UNIVERSIDAD DE GUAYAQUIL

FACULTAD DE INGENIERIA INDUSTRIAL

INGENIERIA EN SISTEMAS DE INFORMACION

MATERIA:

PROGRAMACION

ESTUDIANTE:

ZAMBRANO VASQUEZ ANA MICHELLE

TEMA

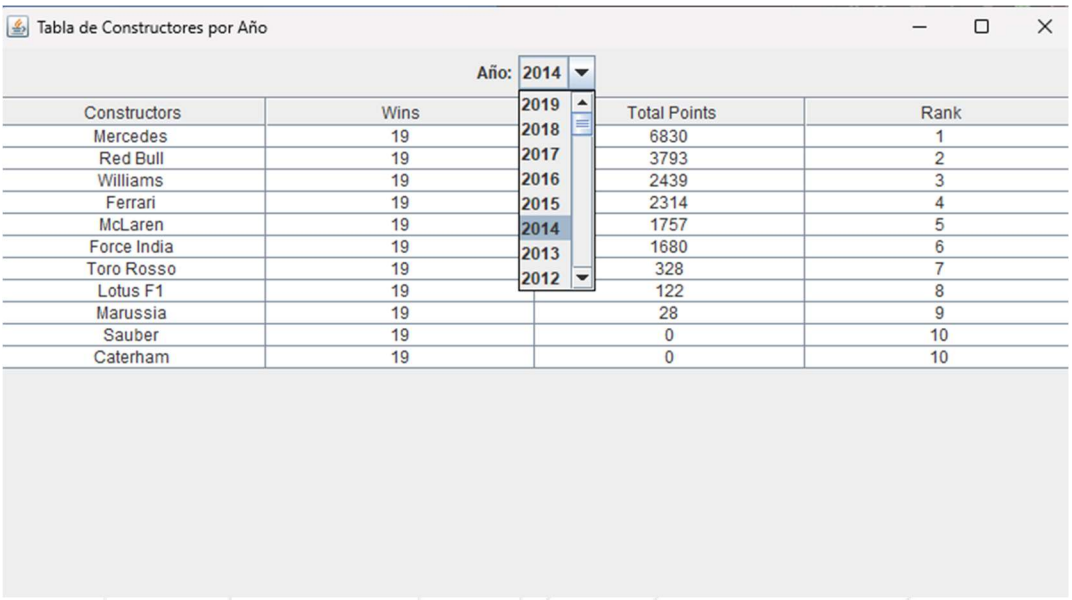
TALLER GRUPAL

```
package InterfazGrafica;

import javax.swing.*;
import javax.swing.table.DefaultTableCellRenderer;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.sql.*;
import java.util.Vector;

public class InterfazGrafica {
    private Connection conn;
    private JFrame frame;
    private JComboBox<String> comboBox;
    private JTable table;
    private DefaultTableModel tableModel;
    private JProgressBar progressBar;

    public InterfazGrafica() {
        // Establecer conexión a la base de datos PostgreSQL
        connectDB();

        // Crear la interfaz gráfica
        createGUI();
    }

    private void connectDB() {
        try {
            String url = "jdbc:postgresql://localhost:5432/Formula1";
            String user = "postgres";
            String password = "password";
            conn = DriverManager.getConnection(url, user, password);
            System.out.println("Conexión establecida con
PostgreSQL.");
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    private void createGUI() {
        frame = new JFrame("Tabla de Constructores por Año");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

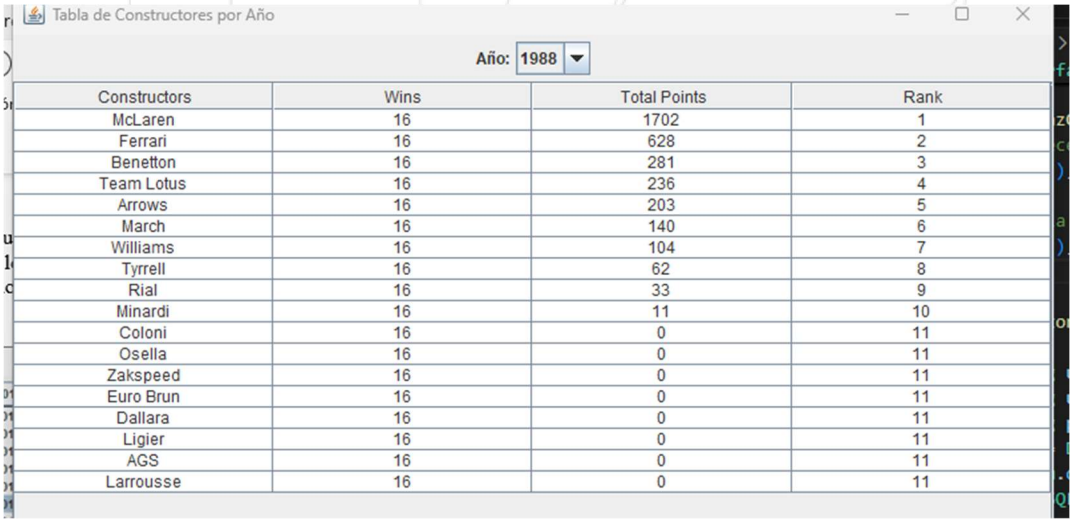
```
        frame.setSize(800, 600);

        // Panel superior con JComboBox para seleccionar el año
        JPanel topPanel = new JPanel();
        topPanel.add(new JLabel("Año:"));
        comboBox = new JComboBox<>();
        populateComboBox();
        comboBox.addActionListener(e -> updateTableInBackground());
        topPanel.add(comboBox);

        // Tabla para mostrar los datos de constructores
        tableModel = new DefaultTableModel();
        table = new JTable(tableModel);
        JScrollPane scrollPane = new JScrollPane(table);

        // Centrar el contenido de las celdas
        DefaultTableCellRenderer centerRenderer = new
DefaultTableCellRenderer();
        centerRenderer.setHorizontalAlignment(JLabel.CENTER);
        table.setDefaultRenderer(Object.class, centerRenderer);

        // Añadir componentes al frame
        frame.getContentPane().setLayout(new BorderLayout());
        frame.getContentPane().add(topPanel, BorderLayout.NORTH);
        frame.getContentPane().add(scrollPane, BorderLayout.CENTER);

        frame.setVisible(true);
    }

    private void populateComboBox() {
        try {
            Statement stmt = conn.createStatement();
            ResultSet rs = stmt.executeQuery("SELECT DISTINCT year
FROM races ORDER BY year DESC");
            while (rs.next()) {
                comboBox.addItem(rs.getString("year"));
            }
            rs.close();
            stmt.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
```

```
    private void updateTableInBackground() {
        String selectedYear = (String) comboBox.getSelectedItem();
        if (selectedYear != null) {
            // Crear un SwingWorker para ejecutar la consulta en
segundo plano
            SwingWorker<Void, Void> worker = new SwingWorker<Void,
Void>() {
                @Override
                protected Void doInBackground() throws Exception {
                    try {
                        // Consulta para obtener los datos de
constructores del año seleccionado
                        String query = "SELECT c.name AS
constructor_name, " +
                                       "COUNT(r.race_id) AS wins, " +
                                       "SUM(cs.points) AS total_points, " +
                                       "RANK() OVER (ORDER BY SUM(cs.points)
DESC) AS rank " +
                                       "FROM constructors c " +
                                       "JOIN constructor_standings cs ON
c.constructor_id = cs.constructor_id " +
                                       "JOIN races r ON cs.race_id =
r.race_id " +
                                       "WHERE r.year = ? " +
                                       "GROUP BY c.name " +
                                       "ORDER BY rank";

                        PreparedStatement pstmt =
conn.prepareStatement(query);
                        int year = Integer.parseInt(selectedYear);
                        pstmt.setInt(1, year);
                        ResultSet rs = pstmt.executeQuery();

                        // Obtener columnas
                        Vector<String> columnNames = new Vector<>();
                        columnNames.add("Constructors");
                        columnNames.add("Wins");
                        columnNames.add("Total Points");
                        columnNames.add("Rank");

                        // Obtener filas
                        Vector<Vector<Object>> data = new Vector<>();
```

```
                        while (rs.next()) {
                            Vector<Object> row = new Vector<>();
                            row.add(rs.getString("constructor_name"))
;
                            row.add(rs.getInt("wins"));
                            row.add(rs.getInt("total_points"));
                            row.add(rs.getInt("rank"));
                            data.add(row);
                        }

                        // Actualizar modelo de la tabla en el hilo
de eventos de Swing
                        SwingUtilities.invokeLater(() ->
tableModel.setDataVector(data, columnNames));

                        rs.close();
                        pstmt.close();
                    } catch (SQLException e) {
                        e.printStackTrace();
                    }
                    return null;
                }

                @Override
                protected void done() {
                    // Aquí puedes realizar acciones adicionales
después de que se completa la carga de datos
                }
            };

            // Iniciar el SwingWorker
            worker.execute();
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(InterfazGrafica::new);
    }
}
```



Constructors	Wins	Total Points	Rank
Mercedes	19	6830	1
Red Bull	19	3793	2
Williams	19	2439	3
Ferrari	19	2314	4
McLaren	19	1757	5
Force India	19	1680	6
Toro Rosso	19	328	7
Lotus F1	19	122	8
Marussia	19	28	9
Sauber	19	0	10
Caterham	19	0	10



Constructors	Wins	Total Points	Rank
McLaren	16	1702	1
Ferrari	16	628	2
Benetton	16	281	3
Team Lotus	16	236	4
Arrows	16	203	5
March	16	140	6
Williams	16	104	7
Tyrrell	16	62	8
Rial	16	33	9
Minardi	16	11	10
Coloni	16	0	11
Osella	16	0	11
Zakspeed	16	0	11
Euro Brun	16	0	11
Dallara	16	0	11
Ligier	16	0	11
AGS	16	0	11
Larrousse	16	0	11