

Navegación por el cuestionario

| | | | | | | | | | |
|----|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 10 | | | | | | | | | |

Finalizar revisión

| | |
|-----------------|------------------------------------|
| Comenzado el | sábado, 30 de marzo de 2019, 17:31 |
| Estado | Finalizado |
| Finalizado en | lunes, 1 de abril de 2019, 17:26 |
| Tiempo empleado | 1 día 22 horas |
| Calificación | 4,00 de 10,00 (40%) |

Información
 Marcar pregunta

**En la plataforma no se pueden ejecutar aplicaciones con interfaz gráfica.
No os guiéis por las correcciones porque lo único que hacen es comparar vuestro código con una posible solución.
Desarrollar y ejecutad el programa en jGrasp y comparad la solución con la planteada.
Para cualquier duda ya sabéis donde estoy.**

Pregunta 1
Correcta
Puntúa 1,00 sobre 1,00
 Marcar pregunta

Indica cuáles son tareas a realizar para manejar un evento en Java

Seleccione una o más de una:

- a. Desarrollar el método actionPerformed dentro de la interfaz correspondiente al evento
- b. Identificar la interfaz que necesita el evento que queremos manejar ✓
- c. Crear una clase oyente que implemente la interfaz correspondiente al evento que queremos manejar ✓
- d. Registrar un objeto de la clase Oyente en la ventana principal de la interfaz gráfica

Respuesta correcta

La respuesta correcta es: Crear una clase oyente que implemente la interfaz correspondiente al evento que queremos manejar, Identificar la interfaz que necesita el evento que queremos manejar

Correcta

Puntos para este envío: 1,00/1,00.

Pregunta 2
Correcta
Puntúa 1,00 sobre 1,00
 Marcar pregunta

Elije el número de métodos que se deben desarrollar en cada una de las siguientes interfaces:

- | | | |
|----------------|---|---|
| MouseListener | 5 |  |
| FocusListener | 2 |  |
| KeyListener | 3 |  |
| ActionListener | 1 |  |

Respuesta correcta

La respuesta correcta es: MouseListener – 5, FocusListener – 2, KeyListener – 3, ActionListener – 1

Correcta

Puntos para este envío: 1,00/1,00.

Pregunta 3
Correcta
Puntúa 1,00 sobre 1,00
 Marcar pregunta

Indica las afirmaciones que describen los eventos en Java

Seleccione una o más de una:

- a. Para gestionar un evento necesitamos al menos una clase que implemente la interfaz correspondiente a ese evento ✓
- b. Todos los eventos se gestionan con el método actionPerformed
- c. Cada tipo de evento se gestiona con unos métodos concretos diferentes ✓
- d. Cada evento se gestiona con un único método que será diferente dependiendo del tipo de evento

Respuesta correcta

La respuesta correcta es: Cada tipo de evento se gestiona con unos métodos concretos diferentes, Para gestionar un evento necesitamos al menos una clase que implemente la interfaz correspondiente a ese evento

Correcta

Puntos para este envío: 1,00/1,00.

Pregunta 4
Correcta
Puntúa 1,00 sobre 1,00
 Marcar pregunta

Indica los métodos que pertenecen a la interfaz FocusListener:

- a. focusGained ✓ focusLost
- b. focusLost ✓
- c. keyPressed
- d. mouseExited

Respuesta correcta

La respuesta correcta es: focusGained, focusLost

Correcta

Puntos para este envío: 1,00/1,00.

Pregunta 5
Incorrecta
Puntúa 0,00 sobre 1,00
 Marcar pregunta

Corrige el código que se plantea para que la ventana que se muestra sea interactiva:

Una vez introducida una temperatura, cuando se pulse el botón, se convertirá a grados fahrenheit y se mostrará de la siguiente manera:

Se asume que el dato que se introduce es válido. Será un número decimal y se utilizará el punto para indicar los decimales.

Respuesta: (penalty regime: 10, 20, ... %)

Reiniciar respuesta

```
1 import java.awt.*;           // Componentes de dibujo
2 import javax.swing.*;         // Componentes de una interfaz gráfica
3 import java.awt.event.*;       // Eventos
4
5 public class ClaseGuiEventos {
6
7     // Se puede incluir el main en la clase
8     public static void main(String[] args) {
9         ClaseGuiEventos gui = new ClaseGuiEventos();
10    }
11
12     // Atributos
13     private JFrame ventana;
14     private JButton boton;
15     private JLabel resultado;
16     private JTextField cajaCelsius;
17
18     // Constructor
19     public ClaseGuiEventos() {
20     }
```

| Expected | Got |
|--|--|
| <pre>✗ // Clase para la interfaz gráfica de nuestro programa import java.awt.*; // Componentes de dibujo import javax.swing.*; // Componentes de una interfaz gráfica import java.awt.event.*; // Eventos public class ClaseGuiEventos { // Se puede incluir el main en la clase public static void main(String[] args) { ClaseGuiEventos gui = new ClaseGuiEventos(); } // Atributos private JFrame ventana; private JButton boton; private JLabel resultado; private JTextField cajaCelsius; // Constructor public ClaseGuiEventos() { // Crea y configura la ventana principal ventana = new JFrame(); ventana.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); ventana.setLocation(400, 300); ventana.setSize(400, 200); ventana.setTitle("EJERCICIO 1"); ventana.setLayout(new FlowLayout()); // Crea y configura los componentes ventana.add(new JLabel("Celsius")); cajaCelsius = new JTextField(10); ventana.add(cajaCelsius); boton = new JButton(); boton.setText("CONVERTIR"); boton.addActionListener(pulsarBoton); // REGISTRAR UN OBJETO DE LA CLASE OYENTE EN EL BOTON Oyente pulsarBoton = new Oyente(); boton.addActionListener(pulsarBoton); ventana.add(boton); resultado = new JLabel("Temperatura en grados Fahrenheit"); ventana.add(resultado); // Al final. Cuando se haya creado todo se visualiza ventana.setVisible(true); } // Crear la clase Oyente // Se crea una clase que tiene el comportamiento del interfaz ActionListener // Se reescribe el método actionPerformed con el código que queremos que se ejecute cuando se pulse el componente private class Oyente implements ActionListener { // Si ocurre el evento ActionEvent lee la temperatura en grados Celsius de la caja de texto cajaCelsius // La convierte a grados Fahrenheit // Muestra el valor obtenido en la etiqueta resultado @Override public void actionPerformed(ActionEvent event) { String celsiusTexto = cajaCelsius.getText(); double celsius = Double.parseDouble(celsiusTexto); double fahrenheit = celsius * 9 / 5 + 32; resultado.setText("Temperatura en grados Fahrenheit: " + fahrenheit); } } }</pre> | <pre>***Error*** Traceback (most recent call last): File "prog.python3", line 1, in <module> comment = "Expected " IndexError: tuple index o</pre> |

Your code must pass all tests to earn any marks. Try again.

Question author's solution:

```
// Clase para la interfaz gráfica de nuestro programa

import java.awt.*;           // Componentes de dibujo
import javax.swing.*;         // Componentes de una interfaz gráfica

public class ClaseGuiEventos {

    // Se puede incluir el main en la clase
    public static void main(String[] args) {
        ClaseGuiEventos gui = new ClaseGuiEventos();
    }

    // Atributos
    private JFrame ventana;
    private JButton boton;
    private JLabel resultado;
    private JTextField cajaCelsius;

    // Constructor
    public ClaseGuiEventos() {

        // Crea y configura la ventana principal
        ventana = new JFrame();
        ventana.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        ventana.setLocation(400, 300);
        ventana.setSize(400, 200);
        ventana.setTitle("EJERCICIO 1");
        ventana.setLayout(new FlowLayout());

        // Crea y configura los componentes
        ventana.add(new JLabel("Celsius"));
        cajaCelsius = new JTextField(10);
        ventana.add(cajaCelsius);
        boton = new JButton();
        boton.setText("CONVERTIR");

        // REGISTRAR UN OBJETO DE LA CLASE OYENTE EN EL BOTON
        Oyente pulsarBoton = new Oyente();
        boton.addActionListener(pulsarBoton);

        ventana.add(boton);
        resultado = new JLabel("Temperatura en grados Fahrenheit");
        ventana.add(resultado);

        // Al final. Cuando se haya creado todo se visualiza
        ventana.setVisible(true);
    }

    // Crear la clase Oyente
    // Se crea una clase que tiene el comportamiento del interfaz ActionListener
    // Se reescribe el método actionPerformed con el código que queremos que se ejecute cuando se pulse el componente
    private class Oyente implements ActionListener {

        // Si ocurre el evento ActionEvent lee la temperatura en grados Celsius de la caja de texto cajaCelsius
        // La convierte a grados Fahrenheit
        // Muestra el valor obtenido en la etiqueta resultado
        @Override
        public void actionPerformed(ActionEvent event) {
            String celsiusTexto = cajaCelsius.getText();
            double celsius = Double.parseDouble(celsiusTexto);
            double fahrenheit = celsius * 9 / 5 + 32;
            resultado.setText("Temperatura en grados Fahrenheit: " + fahrenheit);
        }
    }
}
```

```

boton.addActionListener(pulsarBoton);

ventana.add(botón);
resultado = new JLabel("Temperatura en grados Fahrenheit");
ventana.add(resultado);

// Al final. Cuando se haya creado todo se visualiza
ventana.setVisible(true);
}

// Crear la clase Oyente
// Se crea una clase que tiene el comportamiento del interfaz ActionListener
// Se reescribe el método actionPerformed con el código que queremos que se ejecute cuando se pulse el componente
private class Oyente implements ActionListener {

    // Si ocurre el evento ActionEvent lee la temperatura en grados Celsius de la caja de texto cajaCelsius
    // La convierte a grados Fahrenheit
    // Muestra el valor obtenido en la etiqueta resultado
    @Override
    public void actionPerformed(ActionEvent evento) {
        String celsiusTexto = cajaCelsius.getText();
        double celsius = Double.parseDouble(celsiusTexto);
        double fahrenheit = celsius * 9 / 5 + 32;
        resultado.setText("Temperatura en grados Fahrenheit: " + fahrenheit);
    }
}
}

```

incorrecta

Puntos para este envío: 0,00/1,00.

Pregunta 6

Incorrecta

Puntuación 0,00 sobre
1,00

▼ Marcar
pregunta

Corrige el código que se plantea para que la ventana que se muestra sea interactiva:

Cuando se pulse un botón, se mostrará un mensaje indicando el botón que se ha pulsado de la siguiente manera:

Respuesta: (penalty regime: 10, 20, ... %)

[Reiniciar respuesta](#)

```

1 import java.awt.*;           // Componentes de dibujo
2 import javax.swing.*;         // Componentes de una interfaz gráfica
3 import java.awt.event.*;       // Eventos
4
5
6 public class ClaseGuiEventos06 implements ActionListener {
7
8     // Se puede incluir el main en la clase
9     public static void main(String[] args) {
10         String[] opciones = {"AÑADIR", "MOSTRAR", "BORRAR", "SALIR"};
11
12         ClaseGuiEventos06 gui = new ClaseGuiEventos06(opciones);
13     }
14
15     // Constructor
16     public ClaseGuiEventos06(String[] botones) {
17
18         // Crea y configura la ventana principal
19         JFrame ventana = new JFrame();
20         ventana.setTitle("EJERCICIO 3");

```

| Expected | Got |
|---|---|
| <pre> X import java.awt.*; // Componentes de dibujo import javax.swing.*; // Componentes de una interfaz gráfica import java.awt.event.*; // Eventos public class ClaseGuiEventos implements ActionListener { // Se puede incluir el main en la clase public static void main(String[] args) { String[] opciones = {"AÑADIR", "MOSTRAR", "BORRAR", "SALIR"}; ClaseGuiEventos gui = new ClaseGuiEventos(opciones); } // Constructor public ClaseGuiEventos(String[] botones) { // Crea y configura la ventana principal JFrame ventana = new JFrame(); ventana.setTitle("EJERCICIO 3"); ventana.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); ventana.setSize(300, 300); ventana.setLocation(800, 400); ventana.setLayout(new GridLayout(2, 2)); // Crea y configura los componentes for (int i = 0; i < botones.length; i++) { String texto = botones[i]; JButton boton = new JButton(texto); ventana.add(boton); // REGISTRAR UN OBJETO DE LA CLASE OYENTE EN TODOS LOS BOTONES boton.addActionListener(this); } // Al final. Cuando se haya creado todo se visualiza ventana.setVisible(true); } // Implementamos la interfaz ActionListener. Sobreescrivimos el método actionPerformed // Cuando se pulse un botón, se identificará el botón, se obtendrá su texto // y se abrirá un JOptionPane con el mensaje "Se ha pulsado textoBotón" @Override public void actionPerformed(ActionEvent evento) { Object objetoPulsado = evento.getSource(); JButton botonPulsado = (JButton) objetoPulsado; String texto = botonPulsado.getText(); JOptionPane.showMessageDialog(null, "Se ha pulsado " + texto); } } </pre> | <pre> ***Error*** Traceback (most recent call last): File "prog.py", line 108, in <module> comment = "Expected {1} lines, got {2}".format(l IndexError: tuple index out of range </pre> |

Question author's solution:

```

import java.awt.*;           // Componentes de dibujo
import javax.swing.*;         // Componentes de una interfaz gráfica
import java.awt.event.*;      // Eventos

public class ClaseGuiEventos implements ActionListener {

    // Se puede incluir el main en la clase
    public static void main(String[] args) {
        String[] opciones = {"AÑADIR", "MOSTRAR", "BORRAR", "SALIR"};

        ClaseGuiEventos gui = new ClaseGuiEventos(opciones);
    }

    // Constructor
    public ClaseGuiEventos(String[] botones) {

        // Crea y configura la ventana principal
        JFrame ventana = new JFrame();
        ventana.setTitle("EJERCICIO 3");
        ventana.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        ventana.setSize(300, 300);
        ventana.setLocation(800, 400);
        ventana.setLayout(new GridLayout(2, 2));

        // Crea y configura los componentes
        for (int i = 0; i < botones.length; i++) {
            String texto = botones[i];
            JButton boton = new JButton(texto);
            ventana.add(boton);

            // REGISTRAR UN OBJETO DE LA CLASE OYENTE EN TODOS LOS BOTONES
            boton.addActionListener(this);
        }

        // Al final. Cuando se haya creado todo se visualiza
        ventana.setVisible(true);
    }

    // Implementamos la interfaz ActionListener. Sobreescribimos el método actionPerformed
    // Cuando se pulse un botón, se identificará el botón, se obtendrá su texto
    // y se abrirá un JOptionPane con el mensaje "Se ha pulsado textoBotón"
    @Override
    public void actionPerformed(ActionEvent evento) {
        Object objetoPulsado = evento.getSource();
        JButton botonPulsado = (JButton) objetoPulsado;
        String texto = botonPulsado.getText();
        JOptionPane.showMessageDialog(null, "Se ha pulsado " + texto);
    }
}

```

Incorrecta

Puntos para este envío: 0,00/1,00.

Pregunta 7

Incorrecta

Puntúa 0,00 sobre
1,00▼ Marcar
pregunta

Corrige el código que plantea para que la ventana que se muestra sea interactiva:

Cuando una caja de texto obtenga el foco, su fondo cambiará al color indicado y cuando lo pierda volverá a ser blanco:

En este caso, se implementará la interfaz FocusListener con los siguientes métodos:

- focusGained
- focusLost

Para modificar el color de fondo de una caja de texto se utilizará el método setBackground(Color.GREEN). Los colores se identifican mediante las constantes de la clase Color. Puedes consultar más colores [aquí](#).

Respuesta: (penalty regime: 10, 20, ... %)

Reiniciar respuesta

```

1 import java.awt.*;           // Componentes de dibujo
2 import javax.swing.*;         // Componentes de una interfaz gráfica
3 import java.awt.event.*;      // Eventos
4 import java.util.*;
5 import java.io.*;

6
7 public class ClaseGuiEventos07 implements FocusListener {
8
9     // Se puede incluir el main en la clase
10    public static void main(String[] args) throws FileNotFoundException {
11        Scanner leerFichero = new Scanner(new File("colores.txt"));
12
13        ClaseGuiEventos07 gui = new ClaseGuiEventos07(leerFichero);
14    }
15
16
17    // Constructor
18    public ClaseGuiEventos07(Scanner leerDatos) {
19
20        // Crea y configura la ventana principal
21        JFrame ventana = new JFrame();

```

| Expected | Got |
|---|--|
| <pre> import java.awt.*; // Componentes de dibujo import javax.swing.*; // Componentes de una interfaz gráfica import java.awt.event.*; // Eventos import java.util.*; import java.io.*; public class ClaseGuiEventos implements FocusListener { </pre> | <pre> ***Error*** Traceback (most recent call last): File "prog.pyhon3", line 147, in <module> comment = "Expected {1} lines, got {2}" IndexError: tuple index out of range </pre> |

```

Expected
// Crea y configura los componentes
while (leerDatos.hasNext()) {
    String texto = leerDatos.next();
    JTextField cajaTexto = new JTextField(texto);
    ventana.add(cajaTexto);

    // REGISTRAR UN OBJETO DE LA CLASE OYENTE EN LAS CAJAS DE TEXTO
    cajaTexto.addActionListener(this);
}

// Al final. Cuando se haya creado todo se visualiza
ventana.setVisible(true);
}

// Desarrolla los métodos de la interfaz FocusListener

// Cuando una caja de texto gane el foco de la ventana se cambiará el color de fondo por el indicado
public void focusGained(FocusEvent evento) {
    Object objetoFocus = evento.getSource();
    JTextField cajaTextoFocus = (JTextField) objetoFocus;
    String texto = cajaTextoFocus.getText();
    if (texto.equals("ROJO")) {
        cajaTextoFocus.setBackground(Color.RED);
        return;
    }
    if (texto.equals("VERDE")) {
        cajaTextoFocus.setBackground(Color.GREEN);
        return;
    }
    if (texto.equals("AZUL")) {
        cajaTextoFocus.setBackground(Color.BLUE);
        return;
    }
}

// Cuando una caja de texto pierda el foco de la ventana su color de fondo volverá a ser blanco
public void focusLost(FocusEvent evento) {
    Object objetoFocus = evento.getSource();
    JTextField cajaTextoFocus = (JTextField) objetoFocus;
    String texto = cajaTextoFocus.getText();
    cajaTextoFocus.setBackground(Color.WHITE);
}
}

```

Your code must pass all tests to earn any marks. Try again.

Question author's solution:

```

import java.awt.*;           // Componentes de dibujo
import javax.swing.*;         // Componentes de una interfaz gráfica
import java.awt.event.*;       // Eventos
import java.util.*;
import java.io.*;

public class ClaseGuiEventos implements FocusListener {

    // Se puede incluir el main en la clase
    public static void main(String[] args) throws FileNotFoundException {
        Scanner leerFichero = new Scanner(new File("colores.txt"));

        ClaseGuiEventos gui = new ClaseGuiEventos(leerFichero);
    }

    // Constructor
    public ClaseGuiEventos(Scanner leerDatos) {

        // Crea y configura la ventana principal
        JFrame ventana = new JFrame();
        ventana.setTitle("EJERCICIO 4");
        ventana.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        ventana.setSize(300, 150);
        ventana.setLocation(600, 200);
        ventana.setLayout(new GridLayout(3, 1));

        // Crea y configura los componentes
        while (leerDatos.hasNext()) {
            String texto = leerDatos.next();
            JTextField cajaTexto = new JTextField(texto);
            ventana.add(cajaTexto);

            // REGISTRAR UN OBJETO DE LA CLASE OYENTE EN LAS CAJAS DE TEXTO
            cajaTexto.addActionListener(this);
        }

        // Al final. Cuando se haya creado todo se visualiza
        ventana.setVisible(true);
    }

    // Desarrolla los métodos de la interfaz FocusListener

    // Cuando una caja de texto gane el foco de la ventana se cambiará el color de fondo por el indicado
    public void focusGained(FocusEvent evento) {
        Object objetoFocus = evento.getSource();
        JTextField cajaTextoFocus = (JTextField) objetoFocus;
        String texto = cajaTextoFocus.getText();
        if (texto.equals("ROJO")) {
            cajaTextoFocus.setBackground(Color.RED);
            return;
        }
        if (texto.equals("VERDE")) {
            cajaTextoFocus.setBackground(Color.GREEN);
            return;
        }
        if (texto.equals("AZUL")) {
            cajaTextoFocus.setBackground(Color.BLUE);
            return;
        }
    }

    // Cuando una caja de texto pierda el foco de la ventana su color de fondo volverá a ser blanco
    public void focusLost(FocusEvent evento) {
        Object objetoFocus = evento.getSource();
        JTextField cajaTextoFocus = (JTextField) objetoFocus;
        String texto = cajaTextoFocus.getText();
        cajaTextoFocus.setBackground(Color.WHITE);
    }
}

```

Incorrecta

Puntos para este envío: 0,00/1,00.

Pregunta 8
Incorrecta
Puntuó 0,00 sobre
1,00
 Marcar pregunta

Corrige el código que se plantea para que la ventana que se muestra sea interactiva:

Cuando se pulse cualquier tecla se mostrará en el área de texto la tecla que se ha pulsado. Además, si se pulsa "1" se mostrará un JOptionPane.showMessageDialog:

Si se pulsa "K" se mostrará un JOptionPane.showInputDialog:



Si se pulsa "Ctrl" se mostrará un JOptionPane.showConfirmDialog:



Se usará la interfaz KeyListener. Solo habrá que desarrollar el método keyPressed.

Respuesta: (penalty regime: 10, 20, ... %)

[Reiniciar respuesta](#)

```
1 import javax.swing.* ;
2 import java.awt.event.*;
3 import java.awt.*;
4
5 // Definimos la clase EscuchaTeclas como subclase de JFrame
6 // Implementa la interfaz KeyListener
7 public class EscuchaTeclas extends JFrame implements KeyListener {
8
9     // Crear la interfaz gráfica y establece sus propiedades
10    public static void main(String[] args) {
11        EscuchaTeclas ventana = new EscuchaTeclas();
12        ventana.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE );
13        ventana.setTitle("Ejercicio 08");
14        ventana.setSize(360, 120);
15        ventana.setVisible(true);
16    }
17
18    // Atributos
19    private JTextArea areaTexto;
20}
```

| Expected | Got |
|--|---|
| <pre>✖ import javax.swing.*; import java.awt.event.*; import java.awt.*; // Definimos la clase EscuchaTeclas como subclase de JFrame // Implementa la interfaz KeyListener public class EscuchaTeclas extends JFrame implements KeyListener { // Crear la interfaz gráfica y establece sus propiedades public static void main(String[] args) { EscuchaTeclas ventana = new EscuchaTeclas(); ventana.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); ventana.setTitle("Eventos de teclado"); ventana.setSize(360, 120); ventana.setVisible(true); } // Atributos private JTextArea areaTexto; // Constructor de la clase public EscuchaTeclas () { // Crear objeto JTextArea areaTexto = new JTextArea(); areaTexto.setText("Pulsa cualquier tecla del teclado...\\n\\n "); areaTexto.setEnabled(false); // Creamos un Scroll para la caja de texto // El objeto principal es el JScrollPane y se le pasa el JTextArea JScrollPane scrollPane = new JScrollPane(areaTexto); // Añadir al JFrame el JScrollPane. Con eso también se añade el JTextArea // Necesitamos el método getContentPane() para obtener el contenedor del JFrame donde se añaden los componentes this.getContentPane().add(scrollPane); // Registrar el oyente en el JFrame this.addKeyListener(this); } // Implementar la clase oyente que implemente el interface KeyListener. Hay que desarrollar todos sus métodos. // Cuando se pulsa cualquier tecla, obtenemos el código de tecla y a partir de él su texto identificativo // Lo mostramos en el JTextArea. // Si pulsamos "I" mostramos JOptionPane.showMessageDialog // Si pulsamos "K" mostramos JOptionPane.showInputDialog // Si pulsamos "Ctrl" mostramos JOptionPane.showConfirmDialog public void keyPressed(KeyEvent evento) { int codigoTecla = evento.getKeyCode(); String textoTecla = evento.getText(codigoTecla); areaTexto.append(textoTecla + "\\n"); switch (textTecla) { case "I": JOptionPane.showMessageDialog(null, "Has pulsado la tecla I"); break; case "K": JOptionPane.showInputDialog(null, "Has pulsado la tecla K"); break; case "Ctrl": JOptionPane.showConfirmDialog(null, "Has pulsado la tecla CTRL"); break; } } // Definimos lo que hacer cuando se libera cualquier tecla public void keyReleased(KeyEvent evento) { } // manejar evento de pulsación de una tecla de acción public void keyTyped(KeyEvent evento) { } }</pre> | ***Error*** Traceback (most recent call last): File "prog.python3", line 1, in <module> comment = "Expected" IndexError: tuple index out of range |

Your code must pass all tests to earn any marks. Try again.

Question author's solution:

```
import javax.swing.*;
import java.awt.event.*;
import java.awt.*;

// Definimos la clase EscuchaTeclas como subclase de JFrame
// Implementa la interfaz KeyListener
public class EscuchaTeclas extends JFrame implements KeyListener {
```

```

// Crear la interfaz gráfica y establece sus propiedades
public static void main(String[] args) {
    EscuchaTeclas ventana = new EscuchaTeclas();
    ventana.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    ventana.setTitle("Eventos de teclado");
    ventana.setSize(360, 120);
    ventana.setVisible(true);
}

// Atributos
private JTextArea areaTexto;

// Constructor de la clase
public EscuchaTeclas () {

    // Crear objeto JTextArea
    areaTexto = new JTextArea();
    areaTexto.setText("Pulsa cualquier tecla del teclado...\n\n");
    areaTexto.setEnabled(false);

    // Creamos un Scroll para la caja de texto
    // El objeto principal es el JScrollPane y se le pasa el JTextArea
    JScrollPane scrollPane = new JScrollPane(areaTexto);

    // Añadir al JFrame el JScrollPane. Con eso también se añade el JTextArea
    // Necesitamos el método getContentPane() para obtener el contenedor del JFrame donde se añaden los componentes
    this.getContentPane().add(scrollPane);

    // Registrar el oyente en el JFrame
    this.addKeyListener(this);
}

// Implementar la clase oyente que implemente el interface KeyListener. Hay que desarrollar todos sus métodos.

// Cuando se pulsa cualquier tecla, obtenemos el código de tecla y a partir de él su texto identificativo
// Lo mostramos en el JTextArea.
// Si pulsamos "1" mostramos JOptionPane.showMessageDialog
// Si pulsamos "K" mostramos JOptionPane.showInputDialog
// Si pulsamos "Ctrl" mostramos JOptionPane.showConfirmDialog
public void keyPressed(KeyEvent evento) {
    int codigoTecla = evento.getKeyCode();
    String textoTecla = evento.getText(codigoTecla);
    areaTexto.append(textoTecla + "\n");
    switch (textoTecla) {
        case "1":
            JOptionPane.showMessageDialog(null, "Has pulsado el 1");
            break;
        case "K":
            JOptionPane.showInputDialog(null, "Has pulsado la tecla K");
            break;
        case "Ctrl":
            JOptionPane.showConfirmDialog(null, "Has pulsado la tecla CTRL");
            break;
    }
}

// Definimos lo que hacer cuando se libera cualquier tecla
public void keyReleased(KeyEvent evento) {
}

// manejar evento de pulsación de una tecla de acción
public void keyTyped(KeyEvent evento) {
}
}

```

Incorrecta

Puntos para este envío: 0,00/1,00.

Pregunta 9
Incorrecta
Puntúa 0,00 sobre
1,00
▼ Marcar
pregunta

Añade el código necesario para que la ventana que se muestra sea interactiva:

Cuando se pulse el botón se pedirá el radio de un círculo:

y se calculará el perímetro:

Respuesta: (penalty regime: 10, 20, ... %)

[Reiniciar respuesta](#)

```

1 import java.awt.*;           // Componentes de dibujo
2 import javax.swing.*;         // Componentes de una interfaz gráfica
3 import java.awt.event.*;       // Eventos
4
5 public class ClaseGuiEventos5 implements ActionListener {
6
7     // Se puede incluir el main en la clase
8     public static void main(String[] args) {
9
10        ClaseGuiEventos5 gui = new ClaseGuiEventos5();
11    }
12
13    // Constructor
14    public ClaseGuiEventos5() {
15
16        // Crea y configura la ventana principal
17        JFrame ventana = new JFrame();
18        ventana.setTitle("EJERCICIO");
19        ventana.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
20        ventana.setLocation(800, 400);

```

| Expected | Got |
|--|--|
| <pre> import java.awt.*; // Componentes de dibujo import javax.swing.*; // Componentes de una interfaz gráfica import java.awt.event.*; // Eventos public class ClaseGuiEventos5 implements ActionListener { // Se puede incluir el main en la clase public static void main(String[] args) { ClaseGuiEventos5 gui = new ClaseGuiEventos5(); } // Constructor public ClaseGuiEventos5() { // Crea y configura la ventana principal </pre> | <pre> ***Error *** Traceback (most recent call last): File "prog.python3", line 99, in <module> comment = "Expected {1} lines, got {2}". IndexError: tuple index out of range </pre> |

| Expected | Got |
|---|-----|
| <pre> JFrame ventana = new JFrame(); ventana.setTitle("EJERCICIO"); ventana.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); ventana.setLocation(800, 400); ventana.setLayout(new BorderLayout()); // Crea y configura los componentes JLabel etiqueta = new JLabel("Comprueba lo que hace el botón"); ventana.add(etiqueta, BorderLayout.NORTH); JButton boton = new JButton("PULSAR"); ventana.add(boton, BorderLayout.SOUTH); // REGISTRAR UN OBJETO DE LA CLASE oyente EN EL BOTON boton.addActionListener(this); // Al final. Cuando se haya creado todo se visualiza ventana.pack(); ventana.setVisible(true); } // IMPLEMENTAR LA INTERFAZ ActionListener. SOBREESCRIBIR EL METODO actionPerformed @Override public void actionPerformed(ActionEvent evento) { String radioTexto = JOptionPane.showInputDialog(null, "Introduce el radio de un círculo "); double radioDouble = Double.parseDouble(radioTexto); double perimetro = 2 * Math.PI * radioDouble; JOptionPane.showMessageDialog(null, String.format("Perímetro: %.1f",perimetro)); } } </pre> | |

Your code must pass all tests to earn any marks. Try again.

Question author's solution:

```

import java.awt.*;           // Componentes de dibujo
import javax.swing.*;         // Componentes de una interfaz gráfica
import java.awt.event.*;       // Eventos

public class ClaseGuiEventos5 implements ActionListener {

    // Se puede incluir el main en la clase
    public static void main(String[] args) {

        ClaseGuiEventos5 gui = new ClaseGuiEventos5();

    }

    // Constructor
    public ClaseGuiEventos5() {

        // Crea y configura la ventana principal
        JFrame ventana = new JFrame();
        ventana.setTitle("EJERCICIO");
        ventana.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        ventana.setLocation(800, 400);
        ventana.setLayout(new BorderLayout());

        // Crea y configura los componentes
        JLabel etiqueta = new JLabel("Comprueba lo que hace el botón");
        ventana.add(etiqueta, BorderLayout.NORTH);
        JButton boton = new JButton("PULSAR");
        ventana.add(boton, BorderLayout.SOUTH);

        // REGISTRAR UN OBJETO DE LA CLASE oyente EN EL BOTON
        boton.addActionListener(this);

        // Al final. Cuando se haya creado todo se visualiza
        ventana.pack();
        ventana.setVisible(true);
    }

    // IMPLEMENTAR LA INTERFAZ ActionListener. SOBREESCRIBIR EL METODO actionPerformed
    @Override
    public void actionPerformed(ActionEvent evento) {
        String radioTexto = JOptionPane.showInputDialog(null, "Introduce el radio de un círculo ");
        double radioDouble = Double.parseDouble(radioTexto);
        double perimetro = 2 * Math.PI * radioDouble;
        JOptionPane.showMessageDialog(null, String.format("Perímetro: %.1f",perimetro));
    }
}

```

Incorrecta

Puntos para este envío: 0.00/1.00.

Pregunta 10

Incorrecta

Puntúa 0.00 sobre
1.00

Marcar
pregunta

Añade el código necesario para que la ventana que se muestra sea interactiva:

Cuando se pulse el botón, recogerá el radio introducido, calculará el área y lo mostrará en la etiqueta "Área del círculo" de la siguiente manera:

Respuesta: (penalty regime: 10, 20, ... %)

Reiniciar respuesta

```

1 import java.awt.*;           // Componentes de dibujo
2 import javax.swing.*;         // Componentes de una interfaz gráfica
3 import java.awt.event.*;       // Eventos
4
5 public class ClaseGuiEventos {
6
7     // Se puede incluir el main en la clase
8     public static void main(String[] args) {
9
10    | ClaseGuiEventos gui = new ClaseGuiEventos();
11
12
13    // Atributos
14    private JFrame ventana;
15    private JButton boton;
16    private JLabel etiquetaArea;
17    private JTextField cajaRadio;
18
19    // Constructor
20    public ClaseGuiEventos() {

```

| Expected | Got |
|--|--|
| <pre> import java.awt.*; // Componentes de dibujo import javax.swing.*; // Componentes de una interfaz gráfica import java.awt.event.*; // Eventos public class ClaseGuiEventos { // Se puede incluir el main en la clase public static void main(String[] args) { ClaseGuiEventos gui = new ClaseGuiEventos(); // Atributos private JFrame ventana; private JButton boton; private JLabel etiquetaArea; private JTextField cajaRadio; // Constructor public ClaseGuiEventos() { // Crea y configura la ventana principal ventana = new JFrame(); ventana.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); ventana.setLocation(400, 300); ventana.setSize(300, 200); ventana.setTitle("EJERCICIO"); ventana.setLayout(new FlowLayout()); // Crea y configura los componentes ventana.add(new JLabel("ADIO")); cajaRadio = new JTextField(10); ventana.add(cajaRadio); boton = new JButton(); boton.setText("REA"); ventana.add(boton); etiquetaArea = new JLabel("rea del círculo"); ventana.add(etiquetaArea); // REGISTRAR UN OBJETO DE LA CLASE OYENTE EN EL BOTON Oyente pulsarBoton = new Oyente(); boton.addActionListener(pulsarBoton); // Al final. Cuando se haya creado todo se visualiza ventana.setVisible(true); } // CREAR LA CLASE OYENTE private class Oyente implements ActionListener { @Override public void actionPerformed(ActionEvent evento) { String radioTexto = cajaRadio.getText(); double radio = Double.parseDouble(radioTexto); double area = Math.PI * radio * radio; etiquetaArea.setText(String.format("rea: %.1f", area)); } } } } </pre> | <pre> ***Error*** Traceback (most recent call last): File "prog.pyt3", line 128, in <module> comment = "Expected {1} lines, got {2}".format(lines, len(got_lines)) IndexError: tuple index out of range </pre> |

Your code must pass all tests to earn any marks. Try again.

Question author's solution:

```

import java.awt.*;           // Componentes de dibujo
import javax.swing.*;        // Componentes de una interfaz gráfica
import java.awt.event.*;      // Eventos

public class ClaseGuiEventos {

    // Se puede incluir el main en la clase
    public static void main(String[] args) {

        ClaseGuiEventos gui = new ClaseGuiEventos();

        // Atributos
        private JFrame ventana;
        private JButton boton;
        private JLabel etiquetaArea;
        private JTextField cajaRadio;

        // Constructor
        public ClaseGuiEventos() {

            // Crea y configura la ventana principal
            ventana = new JFrame();
            ventana.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            ventana.setLocation(400, 300);
            ventana.setSize(300, 200);
            ventana.setTitle("EJERCICIO");
            ventana.setLayout(new FlowLayout());

            // Crea y configura los componentes
            ventana.add(new JLabel("ADIO"));
            cajaRadio = new JTextField(10);
            ventana.add(cajaRadio);
            boton = new JButton();
            boton.setText("REA");
            ventana.add(boton);
            etiquetaArea = new JLabel("rea del círculo");
            ventana.add(etiquetaArea);

            // REGISTRAR UN OBJETO DE LA CLASE OYENTE EN EL BOTON
            Oyente pulsarBoton = new Oyente();
            boton.addActionListener(pulsarBoton);

            // Al final. Cuando se haya creado todo se visualiza
            ventana.setVisible(true);
        }

        // CREAR LA CLASE OYENTE
        private class Oyente implements ActionListener {
            @Override
            public void actionPerformed(ActionEvent evento) {
                String radioTexto = cajaRadio.getText();
                double radio = Double.parseDouble(radioTexto);
                double area = Math.PI * radio * radio;
                etiquetaArea.setText(String.format("rea: %.1f", area));
            }
        }
    }
}

```

incorrecta

Puntos para este envío: 0,00/1,00.

