UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR

DE

INGENIERÍA Y SISTEMAS DE TELECOMUNICACIÓN

PROGRAMACIÓN II

BLOQUE I de prácticas

Práctica 2

Semestre de primavera (curso 2013 – 2014)

ÍNDICE

Práctica 2

Los objetivos que se pretenden cubrir con el desarrollo de esta práctica son básicamente los siguientes:

1. Aplicar los conceptos básicos de la programación orientada a objetos (POO)
2. Crear y usar clases y objetos.

Esta práctica tiene una duración de 1 semana.

Enunciado

Una empresa necesita almacenar líquidos y para ello elige hacerlo utilizando depósitos de base cuadrada y de alto variable. Necesita también tener una aplicación de gestión de los depósitos que permita conocer las cantidades almacenadas en cada uno de ellos.

Para codificar la aplicación se ha pensado en crear una clase Deposito que modela los depósitos que se van a utilizar. Dicha clase dispondrá de los siguientes métodos, tal y como se muestran en la documentación generada por javadoc:

Constructor Detail

Deposito

```
public Deposito(java.lang.String codigo,
        double lado,
        double alto)
```

Crea un depósito con un código que lo identifica, con una base cuadrada de lado 'lado' y con una altura de 'alto' metros.

Parameters:

`codigo` - Código usado para identificar al depósito

`lado` - Dimensiones de los lados de la base cuadrada del depósito en metros

`alto` - Alto del depósito en metros.

Method Detail

getCodigo

```
public java.lang.String getCodigo()
```

Devuelve el código asignado al depósito.

Returns:

Código del depósito.

getCapacidad

```
public double getCapacidad()
```

Devuelve la capacidad máxima del depósito en litros.

Returns:

Capacidad del depósito cuando está vacío.

getCuantoCabeAun

```
public double getCuantoCabeAun()
```

Devuelve el número de litros que aun caben en el depósito hasta llenarlo.

Returns:

Número de litros que aun caben en el depósito.

vaciar

```
public boolean vaciar(double litros)
```

Vacía el depósito en la cantidad de litros que se le pasan como parámetro. La acción de vaciar se traduce en restar al número de litros que contiene el depósito los litros que se le pasan como parámetro. Si el número de litros a vaciar es mayor que la cantidad existente se devolverá el valor false y el contenido del depósito permanecerá inalterado. Si se consiguen vaciar los litros solicitados, se devolverá el valor true.

Parameters:

`litros` - Litros que se han de vaciar del depósito.

Returns:

true Si el depósito contiene un número de litros mayor o igual que los que se han solicitado vaciar. Devuelve false si el depósito contiene una cantidad inferior.

llenar

```
public boolean llenar(double litros)
```

Añade al depósito la cantidad de litros que se pasa como parámetro. Si la cantidad de litros es menor o igual que 0 o no caben todos en el depósito no realiza la operación y devuelve false.

Parameters:

`litros` - Número de litros a llenar en el depósito.

Returns:

true si se ha completado la operación con éxito; false en caso contrario.

transvasar

```
public boolean transvasar(double litros,
                          Deposito depositoDestino)
```

Extrae una cantidad de litros de líquido del depósito actual y lo transvasa al depositoDestino. Si la cantidad a transvasar es menor o igual que 0, si el depositoDestino es null o en depositoDestino no caben todos los litros no se realiza el transvase y devuelve false. Si el depósito origen contiene una cantidad de

litros inferior a los indicados para transvasar no se transvasará volumen alguno y devuelve false. Si se consigue realizar el transvase se devuelve true.

Parameters:

`litros` - volumen a transferir.

`depositoDestino` - depósito al que se transvasa el volumen indicado en litros.

Returns:

true si se ha podido transvasar y false en caso contrario.

toString

```
public java.lang.String toString()
```

Muestra un mensaje con el estado del depósito.

Overrides:

`toString` in class `java.lang.Object`

Returns:

Depósito: 7A, capacidad: 10000,0 litros, contiene: 250,0 litros

Recursos

Para el desarrollo de esta práctica se aporta en la plataforma moodle lo siguiente:

- Documentación generada por *javadoc* de la clase ***Deposito*.**
- El código de un programa de prueba que se describe más adelante en la clase **TesterDeposito.**

Prerrequisitos

- Leer el enunciado completo de la práctica.
- Leer la documentación disponible generada por *javadoc*.
- Realizar una primera versión del código Java de la clase ***Deposito*** que se llevará ya lista al laboratorio para su revisión al inicio de la práctica. Deberá tener como mínimo la cabecera de los métodos y los comentarios para JavaDoc. El cuerpo de los métodos puede estar vacío, pero deberá compilar sin errores. De no cumplir esta condición se restará hasta un 20% del valor máximo de la nota asignada a esta práctica.

Desarrollo de la práctica

En la sesión de trabajo, el alumno realizará las siguientes actividades:

1. Completar la codificación de la clase **Deposito** que se ha llevado al laboratorio. Dicha clase deberá codificarse de forma que se añada la información necesaria para que al generar la documentación con javadoc aparezca de forma similar a como se muestra en el enunciado. Los parámetros y valores devueltos por cada método también deben documentarse.
2. Codificar la clase **AplicacionDepositos** con un pequeño programa principal que tenga la siguiente funcionalidad:

a. Creará dos depósitos, o sea dos objetos de clase Deposito. Al primero de los depósitos se le asignará el código Dep1 y tendrá un lado de 1m y un alto de 2m. El segundo de los depósitos se llamará Dep2 y tendrá un lado y un alto de 1m.

b. Llenará con 1000 litros cada uno de los depósitos creados.

c. Transvasar dos tercios del contenido de Dep2 a Dep1. Para hacerlo, se deberá averiguar, usando el método private static double contiene(Deposito dep) que se describe más abajo, cual es la cantidad de líquido que queda en Dep2. En la realización este apartado y los siguientes se deberá comprobar el éxito o no de la operación transvasar, sacando un mensaje en la pantalla que lo indique.

d. Transvasar todo el contenido de Dep1 a Dep2. Se deberán seguir las mismas instrucciones que las indicadas en el apartado anterior.

e. Transvasar de Dep1 a Dep2 la cantidad necesaria para llenar Dep2. Usar las mismas instrucciones que las usadas en los dos puntos anteriores.

Finalizada esta secuencia de acciones Dep1 deberá contener la misma cantidad de líquido que Dep2: 1000 litros ¿Se ha perdido líquido? ¿Por qué?

Para la realización de la aplicación AplicacionDepositos, además del main, se deberá implementar el método:

```
/* Devuelve el volumen de litros que contiene el depósito que se le pasa como
* parámetro
* @param dep Depósito del que desea conocer el volumen de líquido que contiene
* @return Cantidad de líquido que contiene el depósito.
*/
    private static double contiene(Deposito dep){
           // completar por el alumno
    }
```

3. Utilizar la clase **TesterDeposito** que se proporciona ya codificada y que permite comprobar realizando diversas pruebas la correcta implementación de la clase Deposito. Los programas de pruebas son parte de la metodología de desarrollo de software, y normalmente los realiza el mismo programador que escribe el código, pero en este caso se le proporciona utilizando por simplicidad el mecanismo estándar de Java de las aserciones (http://docs.oracle.com/javase/6/docs/technotes/guides/language/assert.html).

   Para compilar y ejecutar el programa anterior ejecute los siguientes mandatos:

```
D:\pruebas>javac TesterDeposito.java
D:\pruebas>java –ea TesterDeposito
Tests de Deposito pasados correctamente.
```

   Si al ejecutar el programa de pruebas no indica que haya algún error y aparece el mensaje anterior entonces su código pasa las pruebas correctamente. La clase Deposito debe encontrarse en el mismo directorio que el programa de pruebas.

NOTA: la clase Scanner lee los números decimales según el formato local. Si recibe un error al introducir los números decimales con puntos, introdúzcalos con comas.

Plazos de realización, entrega y documentación a entregar

La práctica 2 se realizará en la semana lectiva 5 que comienza el 7 de marzo y finaliza el 13 de marzo

Los archivos que son solución a la práctica 2 de este bloque de prácticas deberán entregarse a través de la plataforma Moodle. El alumno subirá cada entrega en un fichero ZIP a la plataforma Moodle con las siguientes características:

a) El archivo ZIP contendrá las clases *Deposito.java* y *AplicacionDeposito.java*
b) El nombre del fichero ZIP debe tener el formato:
   grupolaboratorio_B1_P2_ apellidos_nombre.zip
   Por ejemplo, un alumno que se llamara Rafa Nadal y que estuviera en el grupo V4 subiría a la plataforma Moodle el fichero:
   V4_B1_P2_Nadal_Rafa.

La entrega en Moodle deberá hacerse con plazo máximo de 96 horas (cuatro días) después de finalizar la sesión de laboratorio (Por ejemplo el grupo V4 -viernes de 15:30 a 17:30- podría entregar como máximo hasta las 17:30 del martes de la semana siguiente).

Evaluación

Durante el proceso de codificación se tendrán en cuenta los siguientes requisitos que se deben cumplir para que la práctica sea evaluada:

- En la codificación de la clase **Deposito** no se realizará **ninguna operación de entrada/salida por terminal con el usuario**.
- Todas las **variables miembro** de la clase **Deposito** serán declaradas con acceso **privado** (private) y siempre como variables de instancia (sin static).
- Se comentarán las clases y todos los métodos adecuadamente (descripción, argumentos, valor de retorno si lo hay) para la generación de documentación mediante la herramienta **javadoc**. De no hacerse correctamente se tendrá una penalización de hasta un 30% sobre el peso total de esta práctica.
- Se utilizarán las normas de estilo de Java (mayúsculas, etc.) para los identificadores de clases, atributos, métodos, etc.

Para la evaluación de esta práctica es condición imprescindible que funcione perfectamente, o sea, que pase todas las pruebas del programa de pruebas que se proporciona. **Si no es así se obtendrá la calificación de 0 puntos en esta práctica.**

La práctica se evaluará al final del bloque I junto con el resto de prácticas de este bloque. El peso total en la evaluación del bloque I será de un 30% de la nota total del bloque.