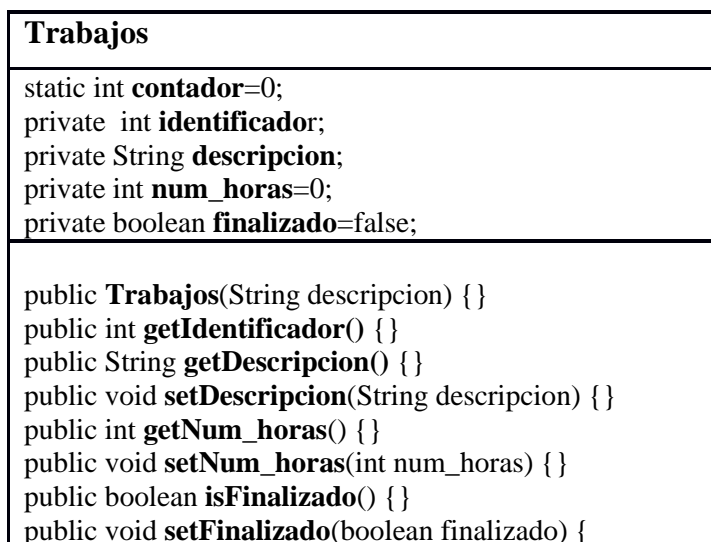
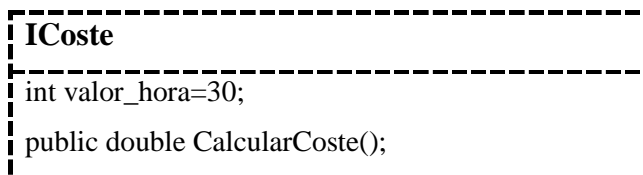
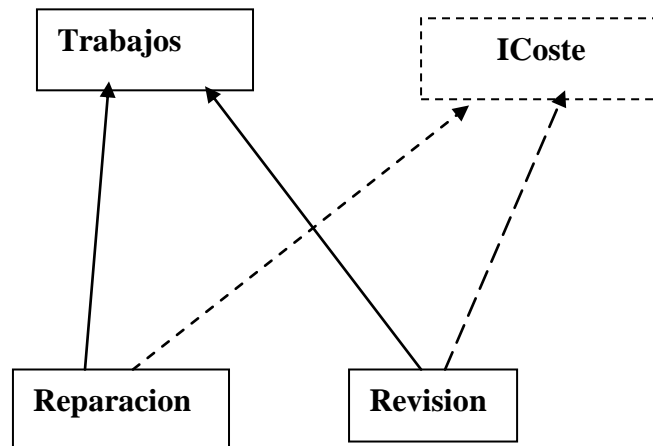


## 1ª Evaluación. Ejercicio 1

Nombre: \_\_\_\_\_

1.- Implementa el siguiente diagrama de clases e interfaces que modelan las revisiones y reparaciones en un taller mecánico (no usar acentos para definir las clases).



Reparacion
private int <b>tipo</b> ; //0 para reparación mecánica y 1 para reparación chapa y pintura private double <b>coste_material</b> =0.0; private double <b>precio</b> =0.0;
public <b>Reparacion</b> (int tipo, String descripcion) {} public int <b>getTipo</b> () {} public double <b>getCoste_material</b> () {} public void <b>setCoste_material</b> (double coste_material) {} public double <b>getPrecio</b> () {} public void <b>setPrecio</b> (double precio) {} public double <b>CalcularCoste</b> (){} <b>@Override</b> public String <b>toString</b> () {}

Revision
private double <b>precio</b> =0.0;
public <b>Revision</b> (String descripcion) {} public double <b>getPrecio</b> () {} public void <b>setPrecio</b> (double precio) {} public double <b>CalcularCoste</b> (){} <b>@Override</b> public String <b>toString</b> () {}

### Descripción general y requisitos:

Se desea realizar una aplicación que permita a los mecánicos de un garaje registrar, consultar y actualizar los trabajos (reparaciones y revisiones) que han sido realizadas o que están en proceso de realización en un taller mecánico.

Cada **Trabajo** se identifica unívocamente por su “identificador de trabajo”. Este **identificador** es un número que se asocia con el trabajo en el momento que se registra. El primer trabajo registrado tendrá identificador 1, el segundo 2 y así sucesivamente.

Los trabajos incluyen una pequeña **descripción** de la reparación o revisión a realizar.

Todos los trabajos incluyen el número de horas que van siendo necesarias para su realización. Al crear un trabajo el número de horas (**num\_horas**) es cero. El número de horas irá aumentando a medida que los mecánicos van dedicando tiempo a realizar la revisión o reparación.

Cuando un trabajo se ha finalizado se marca como “**finalizado**” y el número de horas no se puede volver a cambiar.

Las **Reparaciones** incluyen el coste del material utilizado (piezas o pintura). El atributo **coste\_material** será 0 en el momento de registrarse y va aumentando a medida que los mecánicos van utilizando material en la reparación.

Cuando un trabajo se ha finalizado se marca como “**finalizado**” y el coste de material no se puede volver a cambiar.

Para diseñar el método **CalcularCoste()** ten en cuenta que:

El precio a cobrar (**precio**) por cada trabajo se compone de una **parte fija** que resulta de multiplicar el número de horas empleadas por el valor de la hora. Además dependiendo del tipo de trabajo el coste varía de la siguiente manera.

- Para **reparaciones**:
  - Si es mecánica su precio se calcula como **parte fija** más el coste material multiplicado por 1.1
  - Si es de chapa y pintura su precio se calcula como **parte fija** más el coste material multiplicado por 1.3
- Para **revisiones** se calcula como **parte fija** más 20 €.

La cantidad que vamos a cobrar una vez calculada se asigna al atributo precio, aunque también sea devuelta por el método.

Se debe sobrescribir el método toString() y para cada clase se debe mostrar lo siguiente:

Para las Reparaciones:

**Reparación Mecánica./ Reparación Chapa y Pintura.**

**Identificador:** \_\_\_\_\_

**Descripción:** \_\_\_\_\_

**Nº de Horas:** \_\_\_\_\_

**Finalizado:** SI/NO

**Coste Piezas: / Coste Pintura:** \_\_\_\_\_

**Precio Reparación:** \_\_\_\_\_

Para las Revisiones:

**Revisiones.**

**Identificador:** \_\_\_\_\_

**Descripción:** \_\_\_\_\_

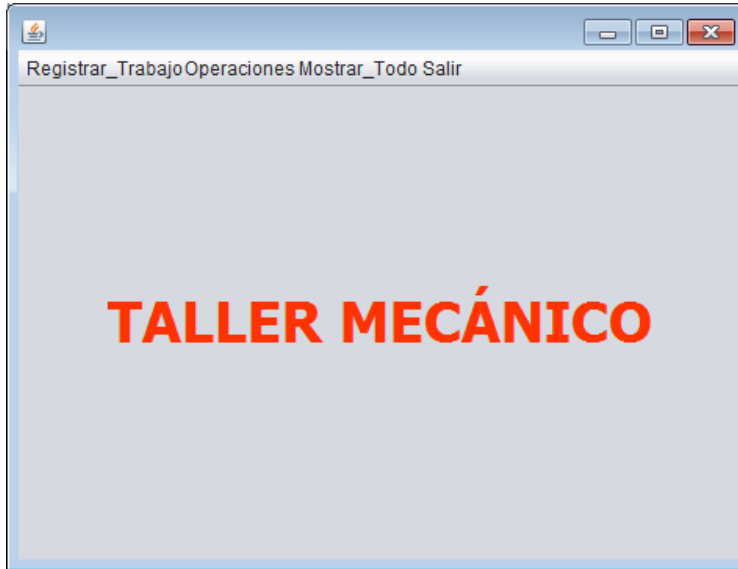
**Nº de Horas:** \_\_\_\_\_

**Finalizado:** SI/NO

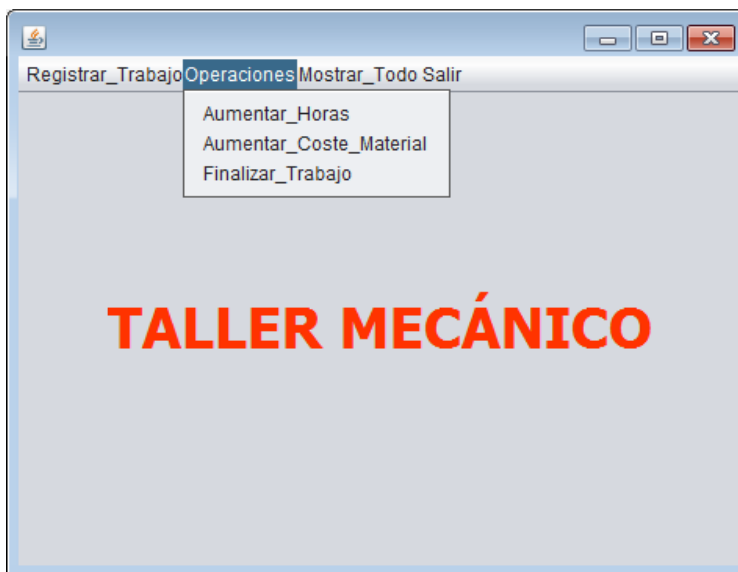
**Precio Revisión:** \_\_\_\_\_

La aplicación se gestionará mediante una aplicación gráfica basada en menú.

## 2.- Diseña el frame **Principal.java**.



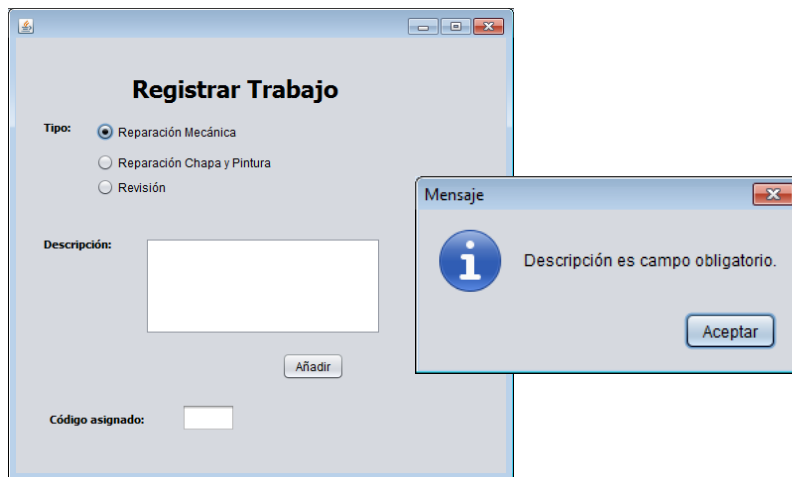
La opción **Operaciones** tiene tres subopciones:



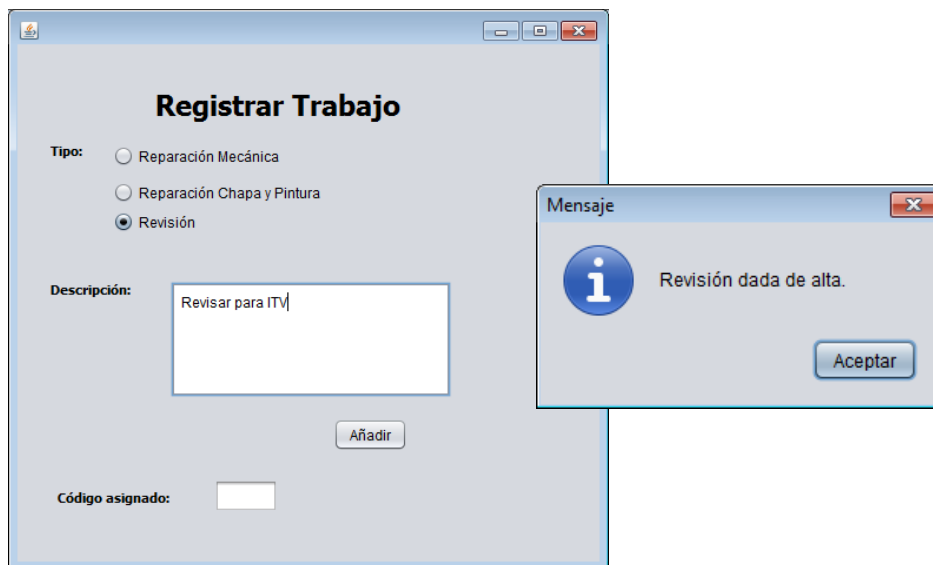
Declara una **lista polimórfica** de nombre **lista** para guardar los objetos **Trabajos** añadidos.

## 3.- Cada opción abre un nuevo frame. Diséñalos.

Empieza con **Registrar.java**



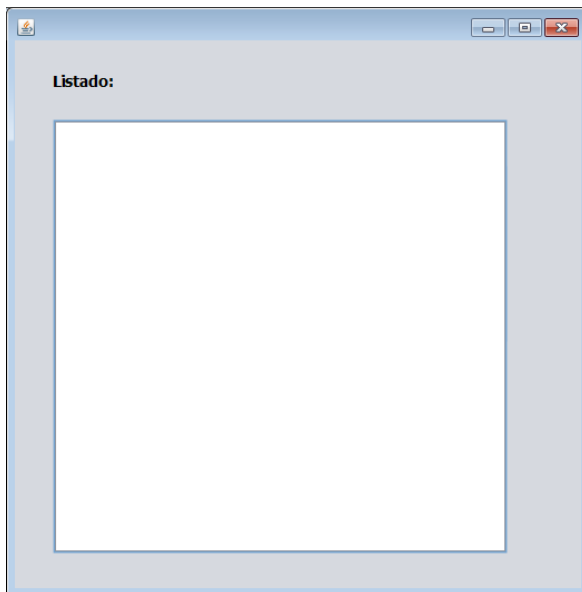
El campo Descripción es obligatorio. Si está vacío muestra mensaje.



Una vez introducidos los datos, al pulsar el botón la aplicación añade el trabajo a la lista y muestra el identificador asignado al trabajo.

4.- Diseña la opción **Mostrar\_Todo**, que abre un nuevo frame.

Antes de sacar el listado se deberá calcular el precio que tiene el trabajo.



Prueba a dar de alta los siguientes trabajos y sacar el listado.

**REPARACIÓN MECÁNICA:**

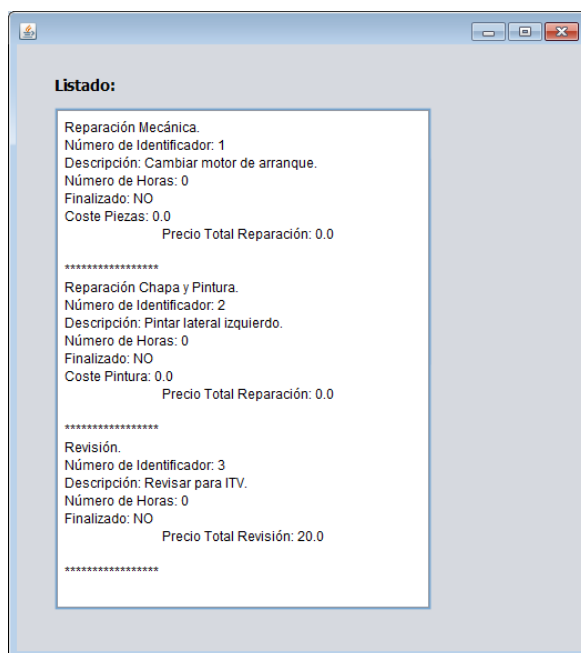
Descripción: Cambiar motor de arranque.

**REPARACIÓN CHAPA Y PINTURA:**

Descripción: Pintar puerta lateral izquierda

**REVISIÓN:**

Descripción: Revisión para ITV: aceite, faros y filtros.



Los precios de los trabajos se calcularán correctamente cuando se hayan introducido valores de horas y de material.

El listado una vez introducidas las horas y los gastos muestra los cálculos correctos. Por ejemplo:

**Listado:**

Reparación Mecánica.  
Número de Identificador: 1  
Descripción: Cambiar motor de arranque.  
Número de Horas: 5  
Finalizado: NO  
Coste Piezas: 400.0  
Precio Total Reparación: 590.0

\*\*\*\*\*

Reparación Chapa y Pintura.  
Número de Identificador: 2  
Descripción: Pintar lateral izquierdo.  
Número de Horas: 3  
Finalizado: NO  
Coste Pintura: 100.0  
Precio Total Reparación: 220.0

\*\*\*\*\*

Revisión.  
Número de Identificador: 3  
Descripción: Revisar para ITV.  
Número de Horas: 2  
Finalizado: NO  
Precio Total Revisión: 80.0

\*\*\*\*\*

5. La opción **Operaciones** tiene tres subopciones.  
Cada subopción abre un nuevo frame. Diséñalos.

### Finalizar.java

**Finalizar trabajo:**

Introduce Código:

Finalizar

**Excepción NumberFormatException**

Formato de código no válido.

Aceptar

**Mensaje**

El código tecleado no está dado de alta.

Aceptar

**Finalizar trabajo:**

Introduce Código:

Finalizar

**Finalizar trabajo:**

Introduce Código:

Finalizar

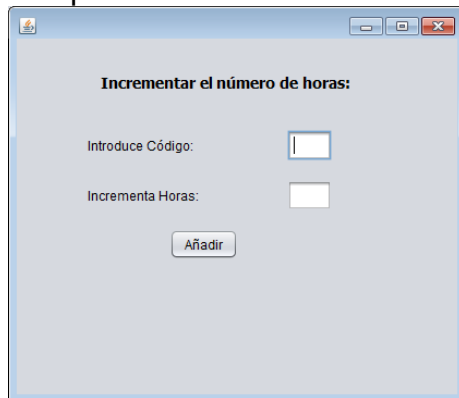
Estado anterior:

Estado Actual:

## Mas\_Horas.java

Muestra una ventana que permite introducir el identificador del trabajo y el número de horas.

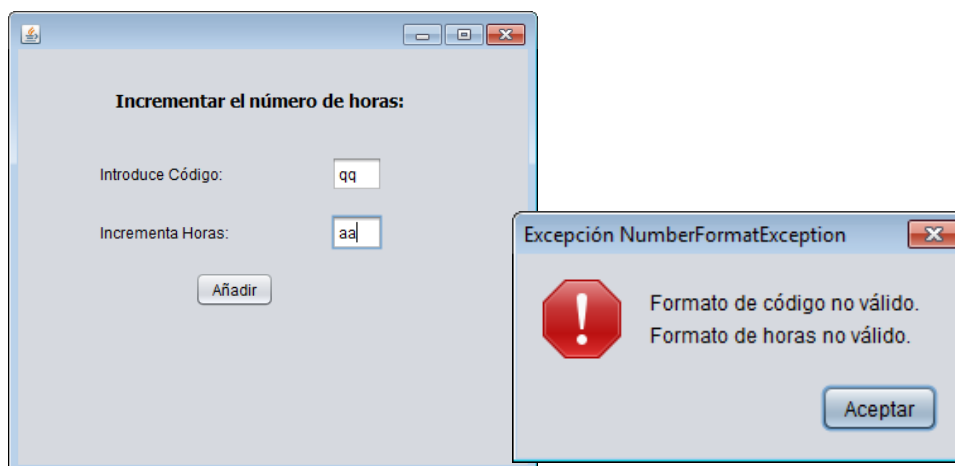
La aplicación aumenta el número de horas.



The screenshot shows a Java Swing window titled "Incrementar el número de horas:". It contains two text input fields. The first field is labeled "Introduce Código:" and is empty. The second field is labeled "Incrementa Horas:" and is also empty. Below the fields is a button labeled "Añadir".

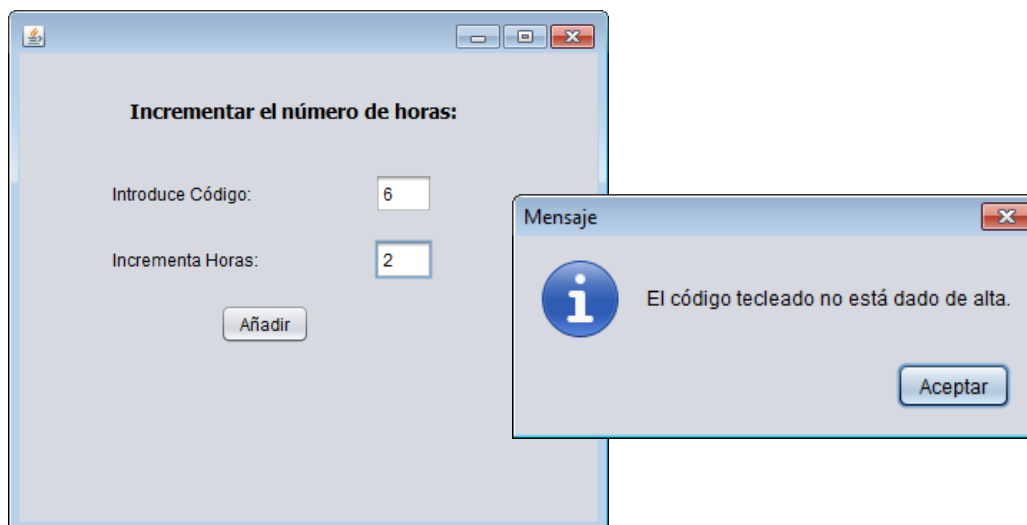
### Se notifican los errores siguientes:

Si el formato del código y/o horas no es correcto.



The screenshot shows the "Incrementar el número de horas:" window with "qq" in the "Introduce Código:" field and "aa" in the "Incrementa Horas:" field. An error dialog box titled "Excepción NumberFormatException" is displayed over the window. It contains a red octagonal warning icon and the text "Formato de código no válido." and "Formato de horas no válido." Below the text is an "Aceptar" button.

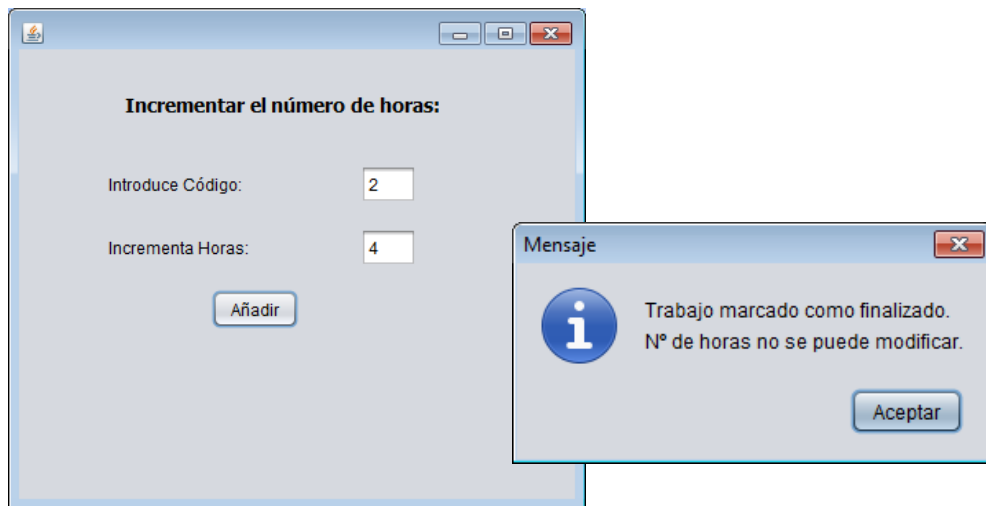
Si código no está dado de alta.



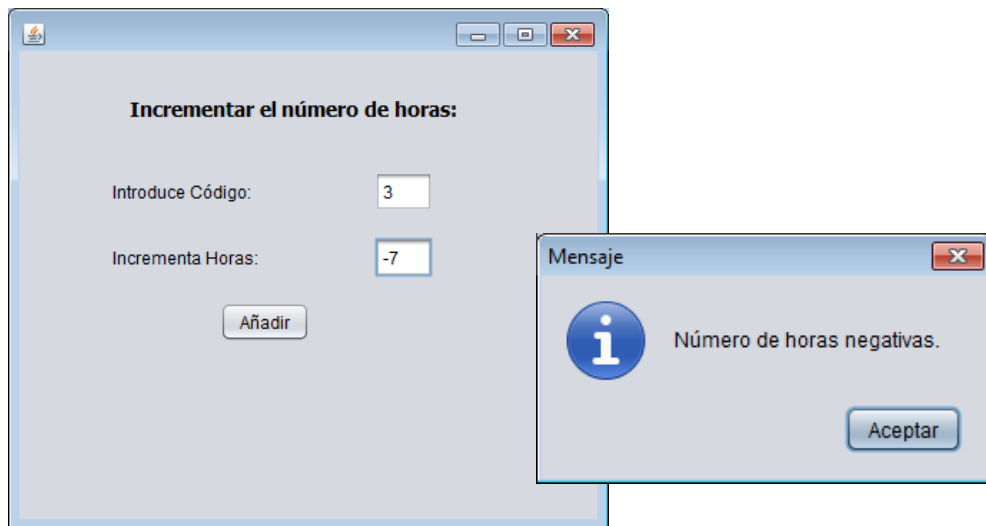
The screenshot shows the "Incrementar el número de horas:" window with "6" in the "Introduce Código:" field and "2" in the "Incrementa Horas:" field. An information dialog box titled "Mensaje" is displayed over the window. It contains a blue circular information icon and the text "El código tecleado no está dado de alta." Below the text is an "Aceptar" button.

Si trabajo está finalizado.

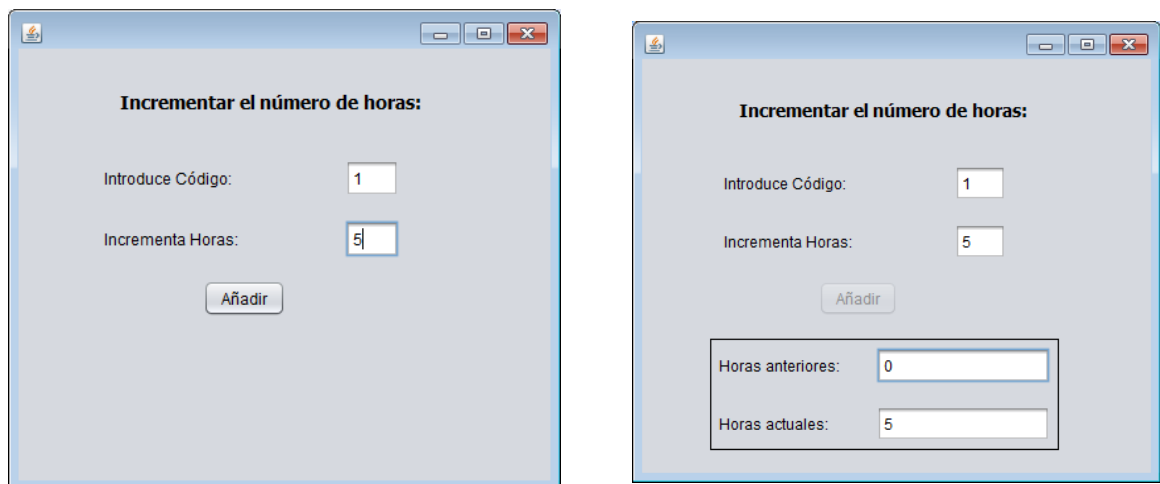




Si el número de horas es negativo.



En otro caso se aumenta el número de horas.



**Incrementar el número de horas:**

Introduce Código:

Incrementa Horas:

**Incrementar el número de horas:**

Introduce Código:

Incrementa Horas:

Horas anteriores:

Horas actuales:

### Mas\_Costes.java

Muestra una ventana que permite introducir el identificador del trabajo y el coste a aumentar.

La aplicación aumenta el coste.

**Incrementar coste de la Reparación:**

Introduce Código:

Incrementa Coste:

### Se notifican los errores siguientes:

Si el formato del código y/o coste no es correcto.

**Incrementar coste de la Reparación:**

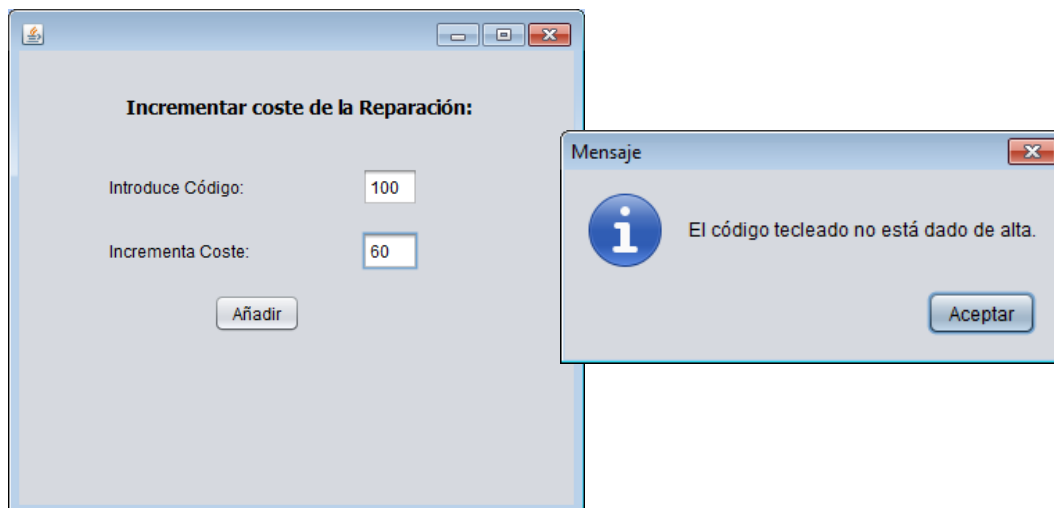
Introduce Código:

Incrementa Coste:

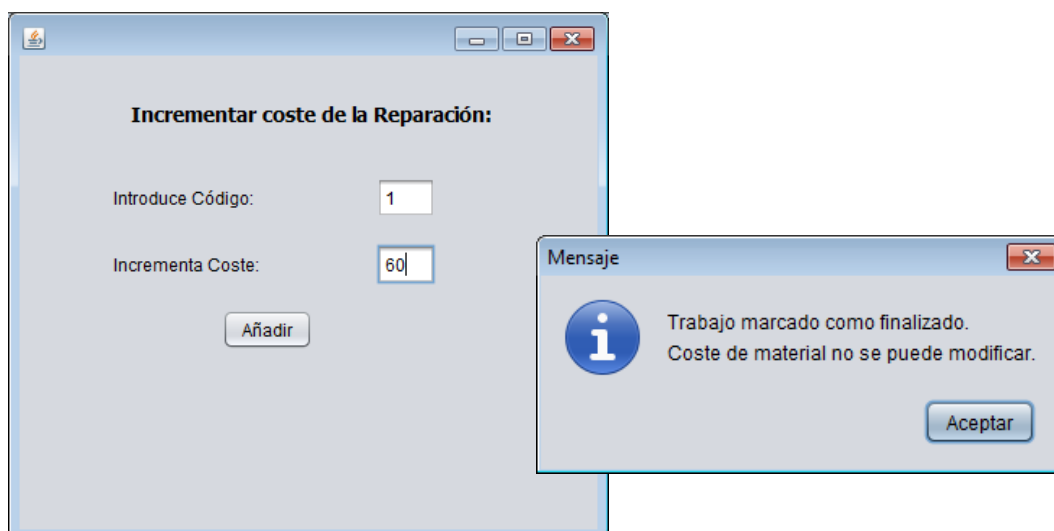
**Excepción NumberFormatException**

Formato de código no válido.  
Formato de horas no válido.

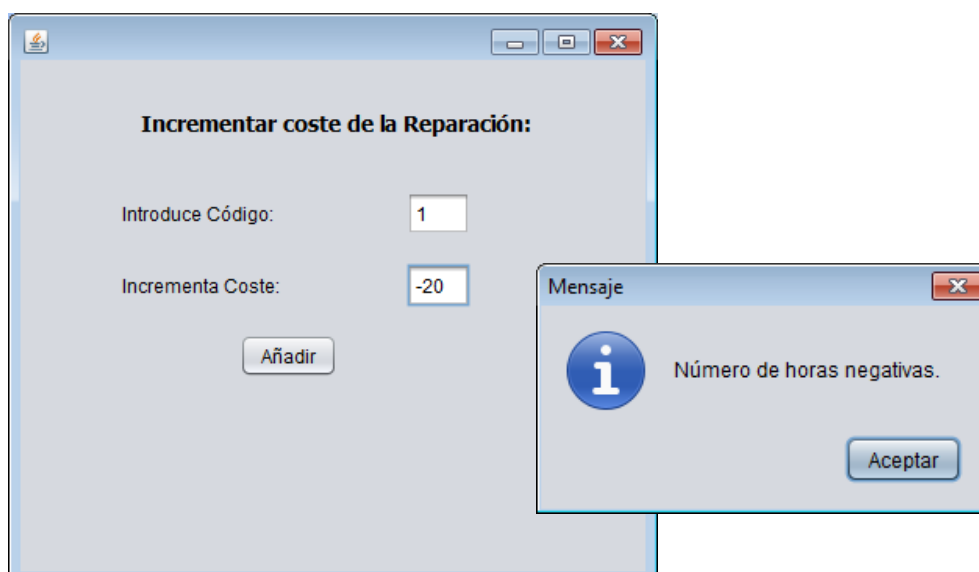
Si código no está dado de alta.



Si trabajo está finalizado.



Si el número de horas es negativo.



No se trata de una Reparación

The image shows two overlapping windows. The background window is titled "Incrementar coste de la Reparación:" and contains two input fields: "Introduce Código:" with the value "2" and "Incrementa Coste:" with the value "45". Below these fields is a button labeled "Añadir". The foreground window is titled "Mensaje" and contains an information icon (a blue circle with a white 'i') and the text "No es una reparación." Below the text is a button labeled "Aceptar".

En otro caso se aumenta el coste de material.

The image displays four instances of the "Incrementar coste de la Reparación:" dialog box, arranged in a 2x2 grid. Each instance shows the "Introduce Código:" and "Incrementa Coste:" fields, along with an "Añadir" button. The top-right instance also includes a section with "Coste anterior:" and "Coste actual:" fields. The values for each instance are as follows:

Introduce Código:	Incrementa Coste:	Coste anterior:	Coste actual:
1	50	0.0	50.0
1	20	50.0	70.0
1	50		
1	20		