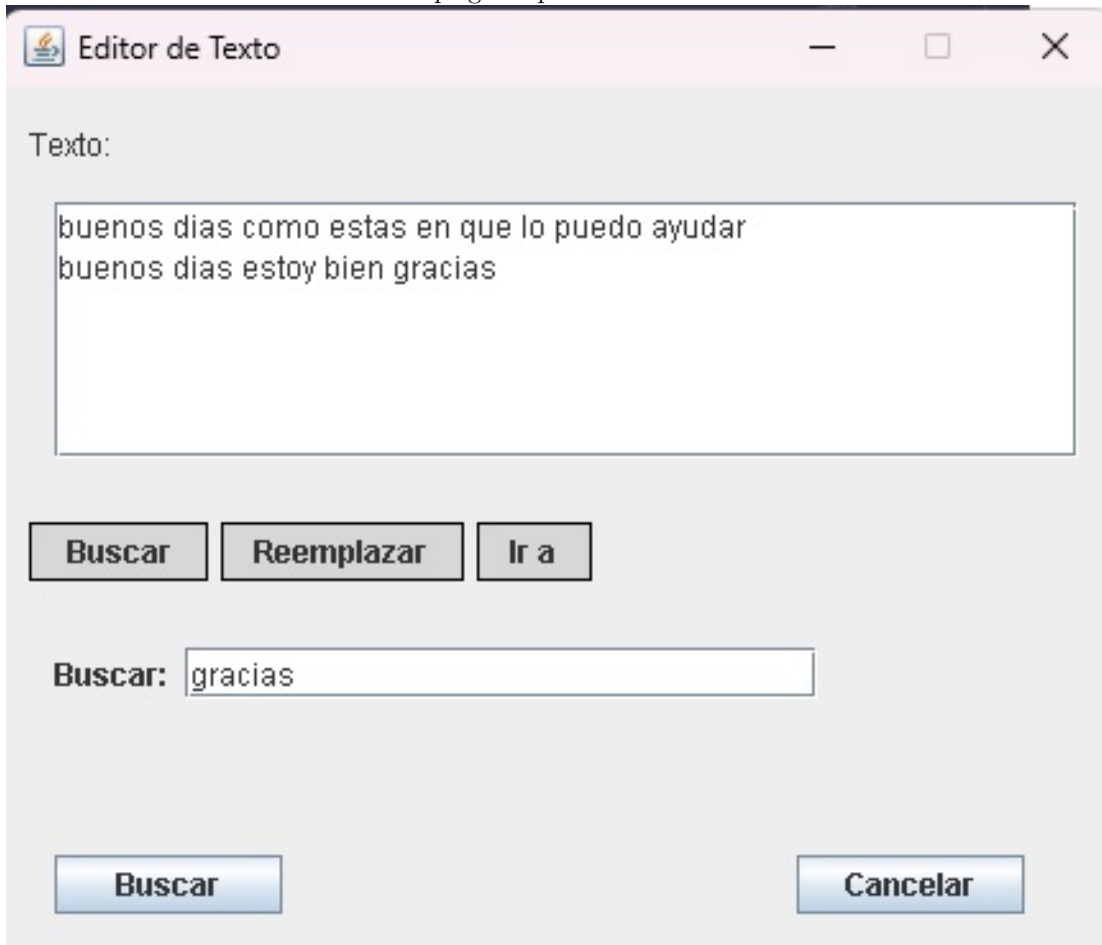
        if (gui.getTextArea().getText().isEmpty()) {
            mensajeError("Ingrese un texto");
        } else if (buscarField.getText().equals("")) {
            mensajeError("Ingrese una palabra a buscar");
        } else if (reemplazarField.getText().equals("")) {
            mensajeError("Ingrese una palabra a reemplazar");
        } else {
            String palabraBuscar = buscarField.getText().trim();
            String palabraReemplazar = reemplazarField.getText().trim();
            if (trie.search(palabraBuscar)) {
                trie.replace(palabraBuscar, palabraReemplazar);
                String texto = gui.getTextArea().getText().replaceAll(palabraBuscar,
                    palabraReemplazar);
                gui.getTextArea().setText(texto);
                buscarField.setText(palabraReemplazar);
                reemplazarField.setText("");
            } else {
                mensaje("La palabra buscada no se encuentra en el texto");
            }
        }
    }
});
btnReemplazar.setBounds(10, 100, 120, 23);
add(btnReemplazar);
}
}
```


5.1. Ejecución y Pruebas

Interfaz gráfica inicial, insertando texto.

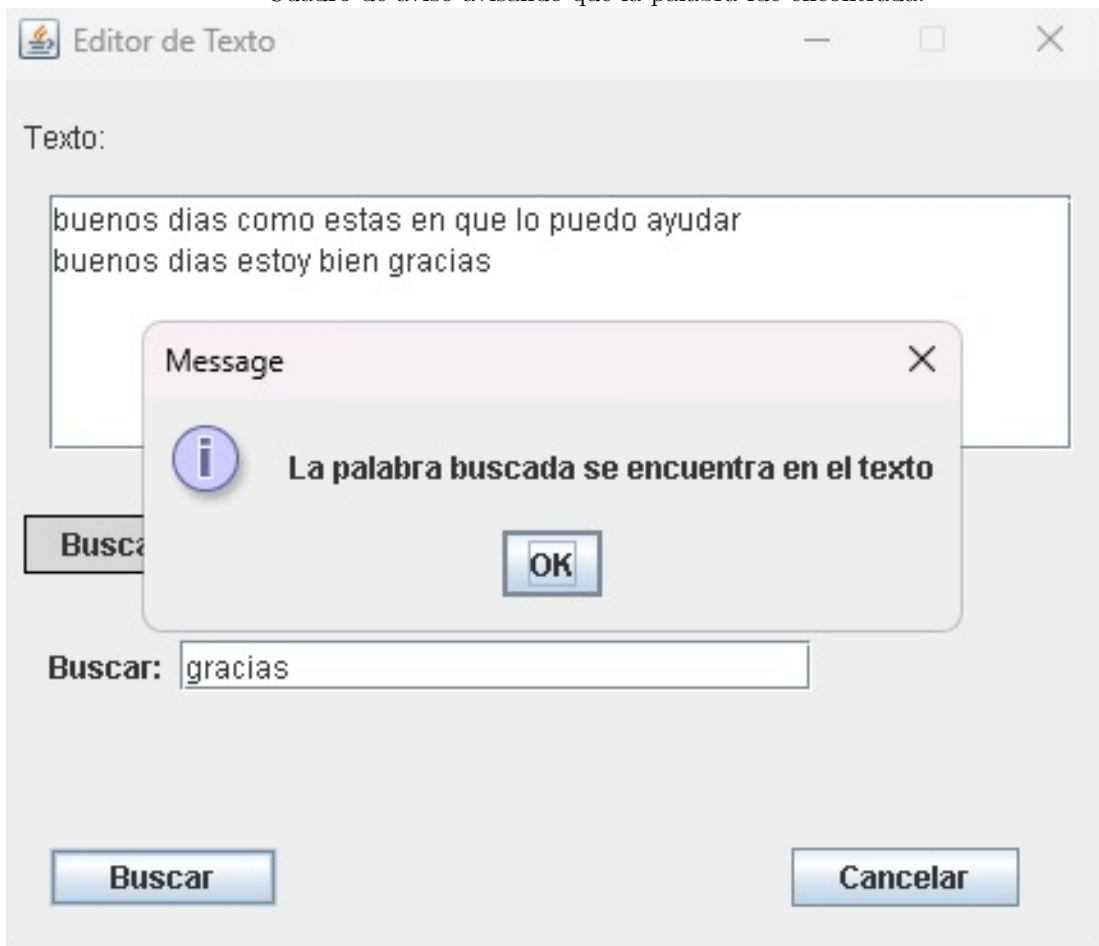


Ventana desplegable presionando el botón "buscar".

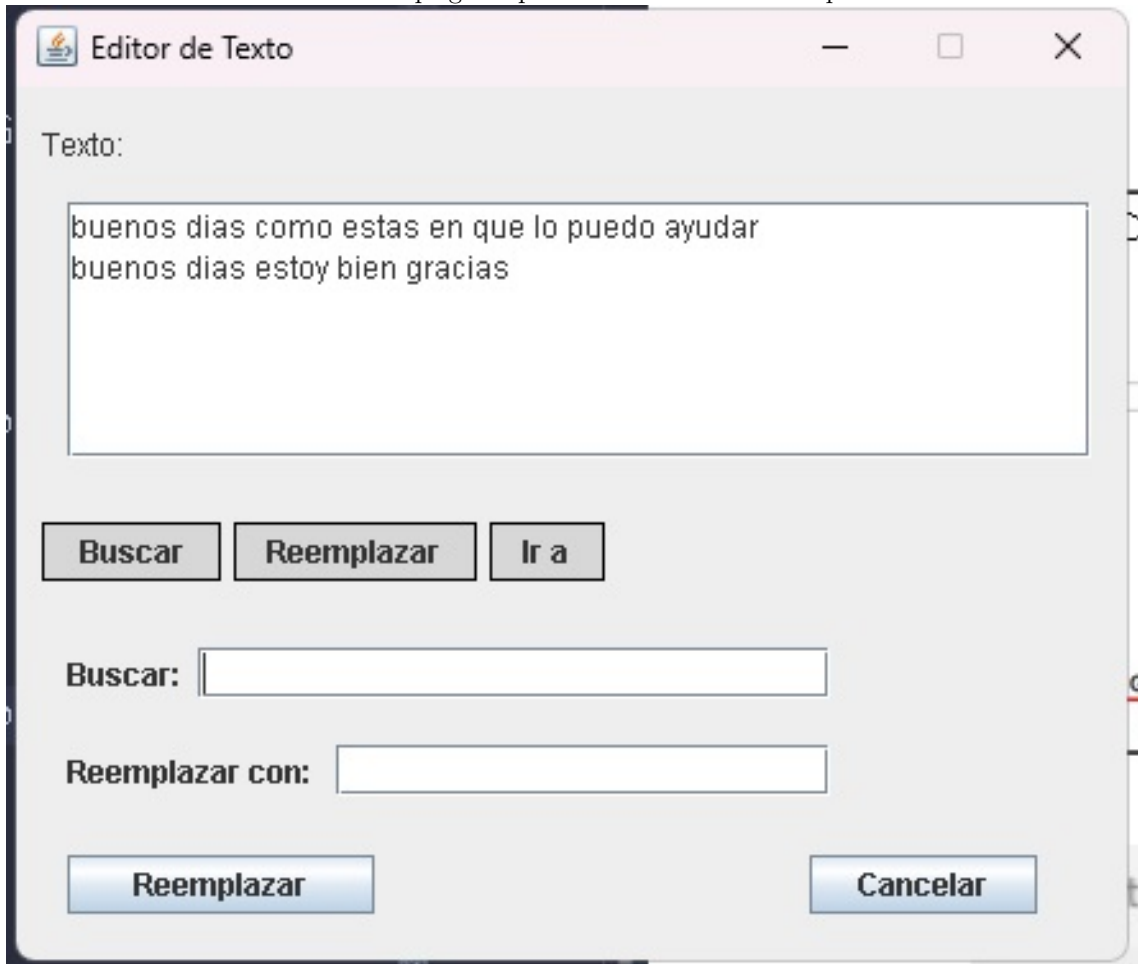


The image shows a screenshot of a software window titled "Editor de Texto". The window has a standard Windows-style title bar with minimize, maximize, and close buttons. Inside the window, there is a text area labeled "Texto:" containing the text:
buenos dias como estas en que lo puedo ayudar
buenos dias estoy bien gracias
Below the text area, there are three buttons: "Buscar", "Reemplazar", and "Ir a". Under the "Buscar" button, there is a label "Buscar:" followed by a text input field containing the word "gracias". At the bottom of the window, there are two buttons: "Buscar" and "Cancelar".

Cuadro de aviso avisando que la palabra fue encontrada.



Ventana desplegable presionando el botón reemplazar”.



Editor de Texto

Texto:

buenos dias como estas en que lo puedo ayudar
buenos dias estoy bien gracias

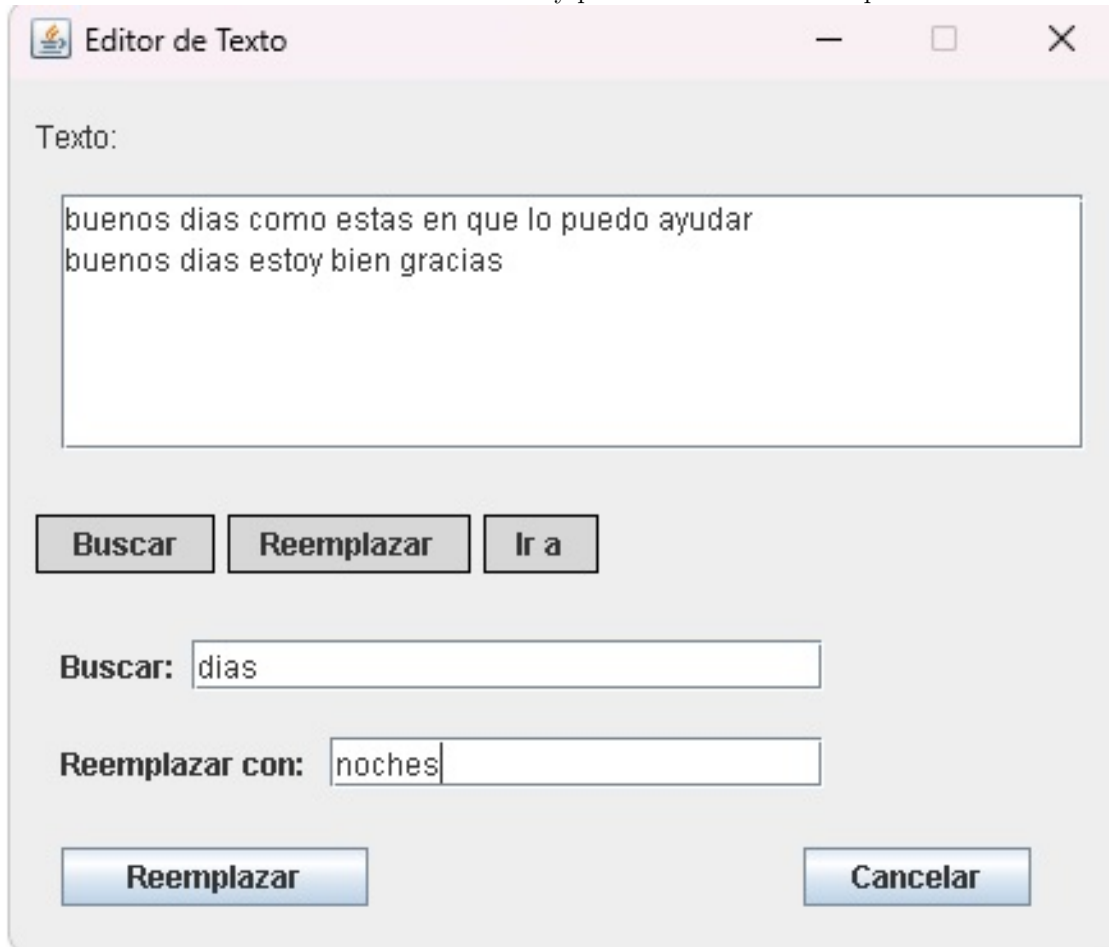
Buscar **Reemplazar** **Ir a**

Buscar:

Reemplazar con:

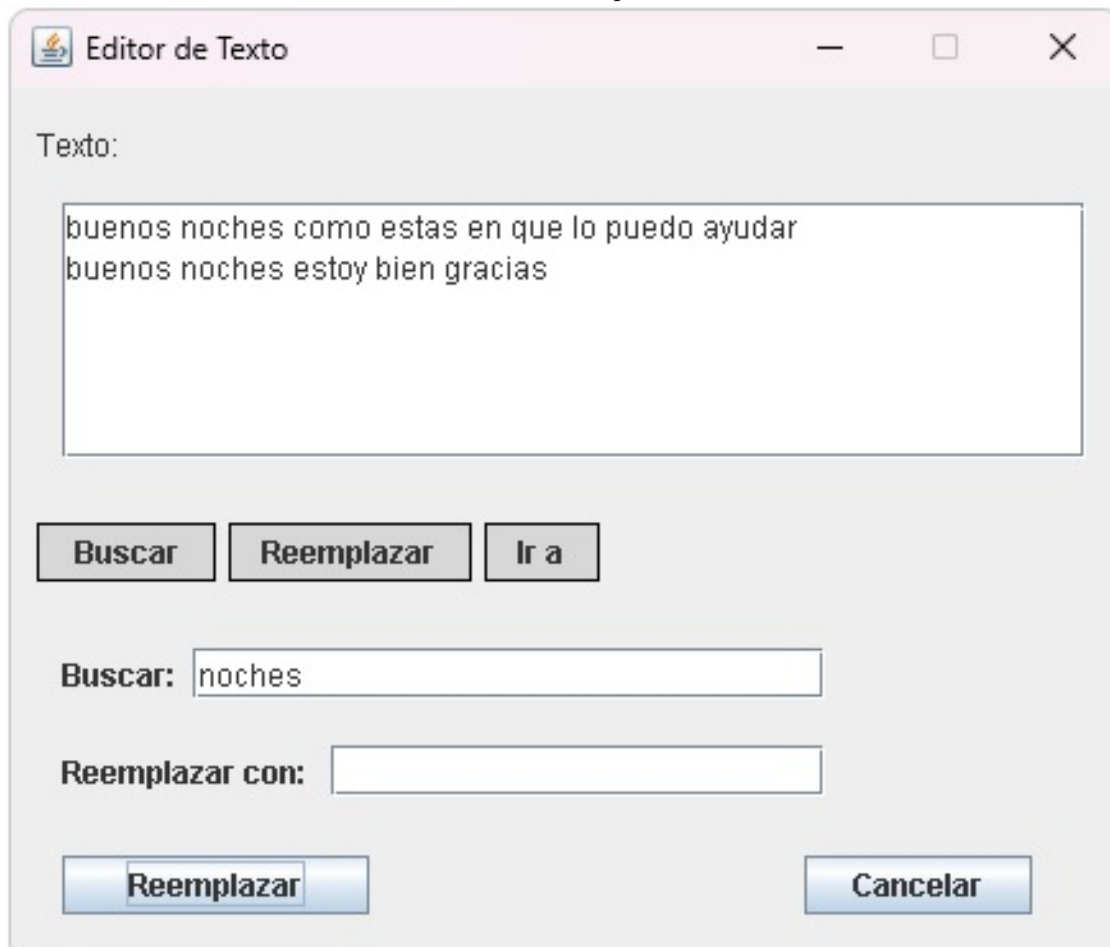
Reemplazar **Cancelar**

Insertando texto a buscar y palabra con la cual reemplazar.



The image shows a window titled "Editor de Texto" with a standard Windows-style title bar (minimize, maximize, close buttons). Inside the window, there is a label "Texto:" followed by a large text area containing two lines of text: "buenos dias como estas en que lo puedo ayudar" and "buenos dias estoy bien gracias". Below the text area, there are three buttons: "Buscar", "Reemplazar", and "Ir a". Further down, there are two input fields. The first is labeled "Buscar:" and contains the text "dias". The second is labeled "Reemplazar con:" and contains the text "noches". At the bottom of the window, there are two buttons: "Reemplazar" and "Cancelar".

Texto reemplazado.



The image shows a screenshot of a text editor window titled "Editor de Texto". The window has a standard Windows-style title bar with minimize, maximize, and close buttons. Inside the window, there is a text area containing the following text:

Textos:
buenos noches como estas en que lo puedo ayudar
buenos noches estoy bien gracias

Below the text area, there are three buttons: "Buscar", "Reemplazar", and "Ir a". Below these buttons, there are two input fields. The first is labeled "Buscar:" and contains the text "noches". The second is labeled "Reemplazar con:" and is empty. At the bottom of the window, there are two buttons: "Reemplazar" and "Cancelar".

6. Cuestionario

6.1. Pregunta: Explique. ¿Cómo se utiliza esta estructura de datos para almacenar prefijos?.

- Para almacenar prefijos, una estructura de datos comúnmente utilizada es el árbol Trie, Trie es una estructura de datos jerárquica que se organiza como un árbol en el que cada nodo representa un carácter individual de las cadenas. Los nodos en el árbol se organizan de manera que los prefijos comunes comparten los mismos nodos en la estructura, lo que lo hace muy útil para almacenar y buscar conjuntos grandes de palabras con prefijos comunes. Para almacenar prefijos usamos: inserción de una palabra, búsqueda de prefijos y eliminación.

El Trie es especialmente útil para problemas relacionados con la búsqueda y recuperación de palabras y prefijos, como la implementación de autocompletado en un motor de búsqueda o el procesamiento de diccionarios. Sin embargo, debe tenerse en cuenta que el consumo de memoria puede ser una preocupación si el Trie es extremadamente grande, ya que puede requerir mucho espacio para almacenar todas las combinaciones de caracteres posibles.

6.2. Pregunta: ¿Cómo realizó la funcionalidad de reemplazar texto?

- Realizamos la funcionalidad aplicando los métodos que anteriormente hemos implementado, primero con `search()` se busco las coincidencias, después con `delete()` se eliminaron y se insertaron en su posición con `insert` la nueva palabra.

7. Rúbricas

7.1. Entregable Informe

Tabla 1: Tipo de Informe

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

7.2. Rúbrica para el contenido del Informe y demostración

- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumple 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 los ítems.
- 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. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X		
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X		
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	X		
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	X		
6. Fechas	Las fechas de modificación del código fuente están dentro de los plazos de fecha de entrega establecidos.	2	X		
7. Ortografía	El documento no muestra errores ortográficos.	2	X		
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			
Total		20			

8. Referencias

- Presentación TRIE hecha en clase por la Dra. Karim Guevara Puente de la Vega
- https://www.youtube.com/watch?v=m9zawMC6QAI&ab_channel=ApnaCollege
- <https://www.educba.com/trie-data-structure-in-java/>