

*TEST: Respuesta correcta +4 puntos, incorrecta -1 punto, en blanco +0. Sólo hay una respuesta correcta*  
*Es obligatorio presentar documento de identificación*

**Apellidos y Nombre:**

**D.N.I :**

1. Las opciones de *Deshacer* y *Rehacer* en un menú:
  - (a) Alivian la carga de la memoria a largo plazo
  - (b) Alivian la carga de la memoria a corto plazo**
  - (c) Alivian la carga de la memoria sensorial
  - (d) Alivian las restricciones impuestas por el sistema visual humano
  - (e) Ninguna de las respuestas anteriores es correcta
2. En una opción de un menú los puntos suspensivos (...) se deben poner:
  - (a) Siempre que a partir de esta opción se vaya a desplegar otra ventana
  - (b) Cuando el título de la opción esté formado por más de dos palabras
  - (c) Cuando se requiera más información para poder completar la ejecución de dicha opción de menú**
  - (d) Siempre que la opción de menú tenga asociado un acelerador
  - (e) Ninguna de las respuestas anteriores es correcta
3. Para tratar de garantizar el principio de *Consistencia* referido a la usabilidad se debe:
  - (a) Notificar los cambios producidos en la interfaz de manera eventual
  - (b) Evitar cambiar las técnicas de interacción que el usuario ya conoce**
  - (c) Utilizar en la medida de lo posible diálogos no modales
  - (d) Más de una respuesta es cierta
  - (e) Ninguna de las respuestas anteriores es cierta
4. Respecto al botón por defecto de una ventana:
  - (a) Es el que responde a la pulsación de la tecla "Enter"**
  - (b) Es obligatorio incorporar botón por defecto pero no es obligatorio incluir botón de cancelación
  - (c) Lleva nemónico
  - (d) Más de una respuesta es cierta
  - (e) Ninguna de las respuestas anteriores es cierta
5. La clase *JOptionPane*:
  - (a) Proporciona un contenedor para poder presentar varias opciones al usuario
  - (b) Proporciona un método estático para presentar un diálogo modal al usuario
  - (c) Proporciona varios métodos estáticos para presentar diferentes diálogos modales al usuario**
  - (d) Proporciona un método estático para presentar un diálogo no modal al usuario
  - (e) Proporciona varios métodos estáticos para presentar diferentes diálogos no modales al usuario
6. Respecto a *Swing* y *AWT* es cierto que:
  - (a) Componentes como *spinners* o *sliders* se incluyen en *AWT* pero no en *Swing*
  - (b) *AWT* y *Swing* forman parte de *AJFC (Advanced Java Foundation Classes)*
  - (c) Para llevar a cabo gestión de eventos haremos uso de la librería *event* de *AWT***
  - (d) Más de una respuesta es cierta
  - (e) Ninguna de las respuestas anteriores es cierta
7. Un *JComboBox* editable:
  - (a) Permite ahorrar tiempo al usuario permitiéndole teclear directamente un valor a localizar en el combo**
  - (b) Podría ser sustituido por un conjunto de checkboxes múltiple
  - (c) Permite añadir nuevos ítems al combobox
  - (d) No existe en java la posibilidad de que un *JComboBox* sea editable
  - (e) Más de una respuesta es correcta

8. Para tratar de evitar la saturación de la memoria sensorial se debe:

- (a) Acudir al reconocimiento
- (b) Acudir a técnicas de ensayo
- (c) Trocear o partir la información
- (d) Evitar la habituación**
- (e) Más de una respuesta es correcta

```
1. public class VentanaPrincipal extends JFrame{
2.     private JTextArea area1 = null;
3.     private ProcesaTecla pT = null ;
4.     ...
5.     class ProcesaTecla implements KeyAdapter {
6.         public void keyTyped(KeyEvent e){
7.             pintar(e);
8.         }
9.     } // fin clase ProcesaTecla
10.    private void initialize(){
11.        pT = new ProcesaTecla();
12.        ...
13.    } // Fin del initialize
14.    private void getArea1{
15.        ...
16.        area1.addKeyListener(pT);
17.    ...}
```

9. Dado el código anterior y considerando que los métodos del interface *KeyListener* son más de dos:

- (a) *pT* representa el objeto fuente del evento y *area1* representa el objeto receptor del evento
- (b) La instrucción de la línea 16 es incorrecta ya que debería ser *area1.addKeyAdapter(pT)*;
- (c) La instrucción de la línea 5 es incorrecta ya que se debería utilizar *extends* en lugar de *implements***
- (d) La clase *ProcesaTecla* es incorrecta ya que se deberían haber implementado todos los métodos del interface *MouseAdapter*
- (e) Más de una respuesta es correcta

```
public void pintar (KeyEvent e){
    1. JTextArea area = e.getSource();
    2. area.setBackground(Color.red);
}
```

10. Supongamos que este es el código del método *pintar* de la pregunta anterior. Indica cual es la afirmación correcta:

- (a) Es un método correcto y se cambiará a color rojo el fondo del objeto fuente del evento
- (b) Es un método incorrecto porque faltan parámetros en el método *getSource* de la línea 1
- (c) Es un método incorrecto porque es necesario añadirle un parámetro que indique el área a pintar
- (d) Es un método incorrecto ya que en la línea 1 es necesario hacer *casting* a la clase *JTextArea***
- (e) Es un método correcto y se cambiará a color rojo el fondo del objeto receptor del evento