UNIVERSIDAD PERUANA LOS ANDES

FACULTAD DE INGENIERÍA

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN

PRACTICA N°1

ASIGNATURA: ARQUITECTURA DE SOFTWARE

DOCENTE: FERNÁNDEZ BEJARANO RAUL ENRIQUE

INTEGRANTE: AQUINO CANO, MOISES ISRAEL

CICLO: VI

HUANCAYO-2024
PERÚ

INFORME DE TRABAJO N°1

TÍTULO: SISTEMA DE REGISTRO DE ALUMNOS

El "Sistema de Registro de Alumnos" es una aplicación de escritorio creada en Java, destinada a la gestión eficiente del registro de estudiantes. Esta herramienta permite a los usuarios agregar, modificar, eliminar y buscar información de los alumnos, incluyendo datos como nombre, apellido, código, DNI, teléfono y fecha de nacimiento.

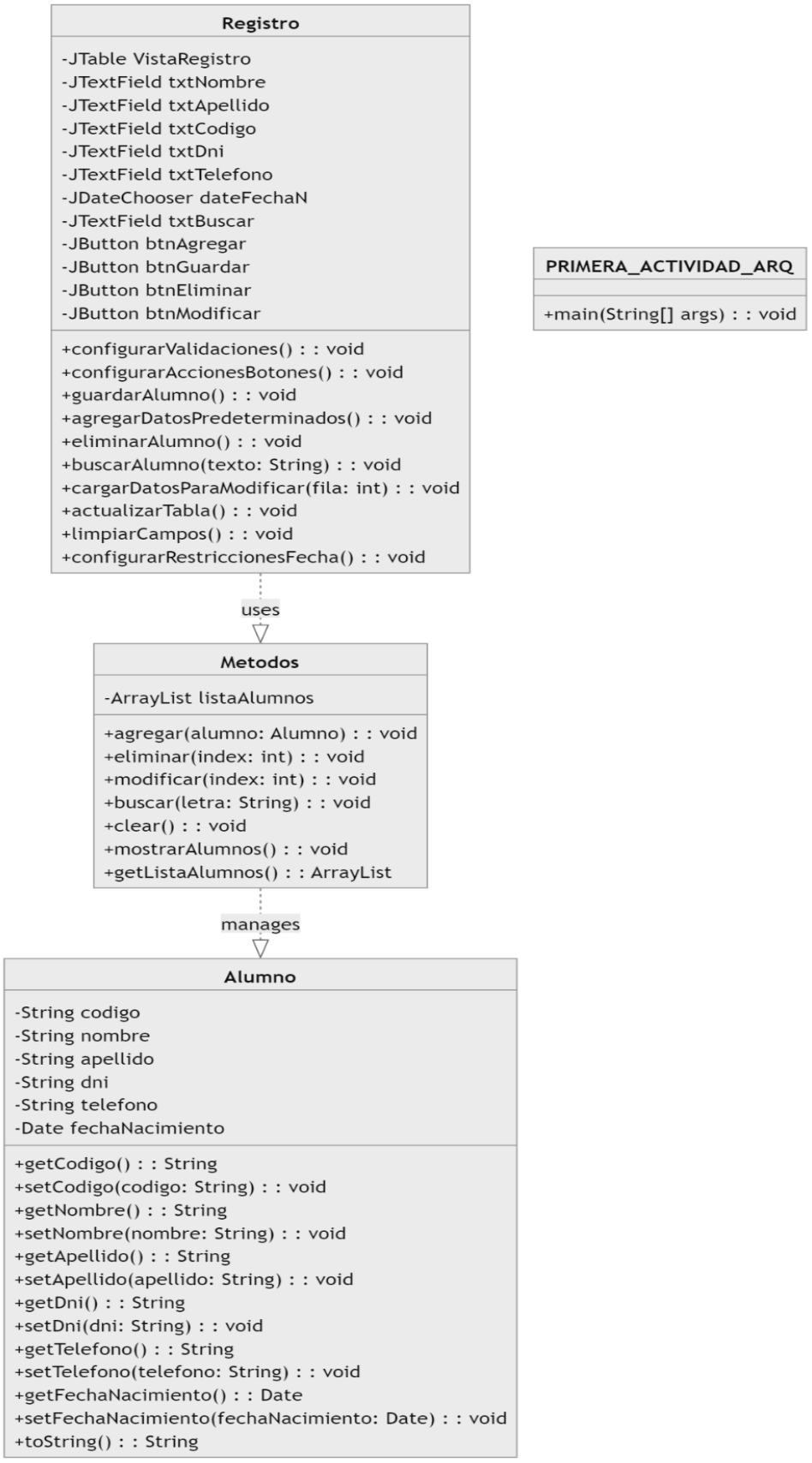
El propósito fundamental de esta aplicación es optimizar la administración de los datos de los estudiantes, garantizando un registro actualizado de manera intuitiva y eficaz. Además, el sistema incluye validaciones que aseguran la precisión de los datos ingresados, como la limitación en el número de dígitos para campos como el DNI y el teléfono. Además del uso de librerías para una mejor interfaz.

REQUERIMIENTOS FUNCIONALES

A continuación, se presenta los requerimientos funcionales del sistema:

1. **Agregar alumnos**:
    - El usuario puede agregar nuevos alumnos al sistema ingresando su nombre, apellido, código, DNI, teléfono y fecha de nacimiento.
    - El sistema valida que el DNI sea de 8 dígitos, el teléfono de 9 dígitos y que el código tenga 7 caracteres (alfanuméricos).
2. **Modificar alumnos**:
    - El usuario puede seleccionar un alumno de la lista y modificar su información. El sistema cargará los datos en los campos correspondientes para su edición.
3. **Eliminar alumnos**:
    - El usuario puede seleccionar un alumno de la lista y eliminarlo del sistema. Esto también actualizará automáticamente la tabla de registro.
4. **Buscar alumnos**:
    - El usuario puede buscar alumnos por nombre, apellido, código o DNI. Los resultados se muestran en tiempo real a medida que el usuario escribe en el campo de búsqueda.
5. **Validación de campos**:
    - La aplicación valida que los datos ingresados sean correctos, limitando la cantidad de caracteres en los campos de DNI, teléfono y código.
6. **Visualización de registros**:
    - Todos los alumnos registrados se muestran en una tabla, con la posibilidad de actualizar la lista al agregar, modificar o eliminar alumnos.

DIAGRAMA DE CLASES



CÓDIGO

CLASE ALUMNO

La clase `Alumno` encapsula toda la información relacionada con un alumno. Sus métodos permiten acceder y manipular los datos del alumno, lo que es esencial para el funcionamiento de la aplicación de registro de alumnos.

```
 */
package com.mycompany.primera_actividad_arq;

import java.util.Date;

/**
 *
 * @author ADMIN
 */
public class Alumno {
   private String codigo;
    private String nombre;
    private String apellido;
    private String dni;
    private String telefono;
    private Date fechaNacimiento;

    // Constructor vacío
    public Alumno() {
    }

    // Constructor con parámetros
    public Alumno(String codigo, String nombre, String apellido, String dni, String telefono,
Date fechaNacimiento) {
        this.codigo = codigo;
        this.nombre = nombre;
        this.apellido = apellido;
        this.dni = dni;
        this.telefono = telefono;
        this.fechaNacimiento = fechaNacimiento;
    }

    // Getters y Setters
    public String getCodigo() {
        return codigo;
    }

    public void setCodigo(String codigo) {
```

```
        this.codigo = codigo;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public String getApellido() {
        return apellido;
    }

    public void setApellido(String apellido) {
        this.apellido = apellido;
    }

    public String getDni() {
        return dni;
    }

    public void setDni(String dni) {
        this.dni = dni;
    }

    public String getTelefono() {
        return telefono;
    }

    public void setTelefono(String telefono) {
        this.telefono = telefono;
    }

    public Date getFechaNacimiento() {
        return fechaNacimiento;
    }

    public void setFechaNacimiento(Date fechaNacimiento) {
        this.fechaNacimiento = fechaNacimiento;
    }

    @Override
    public String toString() {
        return "Alumno{" + "codigo=" + codigo + ", nombre=" + nombre + ", apellido=" + apellido
+
```

```
            ", dni=" + dni + ", telefono=" + telefono + ", fechaNacimiento=" + fechaNacimiento +
'}';
    }
}
```

CLASE METODOS

La clase **Metodos** es responsable de gestionar las operaciones relacionadas con el registro de alumnos en el sistema. Proporciona métodos para **agregar**, **eliminar**, **modificar**, **buscar** y **listar** alumnos almacenados en una colección interna

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change
this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.primera_actividad_arq;

import java.util.ArrayList;
import java.util.Scanner;

/**
 *
 * @author ADMIN
 */

public class Metodos {
    private ArrayList<Alumno> listaAlumnos;

    // Constructor que inicializa el ArrayList
    public Metodos() {
        listaAlumnos = new ArrayList<>();
    }

    // Método para agregar un Alumno al ArrayList
    public void agregar(Alumno alumno) {
        listaAlumnos.add(alumno);
        System.out.println("Alumno agregado: " + alumno);
    }

    // Método para eliminar un Alumno del ArrayList en una posición específica
    public void eliminar(int index) {
        if (index >= 0 && index < listaAlumnos.size()) {
            Alumno eliminado = listaAlumnos.remove(index);
            System.out.println("Alumno eliminado: " + eliminado);
        } else {
            System.out.println("Índice fuera de rango");
        }
```

```
    }

    // Método para modificar un campo específico de un alumno
    public void modificar(int index) {
        if (index >= 0 && index < listaAlumnos.size()) {
            Alumno alumno = listaAlumnos.get(index);
            Scanner sc = new Scanner(System.in);
            System.out.println("Seleccione el campo a modificar:\n1. Nombre\n2. Apellido\n3.
Código\n4. DNI\n5. Teléfono\n6. Fecha de Nacimiento");
            int opcion = sc.nextInt();
            sc.nextLine(); // Limpiar el buffer
            switch (opcion) {
                case 1:
                    System.out.println("Ingrese nuevo nombre:");
                    alumno.setNombre(sc.nextLine());
                    break;
                case 2:
                    System.out.println("Ingrese nuevo apellido:");
                    alumno.setApellido(sc.nextLine());
                    break;
                case 3:
                    System.out.println("Ingrese nuevo código:");
                    alumno.setCodigo(sc.nextLine());
                    break;
                case 4:
                    System.out.println("Ingrese nuevo DNI:");
                    alumno.setDni(sc.nextLine());
                    break;
                case 5:
                    System.out.println("Ingrese nuevo teléfono:");
                    alumno.setTelefono(sc.nextLine());
                    break;
                case 6:
                    System.out.println("Ingrese nueva fecha de nacimiento (yyyy-mm-dd):");
                    // Aquí puedes hacer una conversión adecuada de String a Date
                    // Dependiendo de cómo manejes las fechas en tu programa.
                    // alumno.setFechaNacimiento(fechaConvertida);
                    break;
                default:
                    System.out.println("Opción no válida");
                    break;
            }
            System.out.println("Alumno modificado: " + alumno);
        } else {
            System.out.println("Índice fuera de rango");
        }
    }
```

```
    // Método para buscar por la primera letra de nombre, apellido, código o dni
    public void buscar(String letra) {
        letra = letra.toLowerCase(); // Convertimos a minúsculas para evitar problemas con
mayúsculas
        boolean encontrado = false;

        for (Alumno alumno : listaAlumnos) {
            if (alumno.getNombre().toLowerCase().startsWith(letra) ||
                alumno.getApellido().toLowerCase().startsWith(letra) ||
                alumno.getCodigo().toLowerCase().startsWith(letra) ||
                alumno.getDni().toLowerCase().startsWith(letra)) {
                System.out.println("Alumno encontrado: " + alumno);
                encontrado = true;
            }
        }

        if (!encontrado) {
            System.out.println("No se encontraron alumnos con la letra: " + letra);
        }
    }

    // Método para eliminar todos los alumnos del ArrayList
    public void clear() {
        listaAlumnos.clear();
        System.out.println("Todos los alumnos han sido eliminados.");
    }

    // Método para mostrar todos los alumnos
    public void mostrarAlumnos() {
        if (listaAlumnos.isEmpty()) {
            System.out.println("No hay alumnos en la lista.");
        } else {
            for (Alumno alumno : listaAlumnos) {
                System.out.println(alumno);
            }
        }
    }

    // Método para obtener la lista de alumnos
    public ArrayList<Alumno> getListaAlumnos() {
        return listaAlumnos;
    }
}
```

JFRAME
La clase **Registro** es una interfaz gráfica de usuario (GUI) creada utilizando **Swing**. Su objetivo principal es permitir al usuario gestionar los registros de alumnos, lo que incluye operaciones como agregar, modificar, eliminar, buscar y visualizar alumnos en una tabla.

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template
 */
package Vista;

import com.mycompany.primera_actividad_arq.Alumno;
import com.mycompany.primera_actividad_arq.Metodos;
import java.text.SimpleDateFormat; // Asegúrate de importar esta clase
import javax.swing.table.DefaultTableModel;
import java.util.Date;
import javax.swing.JOptionPane;
import java.util.Calendar;
/**
 *
 * @author ADMIN
 */
public class Registro extends javax.swing.JFrame {

    // Crear una instancia de la clase Metodos para manejar el ArrayList
    Metodos metodos = new Metodos();

    /**
     * Creates new form Registro
     */
    // Variables para control de modificación
    private boolean esModificacion = false;
    private int filaSeleccionada = -1; // Para almacenar el índice de la fila seleccionada para modificar


    public Registro() {
        initComponents();
        configurarValidaciones();
        configurarAccionesBotones();
```

```
        actualizarTabla(); // Inicializa la tabla con los datos actuales
        configurarRestriccionesFecha(); // Establece restricciones de fechas en el
JDateChooser
        agregarDatosPredeterminados();
        setLocationRelativeTo (null);
    }


    // Configurar los KeyListeners para validación de campos
    private void configurarValidaciones() {
        // Para el campo Código (7 caracteres, letras y números)
        txtCodigo.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyTyped(java.awt.event.KeyEvent evt) {
                if (txtCodigo.getText().length() >= 7) {
                    evt.consume(); // Limitar a 7 caracteres
                }
                char c = evt.getKeyChar();
                if (!Character.isLetterOrDigit(c)) {
                    evt.consume(); // Solo letras y números
                }
            }
        });

        // Para el campo DNI (8 dígitos)
        txtDni.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyTyped(java.awt.event.KeyEvent evt) {
                if (txtDni.getText().length() >= 8) {
                    evt.consume(); // Limitar a 8 dígitos
                }
                char c = evt.getKeyChar();
                if (!Character.isDigit(c)) {
                    evt.consume(); // Solo dígitos
                }
            }
        });

        // Para el campo Teléfono (9 dígitos)
        txtTelefono.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyTyped(java.awt.event.KeyEvent evt) {
                if (txtTelefono.getText().length() >= 9) {
                    evt.consume(); // Limitar a 9 dígitos
                }
                char c = evt.getKeyChar();
                if (!Character.isDigit(c)) {
                    evt.consume(); // Solo dígitos
                }
            }
        });
```

```
    }

    private void configurarAccionesBotones() {
    // Acción del botón Agregar
    btnAgregar.addActionListener(evt -> {
        limpiarCampos();  // Limpiar los campos del formulario
        esModificacion = false;  // Indicar que no se está en modo de modificación
    });

    // Acción del botón Guardar
    btnGuardar.addActionListener(evt -> guardarAlumno());

    // Acción del botón Eliminar
    btnEliminar.addActionListener(evt -> eliminarAlumno());

    btnModificar.addActionListener(evt -> {
    filaSeleccionada = VistaRegistro.getSelectedRow(); // Obtener la fila seleccionada
    if (filaSeleccionada >= 0) {
        esModificacion = true;  // Cambiar el estado a "modificación"
        cargarDatosParaModificar(filaSeleccionada); // Cargar los datos del alumno en el
formulario
    } else {
        JOptionPane.showMessageDialog(this, "Seleccione un alumno para modificar");
    }
});

    // Acción para la búsqueda en tiempo real
    txtBuscar.addKeyListener(new java.awt.event.KeyAdapter() {
        public void keyReleased(java.awt.event.KeyEvent evt) {
            buscarAlumno(txtBuscar.getText());
        }
    });
}

    private void guardarAlumno() {
    String nombre = txtNombre.getText();
    String apellido = txtApellido.getText();
    String codigo = txtCodigo.getText();
    String dni = txtDni.getText();
    String telefono = txtTelefono.getText();
    Date fechaNac = dateFechaN.getDate(); // Captura la fecha completa desde
JDateChooser

    // Verificar que todos los campos estén completos
    if (nombre.isEmpty() || apellido.isEmpty() || codigo.isEmpty() || dni.isEmpty() ||
telefono.isEmpty() || fechaNac == null) {
        JOptionPane.showMessageDialog(this, "Todos los campos son obligatorios");
        return;
```

```
    }

    // Verificación adicional del formato de fecha
    SimpleDateFormat sdf = new SimpleDateFormat("dd-MM-yyyy");
    System.out.println("Fecha capturada: " + sdf.format(fechaNac)); // Verificar en consola

    if (esModificacion) {
        // Si es modificación, actualiza el alumno seleccionado
        Alumno alumnoModificado = metodos.getListaAlumnos().get(filaSeleccionada);  //
Obtener el objeto Alumno original
        alumnoModificado.setNombre(nombre);
        alumnoModificado.setApellido(apellido);
        alumnoModificado.setCodigo(codigo);
        alumnoModificado.setDni(dni);
        alumnoModificado.setTelefono(telefono);
        alumnoModificado.setFechaNacimiento(fechaNac); // Actualizar la fecha completa (día,
mes y año)

        esModificacion = false;  // Restablecer el estado
        filaSeleccionada = -1;   // Restablecer la fila seleccionada
        JOptionPane.showMessageDialog(this, "Alumno modificado correctamente.");
    } else {
        // Si no es modificación, agregar un nuevo alumno
        Alumno nuevoAlumno = new Alumno(codigo, nombre, apellido, dni, telefono,
fechaNac);
        metodos.agregar(nuevoAlumno);
        JOptionPane.showMessageDialog(this, "Alumno agregado correctamente.");
    }

    actualizarTabla();  // Actualizar la tabla para reflejar los cambios
    limpiarCampos();    // Limpiar los campos del formulario
}



    private void agregarDatosPredeterminados() {
    SimpleDateFormat sdf = new SimpleDateFormat("dd-MM-yyyy");

    try {
        // Agregar 5 alumnos con datos de ejemplo
        metodos.agregar(new Alumno("001", "Carlos", "Pérez", "12345678", "987654321",
sdf.parse("01-01-2000")));
        metodos.agregar(new Alumno("002", "María", "Gómez", "87654321", "987123456",
sdf.parse("02-02-1999")));
        metodos.agregar(new Alumno("003", "Juan", "Lopez", "12341234", "912345678",
sdf.parse("03-03-1998")));
        metodos.agregar(new Alumno("004", "Ana", "Martínez", "43214321", "956789123",
sdf.parse("04-04-1997")));
```

```
        metodos.agregar(new Alumno("005", "Luis", "Sánchez", "56785678", "987456123",
sdf.parse("05-05-1996")));

        // Después de agregar los alumnos, actualizar la tabla
        actualizarTabla();
    } catch (Exception e) {
        e.printStackTrace();
    }
}


    // Método para eliminar un alumno de la lista y la tabla
    private void eliminarAlumno() {
        int filaSeleccionada = VistaRegistro.getSelectedRow();
        if (filaSeleccionada >= 0) {
            metodos.eliminar(filaSeleccionada);
            actualizarTabla();
        } else {
            JOptionPane.showMessageDialog(this, "Seleccione un alumno para eliminar");
        }
    }

    // Método para buscar alumno por primera letra
    private void buscarAlumno(String texto) {
        DefaultTableModel model = (DefaultTableModel) VistaRegistro.getModel();
        model.setRowCount(0);
        for (Alumno alumno : metodos.getListaAlumnos()) {
            if (alumno.getNombre().toLowerCase().startsWith(texto.toLowerCase()) ||
                alumno.getApellido().toLowerCase().startsWith(texto.toLowerCase()) ||
                alumno.getCodigo().toLowerCase().startsWith(texto.toLowerCase()) ||
                alumno.getDni().toLowerCase().startsWith(texto.toLowerCase())) {
                SimpleDateFormat sdf = new SimpleDateFormat("dd-MM-yyyy");
                String fechaNacimiento = sdf.format(alumno.getFechaNacimiento());
                model.addRow(new Object[]{alumno.getNombre(), alumno.getApellido(),
alumno.getCodigo(), alumno.getDni(), alumno.getTelefono(), fechaNacimiento});
            }
        }
    }

    private void cargarDatosParaModificar(int fila) {
    Alumno alumno = metodos.getListaAlumnos().get(fila);  // Obtener el alumno de la lista
    txtNombre.setText(alumno.getNombre());
    txtApellido.setText(alumno.getApellido());
    txtCodigo.setText(alumno.getCodigo());
    txtDni.setText(alumno.getDni());
    txtTelefono.setText(alumno.getTelefono());
    dateFechaN.setDate(alumno.getFechaNacimiento());  // Establecer la fecha en el
JDateChooser
```

```
}


   private void actualizarTabla() {
    DefaultTableModel model = (DefaultTableModel) VistaRegistro.getModel();
    model.setRowCount(0);  // Limpiar la tabla antes de llenarla nuevamente

    for (Alumno alumno : metodos.getListaAlumnos()) {
        SimpleDateFormat sdf = new SimpleDateFormat("dd-MM-yyyy");
        String fechaNacimiento = sdf.format(alumno.getFechaNacimiento());
        model.addRow(new Object[]{alumno.getNombre(), alumno.getApellido(),
alumno.getCodigo(), alumno.getDni(), alumno.getTelefono(), fechaNacimiento});
    }
}




    // Método para limpiar los campos de texto
    private void limpiarCampos() {
        txtNombre.setText("");
        txtApellido.setText("");
        txtCodigo.setText("");
        txtDni.setText("");
        txtTelefono.setText("");
        dateFechaN.setDate(null); // Limpiar la fecha
    }

     // Método para establecer restricciones de fecha en el JDateChooser
        private void configurarRestriccionesFecha() {
        Calendar cal = Calendar.getInstance();
        Date fechaMaxima = cal.getTime();  // Fecha actual (no permitir fechas futuras)

        cal.add(Calendar.YEAR, -17);  // Restar 17 años para obtener la fecha mínima
permitida
        Date fechaMinima = cal.getTime();

        dateFechaN.setMinSelectableDate(fechaMinima);  // Establece la fecha mínima (17
años atrás)
        dateFechaN.setMaxSelectableDate(fechaMaxima);  // Establece la fecha máxima (hoy)
    }
```

MAIN

Esta clase contiene el método `main` que es el punto de inicio de la ejecución de tu aplicación. Su propósito principal es configurar la apariencia visual del formulario utilizando **JTattoo** y luego inicializar y mostrar la ventana principal de la aplicación.

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 */

package com.mycompany.primera_actividad_arq;

import Vista.Registro;  // Importa la clase Registro del paquete Vista
import javax.swing.UIManager;
/**
 *
 * @author ADMIN
 */
public class PRIMERA_ACTIVIDAD_ARQ {

    public static void main(String[] args) {
        try {
            // Establecer el tema de JTattoo (puedes cambiar el tema si lo deseas)
            UIManager.setLookAndFeel("com.jtattoo.plaf.graphite.GraphiteLookAndFeel");  // Tema Graphite de JTattoo
        } catch (Exception ex) {
            ex.printStackTrace();
        }

        // Crear una instancia del JFrame Registro
        Registro registro = new Registro();

        // Mostrar el formulario
        registro.setVisible(true);
    }
}
```