	Fecha:	CFGS – 1º DAM Entornos de desarrollo
	Examen UT3 – <i>Diseño y realización de pruebas</i>	

### Ejercicio 3.- [3 puntos]

Se desea crear una clase para calcular el consumo de electricidad de un cliente, que tendrá las siguientes variables de clase:

- **Abonado:** identificación del cliente
- **Bono\_social:** identificará si el cliente tiene o no asignado el bono social.
- **Consumo:** consumo en kw del cliente.
- **Antigüedad:** años de antigüedad en la compañía.
- **Total\_factura:** cantidad total a abonar.

Realizaremos un método (además de los constructores, *getter* y *setter*) para calcular el total de la factura, que tomará como parámetros un consumo, una antigüedad y una variable que indique si el cliente tiene bono social, siguiendo estos parámetros:

- Si el cliente tiene bono social el precio de la factura será 0.
- El precio del *kilowatio* será 0.2575 para consumos de menos de 100 kw, 0.2050 entre 100 y 250 y 0.1955 para más de 250.
- El descuento será de un 0% para abonados con menos de 5 años de antigüedad, un 2% entre 5 y 10 años y de un 5% para clientes con una antigüedad de más de 10 años.





**El código fuente lo tenéis en un proyecto adjunto, en el aula virtual**

Se pide:

- Diseña la prueba de cubrimiento para el método **“calcularTotalFactura()”**. Hazlo utilizando pruebas automatizadas.

**NOTA:** Se deberá configurar el parámetro **“delta”** como **0.01**

 <b>IES Ribera del Tajo</b> <small>Enseñanza bilingüe</small>  <small>Castilla-La Mancha</small>	<b>Fecha:</b>	CFGS – 1º DAM Entornos de desarrollo
	<b>Examen UT3 – <i>Diseño y realización de pruebas</i></b>	

- b) Utilizar la herramienta "coverage" disponible en Eclipse para analizar y determinar la cobertura de código obtenida por las pruebas implementadas. Proporciona una captura de pantalla que muestre los resultados de la cobertura específicamente para el método **"calcularTotalFactura()"**, verificando así la efectividad de las pruebas diseñadas.



#### **Ejercicio 4.- [1.5 puntos]**

Consideremos una aplicación de registro para un evento académico, donde los participantes pueden inscribirse en línea proporcionando cierta información y eligiendo entre varias opciones de talleres. La aplicación requiere la siguiente entrada para procesar una inscripción:

- **Nombre Completo:** debe contener al menos dos palabras, con un máximo de 40 caracteres.
- **Correo Electrónico:** debe seguir un formato válido de correo electrónico (por ejemplo, usuario@dominio.com).
- **Tipo de Participante:** puede ser "Estudiante", "Profesor" o "Profesional". Cualquier otro valor es considerado inválido.
- **Selección de Taller:** puede estar en blanco (indicando que no se selecciona ningún taller) o ser uno de los siguientes valores: "Taller A", "Taller B", "Taller C". Cualquier otro valor es considerado inválido.

La aplicación responde de la siguiente manera:

- Si todos los datos son válidos y se ha seleccionado un taller, el participante es registrado en el taller elegido.
- Si ocurre algún error en la entrada de datos, el programa muestra un mensaje de error específico sobre el dato implicado.

 <b>IES Ribera del Tajo</b> <small>Enseñanza bilingüe</small>  <small>Castilla-La Mancha</small>	Fecha:	CFGS – 1º DAM Entornos de desarrollo
	<b>Examen UT3 – <i>Diseño y realización de pruebas</i></b>	

Se pide:

- Definir las clases de equivalencia,
- 2 ejemplos de casos de prueba válidos
- 2 casos de prueba no válidos que cubran dos clases no válidas para la inscripción en el evento académico.

### Ejercicio 5.- [2.5 puntos]

La clase **CuentaBancaria** está diseñada para simular el comportamiento de una cuenta bancaria real en el contexto de una aplicación de software. Esta clase ofrece funcionalidades esenciales para gestionar una cuenta bancaria, permitiendo al usuario realizar operaciones básicas como depositar dinero, retirar dinero siempre que haya saldo suficiente, y consultar el saldo actual de la cuenta.

Se han detectado dos errores lógicos y otro error de compilación.



Se pide:

- Encuentra el error de compilación en el código fuente y arréglalo. Explica qué es un error de compilación y explica (con capturas de pantalla) qué has hecho para arreglarlo.
- Encuentra los 2 errores lógicos depurando paso a paso el código. Señala qué errores son y qué solución has planteado para arreglar el dicho error.

**NOTA:** Explica al menos la depuración del código para arreglar UNO de estos errores.



**El código fuente lo tenéis en un proyecto adjunto, en el aula virtual**

 <b>IES Ribera del Tajo</b> <small>Enseñanza bilingüe</small>  <small>Castilla-La Mancha</small>	Fecha:	CFGS – 1º DAM Entornos de desarrollo
	<b>Examen UT3 – <i>Diseño y realización de pruebas</i></b>	

El main de la clase ***CuentaBancariaPrueba*** debería obtener por pantalla el siguiente resultado:

```
Saldo inicial: 1000.0
Despu s de depositar 500: 1500.0
Despu s de retirar 200: 1300.0
Retiro fallido. Saldo insuficiente.
Saldo final: 1300.0
```

#### NOTAS:

- No se podrá añadir ni eliminar métodos. Solo modificar el código fuente que ya existe.

#### Instrucciones de entrega:

- Se entregarán todos los proyectos y código fuente que se utilice para realizar los ejercicios
- Se entregará un único PDF con las respuestas del examen y capturas de pantalla.
- Se subirá al aula virtual un archivo comprimido con el siguiente formato ***Apellido1\_Apellido2\_Nombre\_ENDE.rar***