“Año del Bicentenario, de la consolidación de nuestra Independencia, y de la conmemoración de las heroicas batallas de Junín y Ayacucho”

“UNIVERSIDAD PERUANA LOS ANDES”

FACULTAD DE INGENIERÍA

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN

Trabajo 1

Alumno:

- Pariona Gozme Maicol

Docente: Fernandez Bejarano Raul Enrique

Asignatura: ARQUITECTURA DE SOFTWARE

Sección: B1

Ciclo: VII

Huancayo - Perú

2024

Sistema de Registro de Alumnos

El objetivo de este proyecto es desarrollar un sistema de registro de alumnos que permita gestionar la información de los estudiantes de una universidad. El sistema debe permitir agregar, modificar, buscar y eliminar registros de alumnos, proporcionando una interfaz de usuario sencilla y clara. La información que se gestionará incluye nombre, fecha de nacimiento, código de estudiante, teléfono, carrera, y semestre.

Requerimientos Funcionales

Nombre:	***R1: Agregar Alumnos***
Resumen:	***El sistema debe permitir agregar un nuevo alumno proporcionando información relevante.***
Entrada:	***Nombre, fecha de nacimiento, código de estudiante, teléfono, carrera, semestre.***
Resultados:	***El nuevo alumno se registra en el sistema con la información proporcionada.***

Nombre:	***R2: Eliminar Alumnos***
Resumen:	***El sistema debe permitir eliminar a un alumno de la lista existente.***
Entrada:	***Selección de un alumno en la lista.***
Resultados:	***El alumno seleccionado se elimina del sistema.***

Nombre:	***R3: Buscar Alumnos***
Resumen:	***El sistema debe permitir buscar alumnos por su nombre.***
Entrada:	***Letra inicial del nombre del alumno.***
Resultados:	***El sistema muestra una lista de todos los alumnos cuyo nombre comienza con la letra proporcionada.***

Nombre:	***R4: Actualizar Información***
Resumen:	***El sistema debe permitir actualizar la información de un alumno registrado.***
Entrada:	***Datos actualizados del alumno (nombre, fecha de nacimiento, código de estudiante, teléfono, carrera, semestre).***
Resultados:	***La información del alumno se actualiza en el sistema.***

Nombre:	***R5: Mostrar Lista de Alumnos***
Resumen:	***El sistema debe mostrar la lista completa de los alumnos registrados.***
Entrada:	***Ninguna entrada requerida.***
Resultados:	***Se muestra una tabla con los detalles de los alumnos registrados en el sistema.***

Diagrama de Clases

1. Clase: `JframeRegistro`

- **Descripción:** Esta clase representa la interfaz gráfica del sistema de registro de alumnos. Permite agregar, borrar, actualizar y buscar alumnos, y muestra los resultados en una tabla.

- **Atributos:**
  - `listaAlumnos`: `ArrayList<Alumnos>` → Lista que almacena los objetos tipo `Alumnos`.
  - `txtNombre`: `JTextField` → Campo de texto para ingresar el nombre del alumno.
  - `txtCodigo`: `JTextField` → Campo de texto para ingresar el código del alumno.
  - `txtTelefono`: `JTextField` → Campo de texto para ingresar el teléfono del alumno.
  - `jComboBoxCarrera`: `JComboBox` → Menú desplegable para seleccionar la carrera del alumno.
  - `jComboBoxSemestre`: `JComboBox` → Menú desplegable para seleccionar el semestre del alumno.

- `btnAgregar`: JButton → Botón para agregar un nuevo alumno.
- `btnBorrar`: JButton → Botón para borrar un alumno de la lista.
- `btnActualizar`: JButton → Botón para actualizar la información de un alumno.
- `tblRegistro`: JTable → Tabla que muestra los registros de los alumnos.

- **Métodos:**
  - `agregarFila(alumno: Alumnos): void`
    - Añade un nuevo alumno a la tabla de registros y la lista.
  - `actualizarTabla(row: int, alumno: Alumnos): void`
    - Actualiza la información de un alumno en una fila específica de la tabla.
  - `eliminarAlumno(): void`
    - Elimina un alumno de la lista y de la tabla.
  - `buscarAlumno(nombre: String): List<Alumnos>`
    - Busca alumnos por su nombre en la lista y devuelve los resultados.

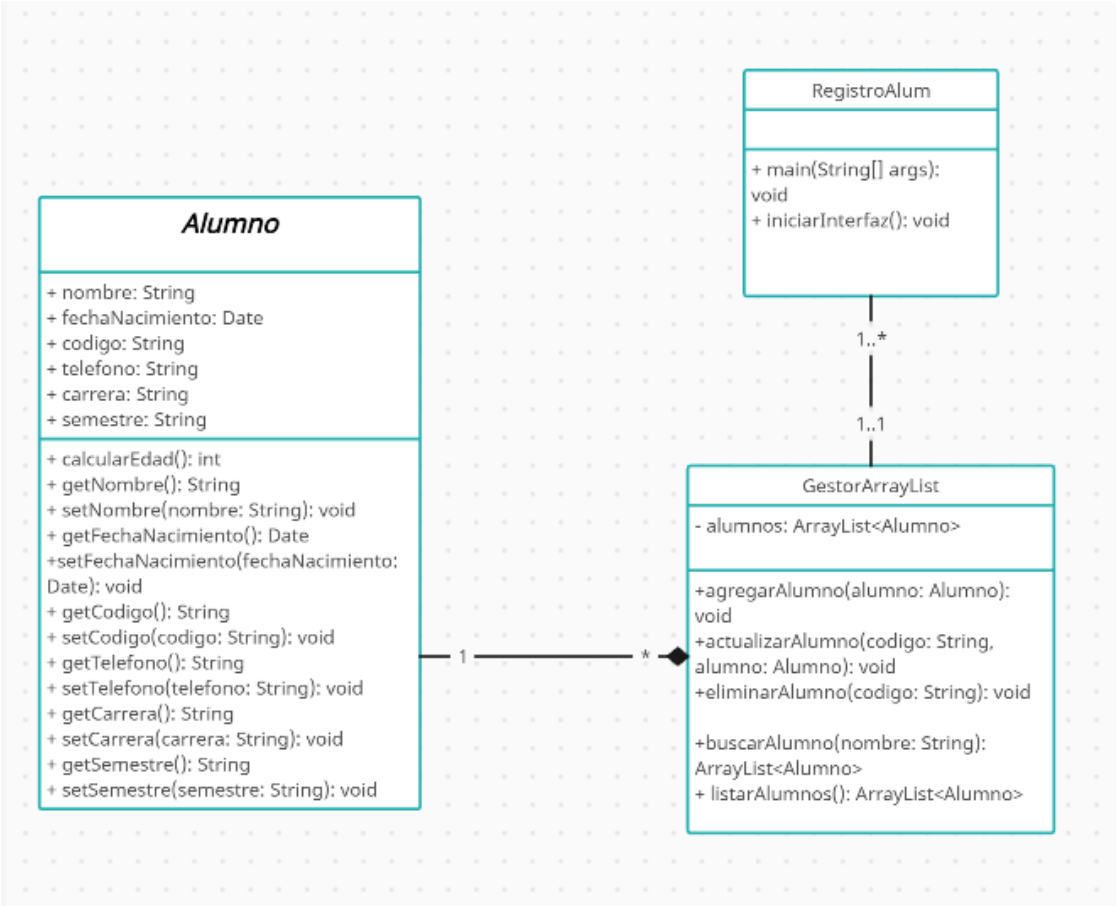
2. Clase: `Alumnos`

- Descripción: Esta clase representa a los alumnos con toda la información requerida, como nombre, código, carrera, semestre, etc.
- **Atributos:**
  - `nombre`: `String` → Nombre del alumno.
  - `fecha`: `String` → Fecha de nacimiento del alumno (se utiliza para calcular la edad).
  - `codigo`: `String` → Código único del alumno.
  - `telefono`: `String` → Número de teléfono del alumno.
  - `carrera`: `String` → Carrera que el alumno está cursando.
  - `semestre`: `String` → Semestre en el que se encuentra el alumno.
- **Métodos:**
  - `calcularEdad(): int`
    - Calcula y devuelve la edad del alumno en base a su fecha de nacimiento.

Relaciones:

- La clase `JframeRegistro` contiene una lista de objetos de la clase `Alumnos` mediante el atributo `listaAlumnos` (composición).

- Dependencias: Las acciones de agregar, actualizar, eliminar y buscar alumnos dependen de la lista gestionada en la clase `JframeRegistro`, mientras que los datos de los alumnos son instancias de la clase `Alumnos`.



Código

Alumnos

```
/*
 * 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 Registro;

/**
 *
 * @author USER 17
 */
public class Alumnos {
    String nombre;
    String fecha;   // Atributo para la fecha (puede ser fecha de nacimiento o inscripción)
    String codigo;
    String telefono;
```

```
    String carrera;
    String semestre;

    // Constructor con parámetros
    public Alumnos(String nombre, String fecha, String codigo, String telefono, String carrera,
String semestre) {
        this.nombre = nombre;
        this.fecha = fecha;
        this.codigo = codigo;
        this.telefono = telefono;
        this.carrera = carrera;
        this.semestre = semestre;
    }

    // Constructor vacío
    public Alumnos() {
    }

    // Getters y Setters
    public String getNombre() {
        return nombre;
    }

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

    public String getFecha() {
        return fecha;
    }

    public void setFecha(String fecha) {
        this.fecha = fecha;
    }

    public String getCodigo() {
        return codigo;
    }

    public void setCodigo(String codigo) {
        this.codigo = codigo;
    }

    public String getTelefono() {
        return telefono;
    }

    public void setTelefono(String telefono) {
```

```
        this.telefono = telefono;
    }

    public String getCarrera() {
        return carrera;
    }

    public void setCarrera(String carrera) {
        this.carrera = carrera;
    }

    public String getSemestre() {
        return semestre;
    }

    public void setSemestre(String semestre) {
        this.semestre = semestre;
    }


    // Método toString para mostrar la información del alumno
    @Override
    public String toString() {
        return "Alumnos{" + "nombre=" + nombre + ", fecha=" + fecha + ", codigo=" + codigo +
", telefono=" + telefono + ", carrera=" + carrera + ", semestre=" + semestre + '}';
    }

}
```

Arrayslit

```
/*
 * 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 Registro;

/**
 *
 * @author USER 17
 */
public class Alumnos {
    String nombre;
    String fecha;   // Atributo para la fecha (puede ser fecha de nacimiento o inscripción)
    String codigo;
    String telefono;
```

```
    String carrera;
    String semestre;

    // Constructor con parámetros
    public Alumnos(String nombre, String fecha, String codigo, String telefono, String carrera,
String semestre) {
        this.nombre = nombre;
        this.fecha = fecha;
        this.codigo = codigo;
        this.telefono = telefono;
        this.carrera = carrera;
        this.semestre = semestre;
    }

    // Constructor vacío
    public Alumnos() {
    }

    // Getters y Setters
    public String getNombre() {
        return nombre;
    }

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

    public String getFecha() {
        return fecha;
    }

    public void setFecha(String fecha) {
        this.fecha = fecha;
    }

    public String getCodigo() {
        return codigo;
    }

    public void setCodigo(String codigo) {
        this.codigo = codigo;
    }

    public String getTelefono() {
        return telefono;
    }

    public void setTelefono(String telefono) {
```

```
        this.telefono = telefono;
    }

    public String getCarrera() {
        return carrera;
    }

    public void setCarrera(String carrera) {
        this.carrera = carrera;
    }

    public String getSemestre() {
        return semestre;
    }

    public void setSemestre(String semestre) {
        this.semestre = semestre;
    }

    // Método toString para mostrar la información del alumno
    @Override
    public String toString() {
        return "Alumnos{" + "nombre=" + nombre + ", fecha=" + fecha + ", codigo=" + codigo +
", telefono=" + telefono + ", carrera=" + carrera + ", semestre=" + semestre + '}';
    }

}
```

Registro frame

```
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

/**
 *
 * @author USER 17
 */
public class JframeRegistro extends javax.swing.JFrame {

    // ArrayList para almacenar los registros de alumnos
    private ArrayList<Alumnos> listaAlumnos = new ArrayList<>();

    public JframeRegistro() {
        initComponents();
```

```
        this.setTitle("Registro de Alumnos");
        this.setLocationRelativeTo(null);
    }


 /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */

private void txtNombreActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void txtCodigoActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void btnAgregarActionPerformed(java.awt.event.ActionEvent evt) {
         // Obtener los valores de los campos de texto y combo box
    String nombre = txtNombre.getText();

    // Obtener la fecha desde JDateChooser
    Date fechaSeleccionada = Calendario.getDate();
    String fecha = "";  // Inicia la variable de fecha vacía
    if (fechaSeleccionada != null) {
        // Convertir la fecha a String utilizando SimpleDateFormat
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");  // Formato de fecha
        fecha = sdf.format(fechaSeleccionada);
    }

    String codigoEstudiante = txtCodigo.getText();
    String telefono = txtTelefono.getText();
    String carrera = jComboBoxCarrera.getSelectedItem().toString();
    String semestre = jComboBoxSemestre.getSelectedItem().toString();

    // Verificar si el registro ya existe (puedes usar el código de estudiante como identificador)
    int selectedRow = tblRegistro.getSelectedRow();
    if (selectedRow != -1) {
        // Actualizar los datos en el ArrayList
        Alumnos alumnoActualizado = listaAlumnos.get(selectedRow);
        alumnoActualizado.nombre = nombre;
        alumnoActualizado.fecha = fecha;
        alumnoActualizado.codigo = codigoEstudiante;
        alumnoActualizado.telefono = telefono;
        alumnoActualizado.carrera = carrera;
        alumnoActualizado.semestre = semestre;
```

```
        // Actualizar la tabla
        DefaultTableModel model = (DefaultTableModel) tblRegistro.getModel();
        model.setValueAt(nombre, selectedRow, 0);
        model.setValueAt(codigoEstudiante, selectedRow, 1);
        model.setValueAt(fecha, selectedRow, 2);
        model.setValueAt(telefono, selectedRow, 3);
        model.setValueAt(carrera, selectedRow, 4);
        model.setValueAt(semestre, selectedRow, 5);

        // Mostrar mensaje de éxito
        JOptionPane.showMessageDialog(this, "Registro actualizado con éxito");
    } else {
        // Crear un nuevo objeto Registro con los datos ingresados
        Alumnos nuevoRegistro = new Alumnos(nombre, fecha, codigoEstudiante, carrera,
semestre, telefono);

        // Agregar el nuevo registro a la lista
        listaAlumnos.add(nuevoRegistro);

        // Agregar el nuevo registro a la tabla
        DefaultTableModel model = (DefaultTableModel) tblRegistro.getModel();
        model.addRow(new Object[]{nombre, codigoEstudiante, fecha, telefono, carrera,
semestre});

        // Mostrar mensaje de éxito
        JOptionPane.showMessageDialog(this, "Registro agregado con éxito");
    }
       // Limpiar los campos después de actualizar
        txtNombre.setText("");
        Calendario.setDate(null);
        txtCodigo.setText("");
        txtTelefono.setText("");
        jComboBoxCarrera.setSelectedIndex(0);
        jComboBoxSemestre.setSelectedIndex(0);
    }

    private void btnBorrarActionPerformed(java.awt.event.ActionEvent evt) {
         // Obtener la fila seleccionada
    int selectedRow = tblRegistro.getSelectedRow();

    if (selectedRow != -1) {
        // Eliminar el registro de la lista
        listaAlumnos.remove(selectedRow);

        // Eliminar la fila seleccionada del modelo de la tabla
        DefaultTableModel model = (DefaultTableModel) tblRegistro.getModel();
        model.removeRow(selectedRow);
```

```
        // Mostrar mensaje de confirmación
        JOptionPane.showMessageDialog(this, "Registro borrado con éxito");
    } else {
        // Si no hay fila seleccionada, mostrar mensaje de error
        JOptionPane.showMessageDialog(this, "Por favor selecciona un registro para borrar.");
    }

    }

    private void btnBuscarActionPerformed(java.awt.event.ActionEvent evt) {
      // Obtener la letra inicial del campo de texto
    String letraInicial = txtBuscar.getText().trim().toUpperCase(); // Asegúrate de tener un
campo de texto para buscar

    // Limpiar la tabla antes de mostrar los resultados
    DefaultTableModel model = (DefaultTableModel) tblRegistro.getModel();
    model.setRowCount(0); // Limpiar la tabla

    // Si el campo de búsqueda está vacío, mostrar todos los registros
    if (letraInicial.isEmpty()) {
        for (Alumnos alumno : listaAlumnos) {
            model.addRow(new Object[]{
                alumno.nombre,
                alumno.codigo,
                alumno.fecha,
                alumno.telefono,
                alumno.carrera,
                alumno.semestre
            });
        }
    } else {
        // Filtrar y buscar en el ArrayList
        for (Alumnos alumno : listaAlumnos) {
            if (alumno.nombre.toUpperCase().startsWith(letraInicial)) {
                // Agregar el registro a la tabla
                model.addRow(new Object[]{
                    alumno.nombre,
                    alumno.codigo,
                    alumno.fecha,
                    alumno.telefono,
                    alumno.carrera,
                    alumno.semestre
                });
            }
        }

        // Mostrar mensaje si no se encontraron resultados
```

```
        if (model.getRowCount() == 0) {
            JOptionPane.showMessageDialog(this, "No se encontraron registros que empiecen
con la letra: " + letraInicial);
        }
    }
    }

    private void txtBuscarActionPerformed(java.awt.event.ActionEvent evt) {
      // Obtener la letra inicial del campo de texto
    String letraInicial = txtBuscar.getText().trim().toUpperCase(); // Asegúrate de tener un
campo de texto para buscar

    // Limpiar la tabla antes de mostrar los resultados
    DefaultTableModel model = (DefaultTableModel) tblRegistro.getModel();
    model.setRowCount(0); // Limpiar la tabla

    // Si el campo de búsqueda está vacío, mostrar todos los registros
    if (letraInicial.isEmpty()) {
        for (Alumnos alumno : listaAlumnos) {
            model.addRow(new Object[]{
                alumno.nombre,
                alumno.codigo,
                alumno.fecha,
                alumno.telefono,
                alumno.carrera,
                alumno.semestre
            });
        }
    } else if (letraInicial.length() == 1) {
        // Filtrar y buscar en el ArrayList
        for (Alumnos alumno : listaAlumnos) {
            if (alumno.nombre.toUpperCase().startsWith(letraInicial)) {
                // Agregar el registro a la tabla
                model.addRow(new Object[]{
                    alumno.nombre,
                    alumno.codigo,
                    alumno.fecha,
                    alumno.telefono,
                    alumno.carrera,
                    alumno.semestre
                });
            }
        }

        // Mostrar mensaje si no se encontraron resultados
        if (model.getRowCount() == 0) {
            JOptionPane.showMessageDialog(this, "No se encontraron registros que empiecen
con la letra: " + letraInicial);
```

```
        }
    } else {
        JOptionPane.showMessageDialog(this, "Por favor ingresa una letra inicial válida.");
    }
    }

    private void btnActualizarActionPerformed(java.awt.event.ActionEvent evt) {
 // Obtener la fila seleccionada
    int filaSeleccionada = tblRegistro.getSelectedRow();

    // Verificar si hay una fila seleccionada
    if (filaSeleccionada != -1) {
        // Obtener los valores actuales de la fila seleccionada
        String nombre = (String) tblRegistro.getValueAt(filaSeleccionada, 0);
        String codigo = (String) tblRegistro.getValueAt(filaSeleccionada, 1);
        String fecha = (String) tblRegistro.getValueAt(filaSeleccionada, 2);
        String telefono = (String) tblRegistro.getValueAt(filaSeleccionada, 3);
        String carrera = (String) tblRegistro.getValueAt(filaSeleccionada, 4);
        String semestre = (String) tblRegistro.getValueAt(filaSeleccionada, 5);

        // Crear una instancia de Alumnos con los datos actuales
        Alumnos alumno = new Alumnos(nombre, fecha, codigo, telefono, carrera, semestre);

        // Llenar los campos del formulario con los datos del JframeRegistro
        llenarCampos(alumno);
    } else {
        JOptionPane.showMessageDialog(this, "Por favor selecciona un registro para
actualizar.");
    }

    }

    private void txtTelefonoActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }
private void llenarCampos(Alumnos alumno) {
    txtNombre.setText(alumno.getNombre());
    txtCodigo.setText(alumno.getCodigo()); // Permitir modificación
    txtTelefono.setText(alumno.getTelefono());
    jComboBoxCarrera.setSelectedItem(alumno.getCarrera());
    jComboBoxSemestre.setSelectedItem(alumno.getSemestre());

    // No actualizamos el campo de fecha directamente, ya que requiere conversión
    // Si se desea, descomentar la siguiente línea para actualizar el JDateChooser:
    // Calendario.setDate(java.sql.Date.valueOf(JframeRegistro.getFecha()));
}
    /**
     * @param args the command line arguments
```

```
     */
    public static void main(String args[]) {
        /* Set the HiFi look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
        /* If HiFi (part of JTattoo) is not available, stay with the default look and feel. */
        try {
            // Set HiFi Look and Feel from JTattoo
            javax.swing.UIManager.setLookAndFeel("com.jtattoo.plaf.hifi.HiFiLookAndFeel");
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(JframeRegistro.class.getName()).log(java.util.logging.Lev
el.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(JframeRegistro.class.getName()).log(java.util.logging.Lev
el.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(JframeRegistro.class.getName()).log(java.util.logging.Lev
el.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(JframeRegistro.class.getName()).log(java.util.logging.Lev
el.SEVERE, null, ex);
        }
        //</editor-fold>
        //</editor-fold>
        //</editor-fold>
        //</editor-fold>
        //</editor-fold>
        //</editor-fold>
        //</editor-fold>
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new JframeRegistro().setVisible(true);
            }
        });
    }


    // Variables declaration - do not modify
    private com.toedter.calendar.JDateChooser Calendario;
    private javax.swing.JButton btnActualizar;
    private javax.swing.JButton btnAgregar;
    private javax.swing.JButton btnBorrar;
```

```
    private javax.swing.JButton btnBuscar;
    private javax.swing.JLabel jCarrera;
    private javax.swing.JLabel jCodigo;
    private javax.swing.JComboBox<String> jComboBoxCarrera;
    private javax.swing.JComboBox<String> jComboBoxSemestre;
    private javax.swing.JLabel jFecha;
    private javax.swing.JLabel jNombre;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JLabel jSemestre;
    private javax.swing.JLabel jTelefono;
    private javax.swing.JLabel jTitulo;
    private javax.swing.JTable tblRegistro;
    private javax.swing.JTextField txtBuscar;
    private javax.swing.JTextField txtCodigo;
    private javax.swing.JTextField txtNombre;
    private javax.swing.JTextField txtTelefono;
    // End of variables declaration
}
```