



Universidad  
Carlos III de Madrid

## Programación de Sistemas

### Grado en Ingeniería de Sistemas de Comunicaciones y Grado en Ingeniería Telemática

Leganés, 21 de marzo de 2014  
Duración de la prueba: 15 min

Examen parcial 1 (teoría)  
Puntuación: 5 puntos sobre 10 del examen

*Sólo una opción es correcta en cada pregunta. Cada respuesta correcta suma 1/2 puntos. Cada respuesta incorrecta resta 1/6 puntos. Las preguntas no contestadas no suman ni restan puntos.*

Marca:  Anula:  No uses:   

- Marca la respuesta a cada pregunta con una equis (“X”) en la tabla de abajo.
- Si marcas más de una opción o ninguna opción, la pregunta se considera no contestada.
- Rellena **tus datos personales** antes de comenzar a realizar el examen.

Nombre:

Grupo:

Firma:

NIA:

--	--	--	--	--	--	--	--	--	--

	A	B	C	D	A	B	C	D
1	<input type="checkbox"/>							
2	<input type="checkbox"/>							
3	<input type="checkbox"/>							
4	<input type="checkbox"/>							
5	<input type="checkbox"/>							
				■				■
	■	■	■		■	■	■	

- 1.- El *layout* que permite organizar los componentes de una interfaz gráfica como una matriz de filas y columnas es:
  - (a) \*\*\* GridLayout.
  - (b) Ninguna de las otras respuestas es correcta.
  - (c) FlowLayout.
  - (d) BorderLayout.
  
- 2.- Indica cuál de las siguientes afirmaciones sobre clases abstractas e interfaces Java es cierta:
  - (a) Una clase que hereda de una clase abstracta sólo puede implementar una interfaz.
  - (b) Una clase abstracta que implementa una interfaz siempre debe declarar alguno de los métodos de dicha interfaz como abstracto.
  - (c) \*\*\* Dos clases que implementan la misma interfaz pueden ofrecer distintas implementaciones de los métodos ofrecidos por dicha interfaz.
  - (d) Las clases abstractas pueden contener atributos, aunque no pueden definir constantes.
  
- 3.- Indica cuál de las siguientes afirmaciones sobre el modificador *static* es correcta:
  - (a) \*\*\* El valor de un atributo declarado con *static* es el mismo para todas las instancias de una clase.
  - (b) Los atributos declarados con *static* son constantes.
  - (c) Para invocar un método declarado con *static* es necesario crear una instancia de la clase que implementa dicho método.
  - (d) El modificador *static* no puede aplicarse a clases.
  
- 4.- Un array de objetos de la clase *Persona* puede contener:
  - (a) Exclusivamente objetos de la clase *Persona*.
  - (b) \*\*\* Objetos de la clase *Persona* y de cualquier subclase de esta.
  - (c) Objetos de las clases *Persona* y *Object*.
  - (d) Objetos de la clase *Persona* y de cualquier superclase de esta.
  
- 5.- Se programa la clase *MiEscuchador* que hereda de la clase *WindowAdapter*. Esta última implementa *WindowListener* entre otras interfaces. Indica cuál de las siguientes afirmaciones es correcta:
  - (a) La clase *MiEscuchador* debe ser abstracta.
  - (b) La clase *MiEscuchador* debe declarar como abstracto el método *windowClosing*.
  - (c) La clase *MiEscuchador* debe implementar todos los métodos que ofrece la interfaz *WindowListener*.
  - (d) \*\*\* La clase *WindowAdapter* proporciona una implementación por defecto de todos los métodos que ofrece la interfaz *WindowListener*.
  
- 6.- Dadas las clases *Persona* y *Alumno* de las cuales se muestran los siguientes fragmentos de código:

```

public class Persona {

}

public class Alumno extends Persona {
    int NIA;
    public Alumno(int NIA) {
        this.NIA = NIA;
        super();
    }
}

```

- (a) El código no produce errores de compilación.
- (b) Se produce un error de compilación porque a través de la sentencia *super()*; se está intentando acceder al constructor de *Persona*, el cual no se ha definido explícitamente.
- (c) \*\*\* Se produce un error de compilación debido a que la sentencia *super()*; está en una posición incorrecta en el código.
- (d) Se produce un error de compilación debido a que el parámetro que recibe el constructor tiene el mismo nombre que el atributo de la clase *Alumno*.

7.- Indica cuál de las siguientes afirmaciones sobre interfaces gráficas es correcta:

- (a) Las clases *JLabel* y *JButton* están incluidas en el paquete *java.awt*.
- (b) Una interfaz gráfica debe contener siempre al menos un botón.
- (c) Una interfaz gráfica se declara en Java como *public interface NombreInterfazGráfica*.
- (d) \*\*\* El método *getContentPane()* devuelve el panel de nivel superior de una ventana.

8.- El siguiente trozo de código de una interfaz gráfica permite deshabilitar un botón una vez pulsado. ¿Qué recibe como parámetro el método *actionPerformed* en el siguiente código?:

```

JButton button = new JButton("Haz click aquí");
button.addActionListener(new ActionListener() {
    public void actionPerformed(.....) {
        JButton clickedButton = (JButton) x.getSource();
        clickedButton.setEnabled(false);
    }
});

```

- (a) Un parámetro de tipo *ActionListener*.
- (b) Un parámetro de tipo *JButton*.
- (c) La referencia especial *this*.
- (d) \*\*\* Un parámetro de tipo *ActionEvent*.

9.- ¿Cuál de las siguientes afirmaciones sobre los constructores en Java es falsa?:

- (a) Un constructor permite inicializar las nuevas instancias de una clase que se crean.
- (b) \*\*\* Al declarar un constructor es necesario indicar explícitamente *void* como el tipo de datos del resultado.

- (c) Una clase puede tener varios constructores.
- (d) Las clases para las que el programador no proporcione ningún constructor cuentan con un constructor por defecto.

10.- Dado el siguiente trozo de código, y teniendo en cuenta que *Profesor* es una subclase de la clase *Persona*, indica cuál de las siguientes líneas de código compila correctamente pero produce un *error durante la ejecución del programa*:

```
Persona persona = new Persona();  
Profesor profesor = new Profesor();
```

- (a) profesor = persona;
- (b) persona = profesor;
- (c) \*\*\*  
profesor = (Profesor) persona;
- (d) persona = (Persona) profesor;