**Texto

Descripción generada automáticamente con confianza media**

**Examen A C.O. de Programación (21-22) – Descripción funcional:**

* Se debe realizar una aplicación siguiendo el **patrón MVC** para consultar y registrar resultados de Eurovisión’22.

La aplicación manejará una base de datos EUROVISION\_22 con una tabla con la siguiente estructura:

CREATE TABLE PARTICIPANTES (

PAIS TEXT PRIMARY KEY,

INTERPRETE TEXT NOT NULL,

CANCION TEXT NOT NULL,

PUNTOS\_JURADO INTEGER DEFAULT 0,

PUNTOS\_PUBLICO INTEGER DEFAULT 0,

PUNTOS INTEGER DEFAULT 0

);

Crea las carpetas, estructura de paquetes y clases necesarias para darle la siguiente funcionalidad:

* La ventana principal debe ser algo así:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

1. Al seleccionar la opción de menú **Salir**, se mostrará un mensaje de confirmación preguntando si realmente se quiere cerrar la aplicación.
2. Al seleccionar la opción de menú **Consulta de Resultados** se mostrará el siguiente panel dentro de la ventana principal:

Tabla

Descripción generada automáticamente

* + La columna posición no es un dato que proceda de la base de datos, sino que será la posición que ocupe cada canción/país en función de los puntos. En la posición 1 deberá aparecer la canción con más puntos.

1. Al seleccionar la opción de menú **Registrar Resultado** se mostrará el siguiente panel dentro de la ventana principal:

Interfaz de usuario gráfica, Tabla

Descripción generada automáticamente

* + **Cada combo deberá estar cargado con los países de la tabla PARTICIPANTES** y en primer lugar la opción “ – “
  + El usuario deberá seleccionar un país en cada combo y pulsar el botón Guardar.
  + Al pulsar el botón Guardar se deberá validar que ningún combo tenga seleccionada la opción “ – “, ni que haya otro combo con el mismo país. Si ocurre una de estas dos situaciones, deberá mostrarse un mensaje de error indicando lo correspondiente.
  + Al pulsar el botón Guardar, si se han seleccionado los datos correctamente se almacenarán en la tabla PARTICIPANTES y se mostrará un mensaje indicando el resultado.
  + Si se pulsa el botón Limpiar todos los combos volverán a tener la opción “ – “ seleccionada.

Interfaz de usuario gráfica

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica

Descripción generada automáticamente

Una vez registrado algún resultado, al volver a la consulta se debería apreciar el cambio de posiciones según los puntos.

Tabla

Descripción generada automáticamente

Rúbrica:

|  |  |
| --- | --- |
| Clase Participante | 0,3 puntos |
| Clase AccesoDB (con properties) | 0,2 puntos |
| Clase ParticipantesPersistencia | 2,4 puntos |
| Clase VPrincipal | 0,4 puntos |
| Clase PConsulta | 1,3 puntos |
| Clase PRegistrar | 2,2 puntos |
| Clase EurovisionControlador | 1,3 puntos |
| Clase EurovisionInicio | 0,4 puntos |
| Funcionalidad | 1,5 puntos |

Se restarán puntos por:

* No crear la estructura de paquetes para implementar el patrón MVC.
* No nombrar a los elementos según las convenciones establecidas.
* No tabular el código.
* No refactorizar.

Ayuda para el diseño:

ALTO = 650

ANCHO = 650

Código de ayuda para el panel de Registrar:

JLabel lblRegistrar = new JLabel("Registrar Puntos Jurado");

lblRegistrar.setFont(new Font("Tahoma", Font.BOLD, 24));

lblRegistrar.setBounds(15, 21, 372, 39);

add(lblRegistrar);

JLabel lbl1Punto = new JLabel("1 Punto para:");

lbl1Punto.setFont(new Font("Tahoma", Font.PLAIN, 16));

lbl1Punto.setBounds(60, 85, 123, 20);

add(lbl1Punto);

JLabel lbl2Punto = new JLabel("2 Puntos para:");

lbl2Punto.setFont(new Font("Tahoma", Font.PLAIN, 16));

lbl2Punto.setBounds(60, 126, 123, 20);

add(lbl2Punto);

JLabel lbl3Punto = new JLabel("3 Puntos para:");

lbl3Punto.setFont(new Font("Tahoma", Font.PLAIN, 16));

lbl3Punto.setBounds(60, 167, 123, 20);

add(lbl3Punto);

JLabel lbl4Punto = new JLabel("4 Puntos para:");

lbl4Punto.setFont(new Font("Tahoma", Font.PLAIN, 16));

lbl4Punto.setBounds(60, 208, 123, 20);

add(lbl4Punto);

JLabel lbl5Punto = new JLabel("5 Puntos para:");

lbl5Punto.setFont(new Font("Tahoma", Font.PLAIN, 16));

lbl5Punto.setBounds(60, 249, 123, 20);

add(lbl5Punto);

JLabel lbl6Punto = new JLabel("6 Puntos para:");

lbl6Punto.setFont(new Font("Tahoma", Font.PLAIN, 16));

lbl6Punto.setBounds(60, 290, 123, 20);

add(lbl6Punto);

JLabel lbl7Punto = new JLabel("7 Puntos para:");

lbl7Punto.setFont(new Font("Tahoma", Font.PLAIN, 16));

lbl7Punto.setBounds(60, 331, 123, 20);

add(lbl7Punto);

JLabel lbl8Punto = new JLabel("8 Puntos para:");

lbl8Punto.setFont(new Font("Tahoma", Font.PLAIN, 16));

lbl8Punto.setBounds(60, 372, 123, 20);

add(lbl8Punto);

JLabel lbl10Punto = new JLabel("10 Puntos para:");

lbl10Punto.setFont(new Font("Tahoma", Font.PLAIN, 16));

lbl10Punto.setBounds(60, 413, 123, 20);

add(lbl10Punto);

JLabel lbl12Punto = new JLabel("12 Puntos para:");

lbl12Punto.setFont(new Font("Tahoma", Font.PLAIN, 16));

lbl12Punto.setBounds(60, 454, 123, 20);

add(lbl12Punto);

cmb1Punto = new JComboBox<String>();

cmb1Punto.setFont(new Font("Tahoma", Font.PLAIN, 14));

// TODO: añadir un model

cmb1Punto.setBounds(243, 85, 315, 26);

add(cmb1Punto);

cmb2Punto = new JComboBox<String>();

cmb2Punto.setFont(new Font("Tahoma", Font.PLAIN, 14));

// TODO: añadir un model

cmb2Punto.setBounds(243, 126, 315, 26);

add(cmb2Punto);

cmb3Punto = new JComboBox<String>();

cmb3Punto.setFont(new Font("Tahoma", Font.PLAIN, 14));

// TODO: añadir un model

cmb3Punto.setBounds(243, 167, 315, 26);

add(cmb3Punto);

cmb4Punto = new JComboBox<String>();

cmb4Punto.setFont(new Font("Tahoma", Font.PLAIN, 14));

// TODO: añadir un model

cmb4Punto.setBounds(243, 208, 315, 26);

add(cmb4Punto);

cmb5Punto = new JComboBox<String>();

cmb5Punto.setFont(new Font("Tahoma", Font.PLAIN, 14));

// TODO: añadir un model

cmb5Punto.setBounds(243, 249, 315, 26);

add(cmb5Punto);

cmb6Punto = new JComboBox<String>();

cmb6Punto.setFont(new Font("Tahoma", Font.PLAIN, 14));

// TODO: añadir un model

cmb6Punto.setBounds(243, 290, 315, 26);

add(cmb6Punto);

cmb7Punto = new JComboBox<String>();

cmb7Punto.setFont(new Font("Tahoma", Font.PLAIN, 14));

// TODO: añadir un model

cmb7Punto.setBounds(243, 331, 315, 26);

add(cmb7Punto);

cmb8Punto = new JComboBox<String>();

cmb8Punto.setFont(new Font("Tahoma", Font.PLAIN, 14));

// TODO: añadir un model

cmb8Punto.setBounds(243, 372, 315, 26);

add(cmb8Punto);

cmb10Punto = new JComboBox<String>();

cmb10Punto.setFont(new Font("Tahoma", Font.PLAIN, 14));

// TODO: añadir un model

cmb10Punto.setBounds(243, 413, 315, 26);

add(cmb10Punto);

cmb12Punto = new JComboBox<String>();

cmb12Punto.setFont(new Font("Tahoma", Font.PLAIN, 14));

// TODO: añadir un model

cmb12Punto.setBounds(243, 454, 315, 26);

add(cmb12Punto);

btnGuardarDatos = new JButton("Guardar Datos");

btnGuardarDatos.setFont(new Font("Tahoma", Font.PLAIN, 14));

btnGuardarDatos.setBounds(104, 519, 153, 29);

add(btnGuardarDatos);

btnLimpiarDatos = new JButton("Limpiar Datos");

btnLimpiarDatos.setFont(new Font("Tahoma", Font.PLAIN, 14));

btnLimpiarDatos.setBounds(361, 519, 153, 29);

add(btnLimpiarDatos);