```
package application;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Scene;
import javafx.scene.layout.AnchorPane;
import javafx.stage.Stage;
import utilities.Paths;

public class App extends Application {

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

   @Override
   public void start(Stage stage) throws Exception {

      AnchorPane load = FXMLLoader.load(getClass().getResource(Paths.inicio)); //
aqui adentro esta la vista cargada
      Scene scene = new Scene(load); // escena donde entra todos
      stage.setScene(scene);

      stage.show();
   }
}
```

```
package controller;

import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.fxml.FXML;
import javafx.scene.control.*;
import javafx.scene.control.cell.PropertyValueFactory;
import utilities.SlotClave;

public class BusquedaLinealController {
```

```
@FXML private TextField nField;
@FXML private ChoiceBox<Integer> digitosChoice;

@FXML private TableView<SlotClave> tabla;
@FXML private TableColumn<SlotClave, Integer> colPos;
@FXML private TableColumn<SlotClave, String> colClave;

@FXML private TextField claveInsertField;
@FXML private TextField claveBuscarField;
@FXML private Label resultadoLabel;

private final ObservableList<SlotClave> data = FXCollections.observableArrayList();
private int digitos = 2; // por defecto
private boolean creada = false;

@FXML
public void initialize() {
  // llenar choice de dígitos (ajústalo si quieres más)
  digitosChoice.getItems().addAll(1, 2, 3, 4);
  digitosChoice.setValue(2);

  colPos.setCellValueFactory(new PropertyValueFactory<>("posicion"));
  colClave.setCellValueFactory(new PropertyValueFactory<>("clave"));

  tabla.setItems(data);
}


@FXML
private void crearEstructura() {
  Integer n = leerEntero(nField);
  if (n == null || n < 1) {
    resultadoLabel.setText("N debe ser un número >= 1.");
    return;
  }

  digitos = digitosChoice.getValue() != null ? digitosChoice.getValue() : 2;
```

```
      data.clear();
      for (int i = 0; i < n; i++) {
        data.add(new SlotClave(i, "")); // clave vacía
      }

      creada = true;
      resultadoLabel.setText("Estructura creada con N=" + n + " y claves de " + digitos + "
dígitos.");
    }

    @FXML
    private void insertarClave() {
      if (!creada) {
        resultadoLabel.setText("Primero debes crear la estructura.");
        return;
      }

      String claveTxt = normalizarClave(claveInsertField.getText(), digitos);
      claveInsertField.setText(claveTxt); // para que el usuario vea el 0 agregado

      if (!claveValidaPorDigitos(claveTxt, digitos)) {
        resultadoLabel.setText("La clave debe tener exactamente " + digitos + " dígitos
(solo números).");
        return;
      }



      // Evitar repetidos (opcional, pero recomendado)
      for (SlotClave s : data) {
        if (claveTxt.equals(s.getClave())) {
          resultadoLabel.setText("Esa clave ya existe en la estructura.");
          return;
        }
      }

      // Inserción simple: primera posición libre
      for (SlotClave s : data) {
        if (s.getClave() == null || s.getClave().isBlank()) {
```

```
            s.setClave(claveTxt);
            tabla.refresh();
            resultadoLabel.setText("Clave " + claveTxt + " insertada en posición " +
s.getPosicion() + ".");
            return;
          }
        }

        resultadoLabel.setText("No hay espacio: la estructura está llena.");
    }

    @FXML
    private void buscarClave() {
        if (!creada) {
            resultadoLabel.setText("Primero debes crear la estructura.");
            return;
        }

        String claveTxt = normalizarClave(claveInsertField.getText(), digitos);
        claveInsertField.setText(claveTxt); // para que el usuario vea el 0 agregado

        if (!claveValidaPorDigitos(claveTxt, digitos)) {
            resultadoLabel.setText("La clave debe tener exactamente " + digitos + " dígitos
(solo números).");
            return;
        }


        int comparaciones = 0;
        long inicio = System.nanoTime();

        for (SlotClave s : data) {
            comparaciones++;
            if (claveTxt.equals(s.getClave())) {
                long fin = System.nanoTime();
                tabla.getSelectionModel().select(s);
                tabla.scrollTo(s);
                resultadoLabel.setText("Encontrada en posición " + s.getPosicion()
```

```
                + " | Comparaciones: " + comparaciones
                + " | Tiempo: " + (fin - inicio) + " ns");
            return;
        }
    }

    long fin = System.nanoTime();
    resultadoLabel.setText("No encontrada | Comparaciones: " + comparaciones + " |
Tiempo: " + (fin - inicio) + " ns");
  }

  @FXML
  private void ordenarClaves() {
    if (!creada) {
      resultadoLabel.setText("Primero debes crear la estructura.");
      return;
  }

    // 1) Tomar todas las claves no vacías
    var claves = data.stream()
        .map(SlotClave::getClave)
        .filter(c -> c != null && !c.isBlank())
        .sorted() // orden ascendente (funciona bien porque están normalizadas con
ceros)
        .toList();

    // 2) Vaciar la estructura
    for (SlotClave s : data) {
      s.setClave("");
    }

    // 3) Reinsertar en orden desde la posición 0
    for (int i =1; i <= claves.size(); i++) {
      data.get(i).setClave(claves.get(i));
    }

    tabla.refresh();
    resultadoLabel.setText("Claves ordenadas de menor a mayor.");
```

```
    }


    private Integer leerEntero(TextField tf) {
        try {
            String t = tf.getText();
            if (t == null || t.trim().isEmpty()) return null;
            return Integer.parseInt(t.trim());
        } catch (Exception e) {
            return null;
        }
    }

    private String normalizarClave(String clave, int digitos) {
    if (clave == null) return "";

    clave = clave.trim();

    // Si no son números, la devolvemos igual
    if (!clave.matches("\\d+")) return clave;

    // Completa con ceros a la izquierda
    return String.format("%0" + digitos + "d", Integer.parseInt(clave));
    }

    private boolean claveValidaPorDigitos(String clave, int digitos) {
        if (clave == null) return false;
        if (clave.length() != digitos) return false;
        for (int i = 0; i < clave.length(); i++) {
            if (!Character.isDigit(clave.charAt(i))) return false;
        }
        return true;
    }
}
```

```
package controller;

import javafx.fxml.FXML;
```

```
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.layout.StackPane;

public class InicioController {

  @FXML
  private StackPane contentPane;

  @FXML
  private void mostrarBusquedaLineal() {
    System.out.println("Abriendo busquedaLineal.fxml");
    loadPanel("busquedaLineal.fxml");
  }

  @FXML
  private void mostrarBusquedaBinario() {
    System.out.println("Clic en Búsqueda Binaria");
  }

  @FXML
  private void mostrarBusquedaHash() {
    System.out.println("Clic en Búsqueda Hash");
  }

  @FXML
  private void mostrarEstructuraEstatica() {
    System.out.println("Clic en Estructura Estática");
  }

  @FXML
  private void mostrarEstructuraDinamica() {
    System.out.println("Clic en Estructura Dinámica");
  }

  private void loadPanel(String fxml) {
    try {
      FXMLLoader loader = new FXMLLoader(getClass().getResource("/" + fxml));
```

```
            Parent panel = loader.load();

            contentPane.getChildren().clear();
            contentPane.getChildren().add(panel);

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

```
package utilities;

public class BusquedaLinealService {

    public record ResultadoBusqueda(
            boolean encontrado,
            int indice,
            int comparaciones,
            long nanos
    ) {}

    public ResultadoBusqueda buscar(int[] arr, int objetivo) {
        int comparaciones = 0;
        long inicio = System.nanoTime();

        for (int i = 0; i < arr.length; i++) {
            comparaciones++;
            if (arr[i] == objetivo) {
                long fin = System.nanoTime();
                return new ResultadoBusqueda(true, i, comparaciones, fin - inicio);
            }
        }

        long fin = System.nanoTime();
        return new ResultadoBusqueda(false, -1, comparaciones, fin - inicio);
    }
}
```

```
package utilities;

public class Paths {

   public static final String inicio = "/inicio.fxml";
}
```

```
package utilities;

import javafx.beans.property.SimpleIntegerProperty;
import javafx.beans.property.SimpleStringProperty;

public class SlotClave {
   private final SimpleIntegerProperty posicion = new SimpleIntegerProperty();
   private final SimpleStringProperty clave = new SimpleStringProperty("");

   public SlotClave(int posicion, String clave) {
      this.posicion.set(posicion);
      this.clave.set(clave);
   }

   public int getPosicion() { return posicion.get(); }
   public void setPosicion(int v) { posicion.set(v); }

   public String getClave() { return clave.get(); }
   public void setClave(String v) { clave.set(v); }
}
```

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<VBox xmlns="http://javafx.com/javafx/17"
    xmlns:fx="http://javafx.com/fxml/1"
    fx:controller="controller.BusquedaLinealController"
    spacing="12">
```

```
  <padding>
    <Insets top="16" right="16" bottom="16" left="16"/>
  </padding>


  <Label text="Búsqueda Secuencial" style="-fx-font-size: 18px; -fx-font-weight:
bold;"/>

  <!-- Configuración -->
  <HBox spacing="10">
    <Label text="Tamaño estructura (N):"/>
    <TextField fx:id="nField" promptText="Ej: 10" prefWidth="100"/>

    <Label text="Dígitos clave:"/>
    <ChoiceBox fx:id="digitosChoice" prefWidth="80"/>

    <Button text="Crear" onAction="#crearEstructura"/>
    <Button text="Ordenar" onAction="#ordenarClaves"/>

  </HBox>

  <!-- Tabla -->
  <TableView fx:id="tabla" prefHeight="260">
    <columns>
      <TableColumn fx:id="colPos" text="Posición" prefWidth="120"/>
      <TableColumn fx:id="colClave" text="Clave" prefWidth="200"/>
    </columns>
  </TableView>

  <!-- Insertar clave -->
  <HBox spacing="10">
    <Label text="Insertar clave:"/>
    <TextField fx:id="claveInsertField" promptText="Ej: 24" prefWidth="120"/>
    <Button text="Insertar" onAction="#insertarClave"/>
  </HBox>

  <!-- Buscar clave -->
  <HBox spacing="10">
```

```
    <Label text="Buscar clave:"/>
    <TextField fx:id="claveBuscarField" promptText="Ej: 24" prefWidth="120"/>
    <Button text="Buscar" onAction="#buscarClave"/>
  </HBox>

  <Label fx:id="resultadoLabel" text="Listo." style="-fx-font-size: 14px;"/>

</VBox>
```

```
<?xml version="1.0" encoding="UTF-8"?>

<?import java.net.*?>
<?import javafx.geometry.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.image.*?>
<?import javafx.scene.layout.*?>
<?import javafx.scene.text.*?>

<AnchorPane prefHeight="400.0" prefWidth="600.0"
xmlns="http://javafx.com/javafx/17.0.12" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="controller.InicioController">
  <!--agregar la hoja de estilos-->
  <stylesheets>
    <URL value="@styles.css" />
  </stylesheets>
  <children>
    <BorderPane prefHeight="400.0" prefWidth="600.0">
      <!-- Lateral izquierdo -->
      <left>
        <Pane prefHeight="343.0" prefWidth="149.0" style="-fx-background-color:
#2262C6; -fx-background-radius: 0 60 0 0;" BorderPane.alignment="CENTER">
          <children>
          <VBox layoutX="3.0" layoutY="38.0" prefHeight="281.0" prefWidth="142.0"
spacing="50">
            <children>
              <!-- Menú Búsquedas -->
              <MenuBar>
                <VBox.margin>
```

```
                    <Insets top="35" />
                  </VBox.margin>
                  <Menu text="Inicio" />
                </MenuBar>
                <MenuBar>
                  <Menu text="Búsquedas">
                    <Menu text="Internas">
                      <MenuItem text="Lineal" onAction="#mostrarBusquedaLineal" />
                      <MenuItem text="Binario" onAction="#mostrarBusquedaBinario"/>
                      <MenuItem text="Función Hash"
onAction="#mostrarBusquedaHash"/>
                    </Menu>
                    <Menu text="Externas">
                      <MenuItem text="Estructura Estática"
onAction="#mostrarEstructuraEstatica" />
                      <MenuItem text="Estructura Dinámica"
onAction="#mostrarEstructuraDinamica" />
                    </Menu>
                  </Menu>
                </MenuBar>

                <!-- Menú Grafos -->
                <MenuBar>
                  <Menu text="Grafos">
                    <Menu text="DFS">
                      <MenuItem text="Recorrido básico" />
                      <MenuItem text="Recorrido completo" />
                    </Menu>
                    <Menu text="BFS">
                      <MenuItem text="Recorrido por niveles" />
                      <MenuItem text="Recorrido completo" />
                    </Menu>
                  </Menu>
                </MenuBar>
              </children>
            </VBox>
            </children>
          </Pane>
```

```
        </left>

        <!-- Área central -->
        <center>
          <StackPane fx:id="contentPane" style="-fx-background-color: white;" />
        </center>

        <!-- Barra superior -->
        <top>
          <Pane prefHeight="57.0" prefWidth="600.0" style="-fx-background-color:
#FFF;" BorderPane.alignment="CENTER">
           <children>
            <ImageView fitHeight="45.0" fitWidth="113.0" layoutX="14.0" layoutY="6.0"
pickOnBounds="true" preserveRatio="true">
              <image>
                <Image url="@icono-data-science.png" />
              </image></ImageView>
            <Label layoutX="75.0" layoutY="6.0" prefHeight="41.0" prefWidth="420.0"
text="CIENCIAS DE LA COMPUTACIÓN II" textFill="#2262c6">
              <font>
                <Font name="Montserrat SemiBold" size="23.0" />
              </font>
            </Label>
           </children></Pane>
        </top>
      </BorderPane>
   </children>
</AnchorPane>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>org.example</groupId>
  <artifactId>AppCiencias2</artifactId>
```

```
<version>1.0-SNAPSHOT</version>

<properties>
  <maven.compiler.source>21</maven.compiler.source>
  <maven.compiler.target>21</maven.compiler.target>
  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

  <javafx.version>21.0.2</javafx.version>
  <javafx.platform>win</javafx.platform>
</properties>

<dependencies>
  <dependency>
    <groupId>org.openjfx</groupId>
    <artifactId>javafx-base</artifactId>
    <version>${javafx.version}</version>
    <classifier>${javafx.platform}</classifier>
  </dependency>

  <dependency>
    <groupId>org.openjfx</groupId>
    <artifactId>javafx-graphics</artifactId>
    <version>${javafx.version}</version>
    <classifier>${javafx.platform}</classifier>
  </dependency>

  <dependency>
    <groupId>org.openjfx</groupId>
    <artifactId>javafx-controls</artifactId>
    <version>${javafx.version}</version>
    <classifier>${javafx.platform}</classifier>
  </dependency>

  <dependency>
    <groupId>org.openjfx</groupId>
    <artifactId>javafx-fxml</artifactId>
    <version>${javafx.version}</version>
    <classifier>${javafx.platform}</classifier>
```

```
    </dependency>

    <!-- Solo si lo usas -->
    <dependency>
      <groupId>org.openjfx</groupId>
      <artifactId>javafx-media</artifactId>
      <version>${javafx.version}</version>
      <classifier>${javafx.platform}</classifier>
    </dependency>
  </dependencies>

  <build>
    <plugins>
      <plugin>
        <groupId>org.openjfx</groupId>
        <artifactId>javafx-maven-plugin</artifactId>
        <version>0.0.8</version>
        <configuration>
          <mainClass>application.App</mainClass>
        </configuration>
      </plugin>
    </plugins>
  </build>

</project>
```