Programación Avanzada

EXAMEN FEBRERO 2015

Por favor siga las siguientes indicaciones:

- Escriba con las hojas de un solo lado.
- Escriba su nombre y número de documento en todas las hojas que entregue.
- Numere las hojas e indique el total de hojas en la primera de ellas.

Problema 1 (33 puntos)

Una agencia de publicidad desea invertir en la construcción de un software que le permita gestionar las campañas de publicidad. Cada campaña involucra dos etapas: primero se configura y luego se implementa.

La definición de la campaña implica elegir qué medios de comunicación se utilizarán. Existen tres tipos de medios de comunicación: TV (interesando saber la cantidad mínima y máxima de minutos de aire permitidos), Internet (interesando saber el canal que se puede utilizar, pudiendo ser Facebook, Twitter o un Portal) y Diario (interesando saber el vendedor con el cual hay que contactarse). Todos los medios tienen un nombre y una descripción.

La implementación de la campaña implica la definición de las pautas que se utilizarán para cada medio. Una pauta contiene la fecha en la cual debe salir y un código, y existen análogamente tres tipos de pautas (pauta para TV, pauta para Internet y pauta para Diario). Por tanto, si en la campaña se configuró el uso de un diario (es decir un medio de comunicación de tipo Diario), entonces al momento de implementar la campaña se debe definir una pauta de tipo Diario, indicando el texto que debe salir impreso en el mismo. Análogamente, si en la campaña se configuró el uso de TV, entonces la implementación de la campaña debe definir una pauta para ese medio, incluyendo la cantidad de minutos de aire (la cual debe ser coherente con la cantidad mínima y máxima de minutos definidos en el medio). Finalmente, si en la campaña se configuró el uso de Internet, entonces la implementación de la campaña debe definir una pauta para ese medio, incluyendo una imagen a ser mostrada en Internet (basta con guardar solo la URL).

Se han definido dos Casos de Uso: Configurar Campaña e Implementar Campaña. A continuación se describe el segundo:

Caso de Uso:	Implementar Campaña
Actores:	Usuario
Descripción:	Este Caso de Uso comienza cuando el usuario selecciona una campaña (ingresando el ID de la misma). Luego se ingresan todas las pautas, una por cada medio definido (eligiendo el medio correspondiente mediante su nombre). Si bien todas las pautas tienen fecha y código, estos datos serán calculados internamente por el Sistema (la fecha será la fecha actual del Sistema y el código será autogenerado). Para las pautas en TV se debe ingresar la cantidad de minutos de aire. Para las pautas en Internet se debe ingresar la imagen (URL). Para las pautas en Diario se debe ingresar el texto. El Caso de Uso termina cuando el usuario confirma la implementación de la campaña.

Se pide:

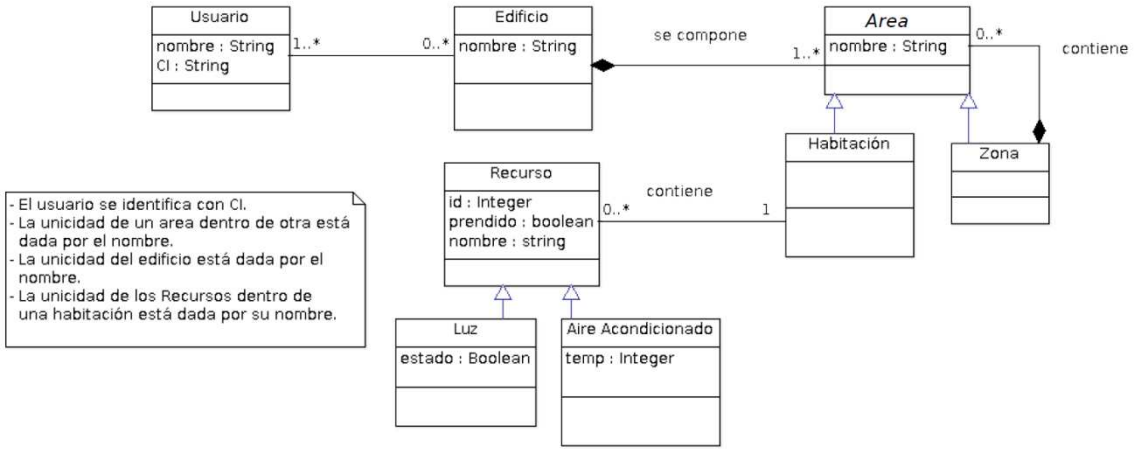
i. Realizar el **Modelo de Dominio** de dicha realidad, incluyendo restricciones en lenguaje natural.

ii. Realizar el **Diagrama de Secuencia del Sistema** correspondiente al **Caso de Uso Implementar Campaña**, incluyendo Datatypes en caso de ser necesario y especificando el uso o no de la memoria del Sistema.

Problema 2 (33 puntos)

PALazyLife es una aplicación que permite manejar tu casa desde la comodidad de un pc o cualquier dispositivo móvil. En el programa pueden configurarse tantos edificios como se desee y asignarle usuarios a cada uno de ellos. Permite configurar su distribución espacial y los recursos contenidos en cada habitación. Cada habitación queda definida de forma global por su path (Ejemplo de path: 'PrimerPiso/Cocina/').

PALazyLife cuenta con dos tipos de recursos, Luz y AireAcondicionado. Luz es capaz de prender y apagar, mientras que AireAcondicionado añade la opción de configurar una temperatura objetivo.



Se consideran los casos de uso crearRecurso, prenderApagarRecurso y crearEdificio, cada uno de los cuales es modelado con una sola operación del sistema, cuyos contratos se especifican a continuación:

`CrearRecurso( edificio: string, path: string, luz: boolean, nombre: string)`	
Desc	Se crea un recurso de tipo luz o aireAcondicionado (instancia de luz si **luz** = true) en la ubicación **path** con el nombre **nombre** en el edificio **edificio**.
Pre	- No existe otro recurso con nombre **nombre** en la ruta **path**. - Existen las instancias de area y habitacion tal que la ruta **path** existe en el sistema para el edificio **edificio**.
Post	- Se crea una instancia de luz o aireAcondicionado, según corresponda. - Se crea un link entre la habitacion correspondiente al último tramo (Ultima zona del path) del **path** y el recurso creado, para el edificio **edificio**.

`crearEdificio(nombre_e: string, nombre_a, ci_u: string)`	
Desc	Se crea un Edificio con nombre **nombre_e** y se crea una zona con nombre **nombre_a** y se asocia al Usuario de cédula **ci_u**.
Pre	- No existe una instancia de Edificio con nombre **nombre_e** en el sistema. - Existe una instancia de Usuario con cédula **ci_u** en el sistema.
Post	- Se crea una instancia de Zona con nombre **nombre_a**. - Se crea una instancia de Edificio con nombre **nombre_e**.. - Se asocian las instancias de Edificio y la instancia de Usuario con cédula **ci_u**. - Se asocian la nueva instancia de Edificio y la nueva instancia de Zona.

`prenderApagarRecurso(edificio:string, path: string, nombre: string, on: boolean)`	
Desc	Si on == true se prende el recurso en la ubicación **path** con el nombre **nombre** en el edificio **edificio**, si on == false se debe apagar.
Pre	No existe otro recurso con nombre **nombre** en la ruta **path** para el edificio **edificio**.
Post	Se asigna **on** al atributo **prendido** del recurso con nombre **nombre**, asociado a la habitación correspondiente al último tramo de la ruta **path** para el edificio **edificio**.

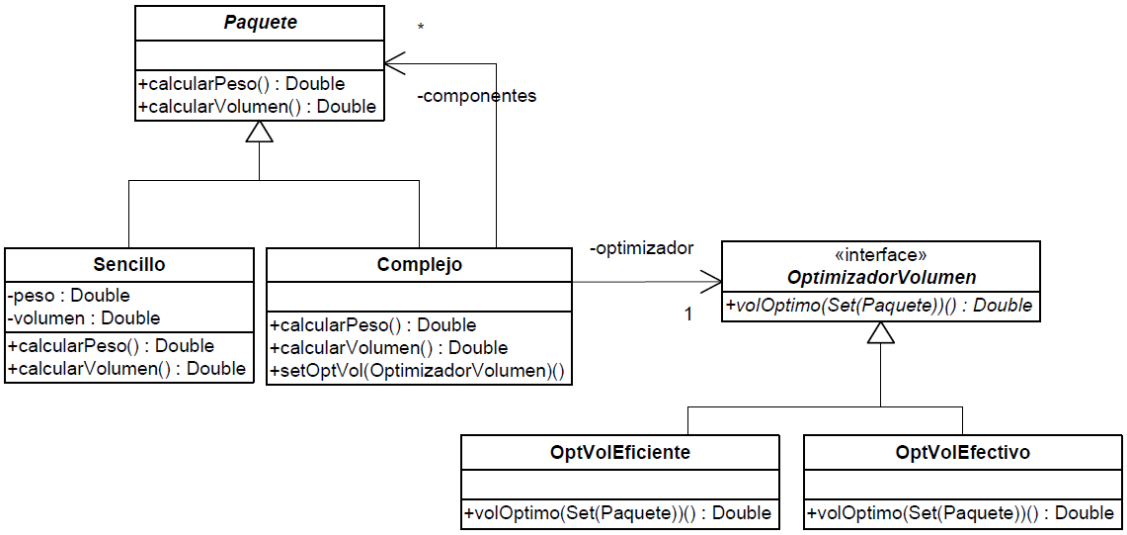
Se pide:

i) Realizar el **Diagrama de Comunicación** para las operaciones previamente declaradas.
ii) Realizar el **Modelo de Clase de Diseño** correspondiente.

Problema 3 (34 puntos)

i) Implementar completamente en C++ una clase singleton denominada `A`, que tiene un atributo privado de tipo int, denominado `dato`.

ii) Una empresa de servicios de transporte interurbano de objetos frágiles, ha encomendado el desarrollo de un sistema de apoyo a la gestión en el armado de los paquetes chicos y medianos. Los objetos son empaquetados en cajas, ya sea de forma individual o agrupados. Los paquetes así conformados, pueden a su vez ser parte de otros paquetes más grandes. La figura muestra el diseño de la estructura determinada para representar la organización de los paquetes.



Para organizar la carga en los vehículos, es de interés poder estimar el peso y volumen de los paquetes. El cálculo del peso es relativamente sencillo, mientras que el del volumen implica la ejecución de un algoritmo de optimización de ubicación de volúmenes, que tiene un alto costo computacional. Por este motivo, para la estimación del volumen de un paquete que está formado por otros, se utiliza un optimizador de volúmenes. Existen varias versiones de dicho optimizador: algunas estiman de mejor forma el volumen con un alto costo computacional (efectivas), mientras que otras hacen una estimación menos ajustada pero ejecutan de manera más rápida (eficientes).

A su vez, se deben tener en cuenta las siguientes consideraciones:

- El peso y el volumen de un paquete `Sencillo` se calculan directamente a partir de sus atributos.
- El peso de un paquete `Complejo` es la suma de los pesos de sus componentes.

- El volumen de un paquete `Complejo` se calcula invocando a la operación `valOptimo` de la interfaz `OptimizadorVolumen`, la cual recibe una colección con los componentes del paquete en cuestión.
- La construcción de un paquete `Complejo` se realiza a partir de sus componentes y de una instancia de una clase que implemente la interfaz `OptimizadorVolumen`.
- La destrucción de un paquete `Complejo` implica la destrucción de todos sus componentes.

Se pide:

Implementar completamente en C++ las clases `Paquete`, `Sencillo`, `Complejo` y la interfaz `OptimizadorVolumen`.

Observaciones:

- Puede suponer la existencia de la interfaz `ICollectible` e implementaciones de `ICollection` (clase `List`) e `IIterator` según sea necesario.
- Es posible utilizar las clases `set<T>` o `vector<T>` de la STL.
- Las implementaciones deben incluir constructores, destructores y operaciones set y get donde corresponda.
- No incluir directivas al precompilador.