



## **Ejercicio Nivel 9**

### **CupiTrenes**

#### **Objetivos del ejercicio**

El objetivo de este ejercicio es que el estudiante comprenda y adquiera práctica en:

- El desarrollo de aplicaciones siguiendo un proceso incremental.
- La implementación y uso de estructuras lineales sencilla y doblemente encadenadas.
- La creación de algoritmos de adición, eliminación y búsqueda en listas.
- El desarrollo de nuevos elementos gráficos para construir interfaces de usuario.

Los siguientes pasos conforman el plan sugerido para desarrollar el ejercicio. La idea es ir desarrollando y probando incrementalmente los métodos de las clases. **No se preocupe si las clases de la interfaz o de las pruebas (test) tienen errores. Estos desaparecerán cuando termine (correctamente) los cambios en el modelo del mundo.**

Este ejercicio debe ser realizado de manera **INDIVIDUAL**.

#### **Preparación**

1. Descargue del sitio web del curso el archivo demo de la aplicación (del enlace llamado **n9\_cupiTrenes\_demo.mp4**) y ejecútelo para conocer el funcionamiento esperado del programa.
2. Descargue del sitio web del curso el esqueleto del ejercicio (del enlace llamado **n9\_cupiTrenes\_Esqueleto**). Descomprima este archivo e importe el proyecto llamado **n9\_cupiTrenes** en Eclipse.
3. Lea el enunciado del problema disponible en:  
**n9\_cupiTrenes/docs/specs/Descripcion.pdf**.
4. Estudie el documento de requerimientos funcionales disponible en:  
**n9\_cupiTrenes/docs/specs/RequerimientosFuncionales.pdf**.
5. Estudie el documento de requerimientos no funcionales disponible en:  
**n9\_cupiTrenes/docs/specs/RequerimientosNoFuncionales.pdf**.
6. Estudie el modelo del mundo diseñado para este ejercicio. Este modelo se encuentra en:  
**n9\_cupiTrenes/docs/specs/ModeloConceptual.jpg**. Identifique las clases, relaciones entre clases, constantes, atributos y métodos.

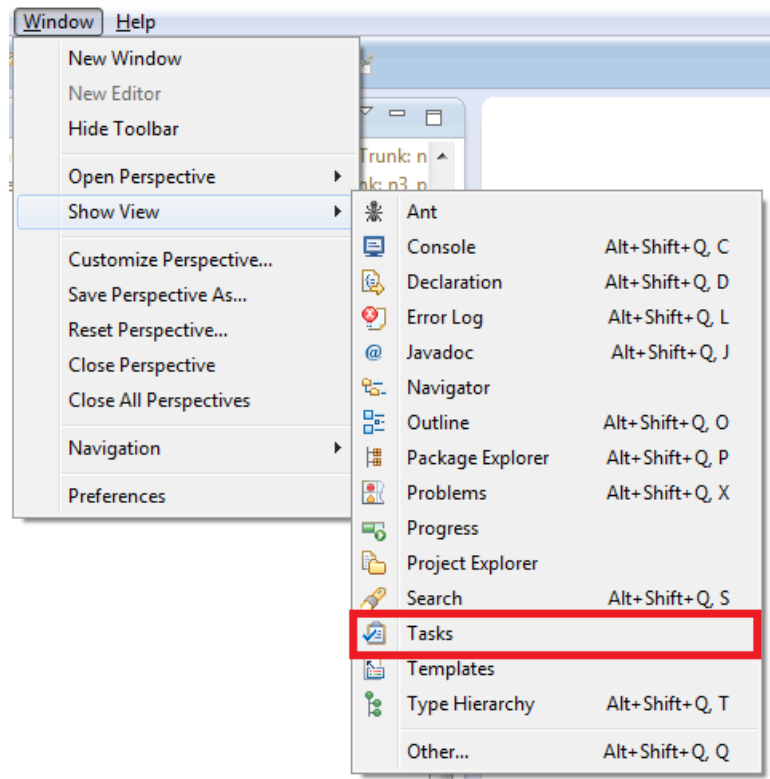
7. Estudie el documento de consideraciones adicionales de diseño disponible en: **n9\_cupiTrenes/docs/spec/ConsideracionesAdicionalesDeDisenno.pdf**.
8. Desde Eclipse revise la documentación de las clases del mundo. Esto le permitirá entender para qué sirve cada método y cada uno de los atributos. También puede generar la documentación del proyecto (archivos .html) ejecutando el programa **doc** que se encuentra en **n9\_cupiTrenes/bin/win** (para Windows) y en **n9\_cupiTrenes/bin/mac** (para Mac).
9. Asegúrese de tener activado el uso de aserciones para la ejecución del programa. Ver el tutorial en:  
[http://cupi2.uniandes.edu.co/sitio/images/cursosCupi2/apo2/tutoriales/n7\\_assert.pdf](http://cupi2.uniandes.edu.co/sitio/images/cursosCupi2/apo2/tutoriales/n7_assert.pdf)

## Desarrollo

Dentro del código del esqueleto se encuentran indicados los puntos donde usted debe realizar alguna modificación (añadir atributos, completar métodos, construir nuevos métodos, etc.), por medio de comentarios de la siguiente forma:

**// TODO ParteX PuntoY:** Breve explicación de la modificación que debe realizar.

Donde X hace referencia a una parte de la guía de trabajo, y Y hace referencia a un punto de la parte X. Para visualizar los **TODOs** vaya al menú Window -> Show View -> Tasks como se muestra en la siguiente figura:



En el siguiente vínculo encuentra un video que le explica cómo consultar los TODOs de un ejercicio:

<https://www.youtube.com/watch?v=pV54O42D2ow>

## Parte 1: Listas sencillamente encadenadas

Completar los TODOs que comienzan por: **//TODO Parte1 PuntoY**

## Parte 2: Listas doblemente encadenadas

Completar los TODOs que comienzan por: **//TODO Parte2 PuntoY**

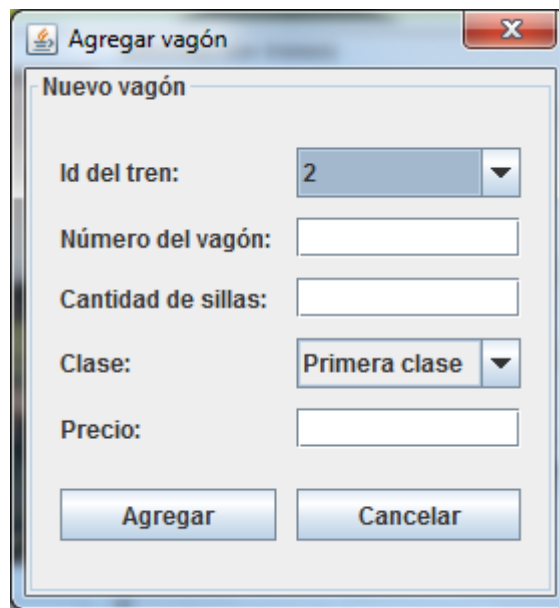
## Parte 3: Invariantes

1. Llame el método `verificarInvariante` en todos los métodos modificadores de la clase `Vagon`
2. Llame el método `verificarInvariante` en todos los métodos modificadores de la clase `Tren`
3. Llame el método `verificarInvariante` en todos los métodos modificadores de la clase `CupiTrenes`

**Nota:** Tenga en cuenta que para comparar dos fechas puede usar los métodos `compareTo`, `after` o `before`.

## Parte 3: Interfaz gráfica

1. `JDialog`:
  - a. Cree un diálogo `DialogoAgregarVagon` (a partir de la clase `JDialog`) que permita agregar una función a una ciudad. **Nota:** Puede usar como referencia `DialogoAgregarTren`.
  - b. Implemente el diálogo usando un distribuidor gráfico de tipo `GridBagLayout`.
  - c. Use `JComboBox` para guardar los identificadores de los trenes (se puede guiar por la clase `DialogoEliminarVagon`).
  - d. Use `JComboBox` para elegir la clase del tren.
  - e. Valide los datos ingresados por el usuario.
  - f. Llame al método de la interfaz principal `agregarVagon` para agregar el vagón a la ruta.
  - g. El diálogo debe verse como se muestra a continuación:



## Validación

Para comprobar el funcionamiento de su ejercicio usted puede:

1. Ejecutar las pruebas automáticas disponibles en el ejercicio (validación funcional del mundo) que le permitirán verificar la correcta implementación de sus métodos. Para ejecutar las pruebas en el entorno de Eclipse presione clic derecho sobre el paquete “uniandes.cupi2.cupiTrenes.test” (o alguna de sus clases internas), seleccione la opción *Run as* y posteriormente la opción *JUnitTest*. En el siguiente vínculo encuentra un video que le explica cómo ejecutar e interpretar las pruebas automáticas:

<https://www.youtube.com/watch?v=h3r7wSFaIOo>

2. Ejecutar el programa e interactuar con todas las opciones disponibles en la interfaz.

## Entrega

Este ejercicio debe ser realizado de manera **INDIVIDUAL**.

1. Construya el archivo entregable con el ejercicio completo. No olvide revisar que su entrega cumple con lo especificado en las normas del curso referentes a entregas de ejercicios. Consultar:

<http://cupi2.uniandes.edu.co/sitio/index.php/cursos/apo1/normas-administrativas>

2. Entregue el archivo del ejercicio vía SicuaPlus, de acuerdo con las normas, fecha y hora de entrega.

**NOTA.** No olvide:

1. Renombrar el archivo de entrega con su login de estudiante. El nombre del archivo que contiene el ejercicio debe cumplir el siguiente formato:

n<nivel del ejercicio>\_<login estudiante>.zip  
(por ejemplo: n9\_tsuares.zip)

2. En el siguiente vínculo encuentra un video que le explica cómo indentar el código de su proyecto:

<https://www.youtube.com/watch?v=BH9H0e-Z56E>