FUNDAMENTOS DE INFORMÁTICA

PRÁCTICA

CURSO 2008/2009

Contenido

INTRODUCCIÓN

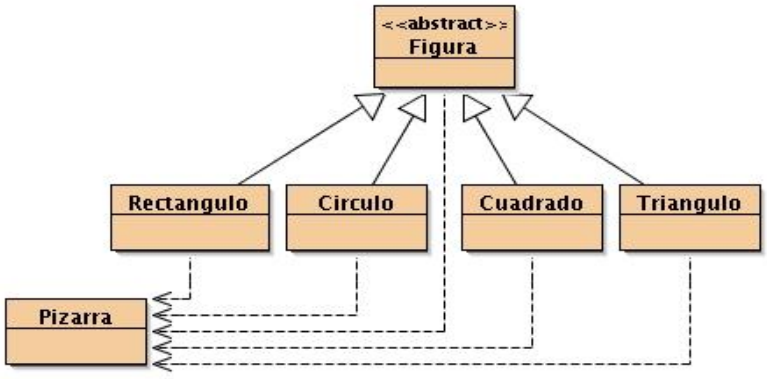
Los alumnos que cursen la asignatura de Fundamentos de Informática de las Ingenierías Técnicas de Industriales **tienen que realizar las prácticas de forma obligatoria**. El objetivo de la práctica es sustentar el estudio de la asignatura. Las prácticas serán monitorizadas y evaluadas por los tutores en los centros asociados. La práctica está organizada en tres etapas para facilitar la temporización del estudio de la asignatura a lo largo del cuatrimestre. Los alumnos deberán asistir a tres sesiones presenciales donde podrán comentar y contrastar cada uno de los apartados de los que consta dicha práctica. La práctica está planteada para introducir a los alumnos en conceptos básicos de la programación orientada a objetos de una forma incremental. Al final de la realización de esta práctica el alumno adquirirá la base necesaria para poder resolver problemas utilizando la metodología enseñada.

La práctica es un trabajo personal, y se advierte que si se detectan copias, conllevará un suspenso para todas las convocatorias del curso académico 2008/2009, de todos los implicados.

DESCRIPCIÓN DE LA PRÁCTICA

La idea fundamental de la práctica es que el alumno aprenda a modelar de forma incremental y reutilizar diseños. En concreto vamos a comenzar modelando objetos que van a visualizarse gráficamente, La visualización de un objeto se realizará mediante la combinación de un conjunto de figuras geométricas que se le proporcionarán al inicio de la práctica y que disponen de un método que permite pintarse en una clase *Pizarra*. Además de pintarse por medio de una clase Pizarra, similar a la clase *Canvas* del ejercicio 1.2 del libro de referencia de la unidad Didáctica II, deberá modelarse el comportamiento de cada uno de los elementos

Se parte de las siguientes clases dadas:



- La clase **Pizarra** permite dibujar en una ventana diversas figuras geométricas.
- La clase **Figura** representa una figura geométrica que se ubicará en unas coordenadas '*x*' e '*y*', tendrá un color determinado y será capaz de pintarse en la pizarra por medio de un método “draw” que hace uso de la clase Pizarra .

- Además, como tipos específicos de figuras se tienen las siguientes clases:

  - **Rectángulo**, **Círculo, Cuadrado** y **Triángulo**, que representan las respectivas figuras geométricas con los atributos necesarios para especificar las características de cada tipo de figura. Por ejemplo, un rectángulo se caracteriza por su base y su altura. Cada una de estas clases implementa de forma específica el método "draw" (ver código fuente en el apéndice).

(Nota: El código fuente de estas clases puede encontrarse en el apéndice de esta memoria, así como en el apartado de la práctica dentro del entorno virtual de CiberUNED)

Los diagramas UML de las clases correspondientes a las diferentes figuras geométricas son:

Figura
#color : **String**
+ setColor(newColor: **String**): **void** + getColor(): **String** + erase(): **void** + draw(): **void**

Rectangulo
- base : **int** - altura : **int**
+ Rectangulo(base: **int**, altura: **int**, color: **String**): **void** + setBase(base: **int**): **void** + setAltura(altura: **int**): **void** + getBase(): **int** + getAltura(): **int** + draw(x: **int**, y: **int**): **void**

Circulo
- diametro : **int**
+ Circulo(diametro: **int**, color: **String**): **void** + setDiametro(diametro: **int**): **void** + getDiametro(): **int** + draw(x: **int**, y: **int**): **void**

Cuadrado
- lado : **int**
+ Cuadrado(lado: **int**, color: **String**): **void** + setLado(lado: **int**): **void** + getLado(): **int** + draw(x: **int**, y: **int**): **void**

Triangulo
- alto : **int** - ancho : **int**
+ Triangulo(alto: **int**, ancho: **int**, color: **String**): **void** + setAlto(alto: **int**): **void** + setAncho(ancho: **int**): **void** + getAlto(): **int** + getAncho(): **int** + draw(x: **int**, y: **int**): **void**

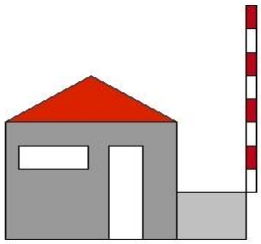
En esta práctica se considera que una clase ha sido correctamente implementada cuando:

1. Contenga todos los atributos que permitan caracterizar cada instancia (objeto) de la clase.
2. Incluya los métodos necesarios para acceder o actualizar los valores de sus atributos (métodos *get* y *set*)
3. Incluya los métodos necesarios para modelar el comportamiento del objeto
4. Permita representar gráficamente el objeto por medio de la clase Pizarra, empleando composición sobre figuras geométricas básicas.

A continuación, se presenta un ejemplo. Si se tuviera que modelar un apeadero, en donde lo que interesa es caracterizar: (1) los billetes que se han vendido en un determinado periodo de tiempo y (2) la recaudación suponiendo que sólo se vende un tipo de billete y (3) producir un dibujo del edificio, en forma de caseta con una puerta, una ventana y una barrera de control podría hacerse como sigue:

Apeadero
- numVentanas : **int** - numPuertas : **int** - billetesVendidos : **int** - barrera : **boolean**
+ Estacion(numVentanas: **int**, numPuertas: **int**, barrera: **boolean**): **void** + setNumVentanas(numVentanas: **int**): **void** + setNumPuertas(numPuertas: **int**): **void** + getNumVentanas(): **int** + getNumPuertas(): **int** + recaudacion(fecha: **date**): **int** + draw(x: **int**, y: **int**): **void** + venderBillete(fecha: **date): void**

donde el método *draw()* debería dibujar en unas coordenadas '*x*' e '*y*' algo similar a:



y los métodos permitirán acceder al valor de los diferentes atributos definidos, así como a la recaudación de ésta.

PARTE 1

OBJETIVOS

En la primera etapa de la práctica se van a afianzar los conceptos básicos de la programación en Java. Trabajaremos con las características fundamentales de **clase, atributos y métodos** (constructores, métodos que ayudan a la encapsulación de la clase y métodos de visualización).

En cuanto a los atributos, aparte de los tipos primitivos, haremos uso de tipos enumerados para definir el conjunto de valores que pueden ser asignados a un atributo.

REQUISITOS PREVIOS

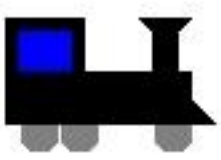
Esta primera parte requiere haber estudiado los temas 4, 5 y 6 del temario detallado de la asignatura, correspondientes a los capítulos 1, 2 y 3, así como los apéndices B, C, D y G del libro de referencia de la Unidad Didáctica II.

ENUNCIADO

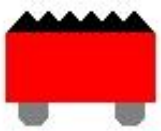
Se desea modelar un sistema informático cuyo fin sea la creación y representación de un **ferrocarril**, entendiendo como tal el conjunto de vehículos que constituyen este medio de transporte, y empleando la clase dada Pizarra para la representación de los objetos.

Dentro de esta primera parte, deberá crearse una clase que modele una locomotora y un vagón. Estas clases tendrán asociadas una serie de atributos y un método que, análogamente a las clases dadas, permita pintarlos en una instancia de la clase Pizarra.

Una locomotora se caracteriza por tener una marca, un modelo, un peso y una capacidad de arrastre máxima (en kilos) . Por otro lado, un objeto de la clase locomotora deberá presentar un aspecto similar al siguiente:



Un vagón se caracteriza por su peso (sin carga), su peso máximo de carga, un tipo de carga para el que está destinado ("hierro", "carbón" o "madera") y el peso de la carga que lleva. El aspecto que deberá presentar un vagón cargado a su máxima capacidad es similar al siguiente:



Y uno a media carga, con menos triángulos:

El alumno debe diseñar e implementar el esquema de clases y relaciones necesarios para poder instanciar y pintar los objetos descritos. Deberá instanciarse un objeto locomotora con una marca "MEVA", un modelo "LSI0809" y un peso de 4500 kg, una capacidad de arrastre de 30.000 kg así como un vagón de carga máxima 3000 kg que deberá tener un peso de 1300 Kg y una carga de 500 Kg de madera., y un vagón de carga máxima 3000kg , con un peso de 1300kg y una carga de carbón de 3000 kg.

PARTE 2

OBJETIVOS

En este apartado vamos a trabajar con **estructuras de almacenamiento** para mantener en memoria los objetos que vamos creando sin necesidad de utilizar un atributo diferente para cada uno de ellos. Mediante estas estructuras de almacenamiento se podrá modelar un objeto compuesto por un número indeterminado de otros objetos. En nuestro caso, será un tren compuesto por una locomotora y varios vagones.

Los conceptos que se deben aplicar en esta parte incluyen la creación de estructuras (dependiendo del tipo de objeto que se vaya a almacenar), la inserción de objetos en este tipo de estructuras y la iteración sobre éstas. En esta parte también aprenderemos a instanciar clases creando objetos y usaremos un método especial denominado *main()*, método que hace de punto de partida para la ejecución de un programa.

REQUISITOS PREVIOS

Esta etapa requiere haber estudiado los temas 7, 8 y 9 del temario detallado de la asignatura, correspondientes al Capítulo 4, las secciones 5.1, 5.5, 5.11 y 5.13 del Capítulo 5, así como las secciones de 7.3 a 7.6 y el Apéndice E del libro base para la Unidad Didáctica II.

ENUNCIADO

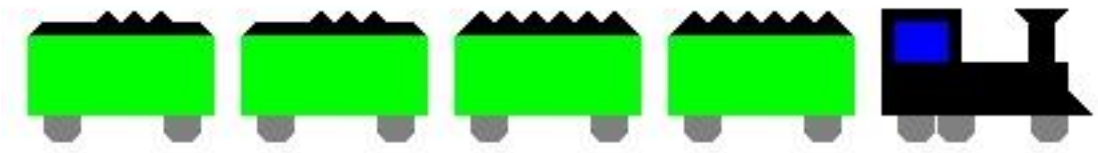
Queremos seguir completando la arquitectura de nuestro sistema de ferrocarriles incluyendo la creación de una clase tren. Como en el apartado anterior, esta clase tendrá asociada una serie de atributos y métodos que, entre otras funcionalidades, permitirán pintar una instancia de la clase tren en un objeto de la clase Pizarra, así como acceder a información sobre la carga que arrastra.

Un tren se caracteriza por contar con una locomotora capaz de arrastrar un conjunto no determinado a priori de vagones. Por tanto, esta clase tren deberá tener métodos que permitan el enganche y desenganche de vagones al tren.

Vamos a suponer que todos los vagones tienen el mismo peso máximo

Dado que los vagones que puede arrastrar nuestro tren tienen diferente tipo de carga y ésta tiene un precio en mercado distinto (el kilo de madera cuesta 500€, el de carbón 100€ y el de hierro 300€), el sistema deberá ser capaz de calcular el precio total de la carga del tren. Además, deberá poder acceder al peso total de la carga del tren y al peso de la mercancía que arrastra.

Dentro de esta parte de la práctica, deberá instanciarse y pintarse un objeto tren con cuatro vagones, de los cuales, dos de ellos deberán de ser de madera y contener un peso de carga de 500Kg, uno de carbón de 2000 Kg y, finalmente, un último vagón de hierro de 2500 Kg. de carga. La representación del tren puede ser algo similar a la siguiente:



El alumno deberá diseñar e implementar el esquema de clases, relaciones y métodos necesarios para poder pintar los objetos descritos, a la vez que mostrar por pantalla información relativa a:

- El **peso total del tren**, correspondiente al peso de la locomotora, los vagones y la carga del tren instanciado.
- El **peso total de la mercancía** que arrastra.
- El **valor total en euros de la mercancía** que arrastra. En el caso de que el peso total del tren supere los 5000 Kg, el valor de la mercancía se deberá incrementar en un 10%.
- NOTA : se recomienda pensar que el comportamiento general, es que un usuario indique qué quiere transportar y el sistema oferte los trenes necesarios.

PARTE 3

OBJETIVOS

En este apartado de la práctica se tratará de consolidar los conceptos aprendidos en las dos primeras partes, así como la correcta reutilización del código implementado en apartados previos.

También trabajaremos con el concepto de **herencia** y **polimorfismo**, ya que las estructuras de datos que utilizaremos contendrán instancias de diferentes clases, las cuales tienen métodos comunes (heredados), pero con implementaciones diferentes.

REQUISITOS PREVIOS

Esta etapa requiere haber estudiado los capítulos del libro base de las etapas anteriores, así como los temas 10, 11 y 12 del temario detallado de la asignatura, correspondientes a las Secciones de la 7.3 a la 7.6 ,y los Capítulos 6, 8 y 9 del libro base para la Unidad Didáctica II.

ENUNCIADO

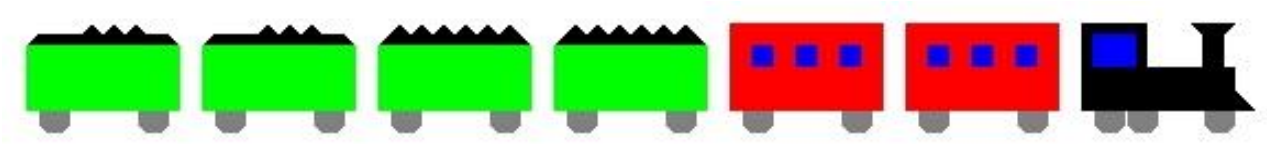
En esta parte de la práctica deberá rediseñarse el sistema, de modo que ahora se permitirá que los vagones arrastrados sean de dos tipos: de viajeros y de mercancía.

Los vagones de pasajeros deberán tener un peso de vagón, un número de viajeros y una capacidad máxima, mientras que los de mercancía mantendrán el peso de vagón, el peso de carga y el tipo de carga como en las etapas anteriores.

Hay que controlar, cuando se crea un tren, que no se sobrepase la capacidad de arrastre de la locomotora.

Deberá instanciarse un tren con cuatro vagones de mercancía, de los cuales, uno de ellos de madera con un peso de carga de 500Kg, uno de carbón de 1500 Kg y, finalmente, dos vagones de hierro de 2750 Kg. de carga. Todos los vagones tendrán un peso de vagón de 2000 Kg. Además, deberá de arrastrar dos vagones de pasajeros (con capacidad para 200 personas) con 150 y 145 personas respectivamente.

La representación del tren puede ser algo similar a la siguiente:



El alumno deberá diseñar e implementar el esquema de clases, relaciones y métodos necesarios para poder pintar los objetos descritos, a la vez que mostrar por pantalla información relativa a:

- El **valor total en euros de la mercancía** que arrastra. En el caso de que el peso total del tren supere los 5000 Kg, el valor de la mercancía se deberá incrementar en un 10%.
- El **peso total de la mercancía que arrastra.**
- **El valor de los billetes vendidos**, teniendo en cuenta que un pasaje en nuestro tren tiene un coste de 45€.
- El **peso total del tren**, correspondiente al peso de la locomotora, los vagones y la carga (materiales y viajeros) del tren instanciado, y considerando que, en término medio, una persona pesa 75 Kg.

Además, todo el código generado deberá estar documentado utilizando la herramienta Javadoc. La documentación debe incluir una breve descripción de cada clase, atributo y método. Para las clases y los atributos se debe describir lo que representa. En el caso de los métodos, debe especificarse los parámetros en entrada y salida con sus tipos y valores admitidos y el proceso que realiza.

NORMAS DE ENTREGA

FECHAS

La realización de la práctica se llevará a cabo en los centros asociados, siendo las sesiones presenciales organizadas y supervisadas por el tutor de la asignatura. Habrá como mínimo tres sesiones presenciales de obligatoria asistencia. En cada sesión se abordará cada una de las partes de las que consta la práctica. Los alumnos deberán ponerse en contacto con su centro asociado para informarse acerca de cuándo tendrán que asistir a las sesiones. Las fechas orientativas para la realización de cada una de las etapas serán:

- Finales de Noviembre. Realización de la primera parte de la práctica.
- Antes de Navidad. Realización de la segunda parte de la práctica.
- Después de Navidad. Realización de la tercera parte de la práctica.

DOCUMENTACIÓN A ENTREGAR

La entrega de la práctica tiene que constar de dos partes:

- Una memoria de **no más de 6 hojas** donde se explique la especificación y el diseño realizados en cada parte de la práctica.
- Para cada parte de la práctica se **deben entregar tanto los ficheros fuente como los ejecutables**.
- La documentación JavaDoc generada.

Los tutores de la asignatura deberán mandar un informe (*) y una calificación orientativa de cada alumno **antes del día 26 de Enero de 2009.** Deberán subir al curso virtual (**) un archivo comprimido con los códigos de todas las prácticas de los alumnos de su centro asociado, de modo que el equipo docente pueda revisarlas. Se usará software de detección de copias.

(*) Los informes se mandarán al equipo docente a través de una herramienta Web que estará disponible a partir de Enero.

(**) Los códigos comprimidos deberán subirse por medio de un mensaje en foro de tutores del entorno virtual.

NOTA IMPORTANTE: Los tutores tienen que cumplir una serie de requisitos ante los alumnos debido a que la práctica cuenta para la calificación de la asignatura. Por tanto antes de entregar las calificaciones al equipo docente deberán:

1. Publicar la nota de las prácticas en un lugar accesible para los alumnos (ya sea vía web o mandar un fax al centro asociado).

2. Establecer un día de revisión de prácticas (previo al periodo de entrega de las calificaciones al equipo docente), dado que éstas forman parte de la evaluación del alumno.

EVALUACIÓN DE LA PRÁCTICA

Los alumnos que quieran realizar el examen previamente tienen que haberse presentado a las prácticas y tener un aprobado en éstas. El informe del tutor se considera a efectos de subir nota.

Las prácticas no se guardan de un curso para otro.

PREGUNTAS AL EQUIPO DOCENTE

El equipo docente atenderá preguntas de carácter metodológico y de diseño, preferentemente a través del foro y el tablón de anuncios del curso virtual. **Las preguntas relativas a la instalación del entorno de desarrollo, puesta en funcionamiento y errores de compilación deben ser tratadas con los tutores de los centros asociados.**

Felisa Verdejo Maillo, Catedrática

Tutorías: Jueves de 16:00 a 20:00

Teléfono: 91.398.64.84

Mail: felisa@lsi.uned.es

Enrique Amigó Cabrera, Profesor Ayudante

Tutorías: Jueves de 16:00 a 20:00

Teléfono: 91.398.86.51

Mail: enrique@lsi.uned.es

Víctor Fresno Fernández, Profesor Ayudante Doctor

Tutorías: Jueves de 16:00 a 20:00

Teléfono: 91.398.82.17

Mail: vfresno@lsi.uned.es

Alberto Pérez García-Plaza, Profesor Ayudante

Tutorías: Jueves de 16:00 a 20:00

Teléfono: 91.398.79.57

Mail: alpgarcia@lsi.uned.es