Programación 4

EXAMEN JULIO 2020

Por favor siga las siguientes indicaciones:

- Escriba con lápiz y de un solo lado de las hojas.
- 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.
- Recuerde entregar su número de examen junto al examen.
- Está prohibido el uso de computadoras, tabletas o teléfonos durante el parcial.

Problema 1 (30 puntos)

Se desea crear un sistema para la gestión de historias médicas, registro de reservas de consultas y almacenamiento de información de usuarios. Existen tres categorías de usuarios del sistema: socios, médicos y administrativos. Los usuarios pueden estar activos o inactivos.

Para comenzar a utilizar el sistema, los usuarios son registrados especificando su cédula (que los identifica), sexo y fecha de nacimiento. Además, de los usuarios se conoce su nombre y apellido, así como su edad, calculada a partir de la fecha de nacimiento. Además, se debe especificar la categoría a la que pertenece el usuario (socio, administrativo, médico). Tener en cuenta que puede haber usuarios con dos categorías. La única combinación que no está permitida por la institución es la de ser administrativo y médico a la vez. Se requiere conocer los usuarios dados de alta por los usuarios administrativos.

Los socios pueden hacer reservas de consultas comunes, de las cuales interesa conocer la fecha de reserva, fecha y hora de consulta, y el médico tratante. Además, los socios pueden hacer consultas de emergencia, las cuales no tienen reservas. De estas consultas interesa conocer la fecha y hora de la consulta, así como el motivo de la misma.

Los médicos deben llevar registro de la asistencia de los socios a una consulta (de cualquier tipo). En una consulta el médico puede dar múltiples diagnósticos y para cada uno indicar, si corresponde, varios tratamientos. Un diagnóstico está definido por un nombre del problema de salud y una descripción libre que le permite al médico extenderse como lo desee. Los tratamientos tienen una descripción y pueden opcionalmente requerir un médico tratante.

Por último, se desea mantener un historial para cada paciente, donde se conozcan todas las consultas realizadas junto con el médico tratante, los diagnósticos y los tratamientos asignados, si es que éstos existieron.

Se pide:

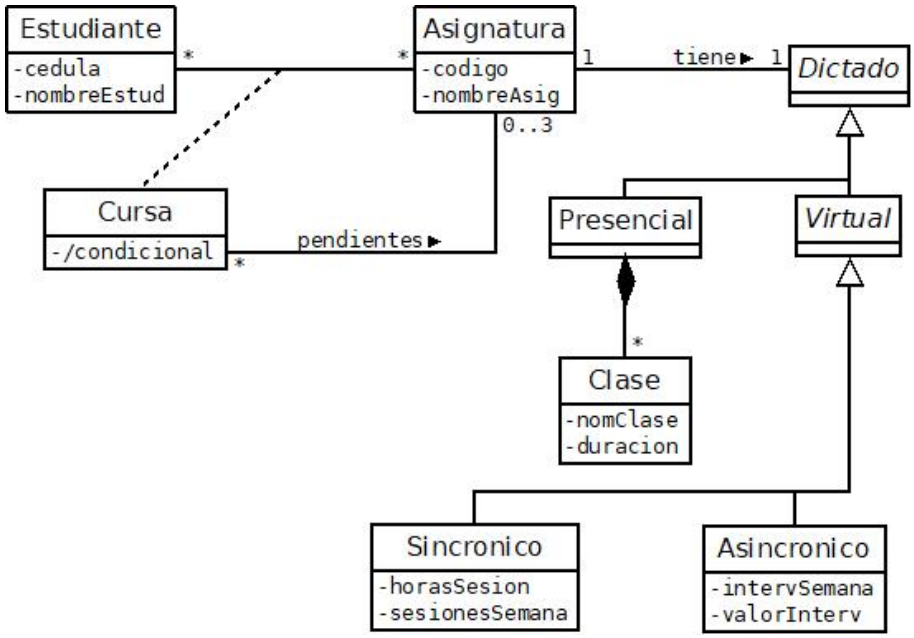
a) Construir el Modelo de Dominio a partir de la visión de la descripción planteada. Presentarlo en un diagrama utilizando UML
b) Expresar cada una de las restricciones del modelo en lenguaje natural.

Problema 2 (35 puntos)

Debido a la emergencia sanitaria declarada por la pandemia de COVID-19, han ocurrido cambios en la modalidad de dictado de las clases en la facultad. En este nuevo escenario se desea estimar las horas de dedicación de los estudiantes a las diferentes asignaturas que cursan. Para ello se ha realizado el modelo de dominio de la figura (siguiente página), que representa la situación de los estudiantes en un semestre determinado.

Los estudiantes cursan asignaturas. Dado que se han flexibilizado algunas condiciones, para cada asignatura que cursa un estudiante, puede tener hasta tres asignaturas previas pendientes de aprobación; el valor del atributo booleano `condicional` depende de si el estudiante tiene alguna asignatura pendiente (para la que está cursando) o si no tiene ninguna. Cada asignatura tiene información de la carga horaria semanal según la modalidad de dictado.

Si la modalidad es presencial, se tiene un conjunto de clases semanales (por ejemplo, teórico 1, teórico 2, práctico y monitoreo), cada una con su duración. Si la modalidad es virtual sincrónica, se sabe cuantas horas dura cada sesión y la cantidad de sesiones por semana. Si la modalidad es virtual asincrónica, se sabe cuantas intervenciones por semana se espera que haga el estudiante y cuanto tiempo se estima que insume cada intervención.



Se pide:

a) Realizar los Diagramas de Comunicación (indicando visibilidades) de las operaciones del sistema dadas por los siguientes contratos.

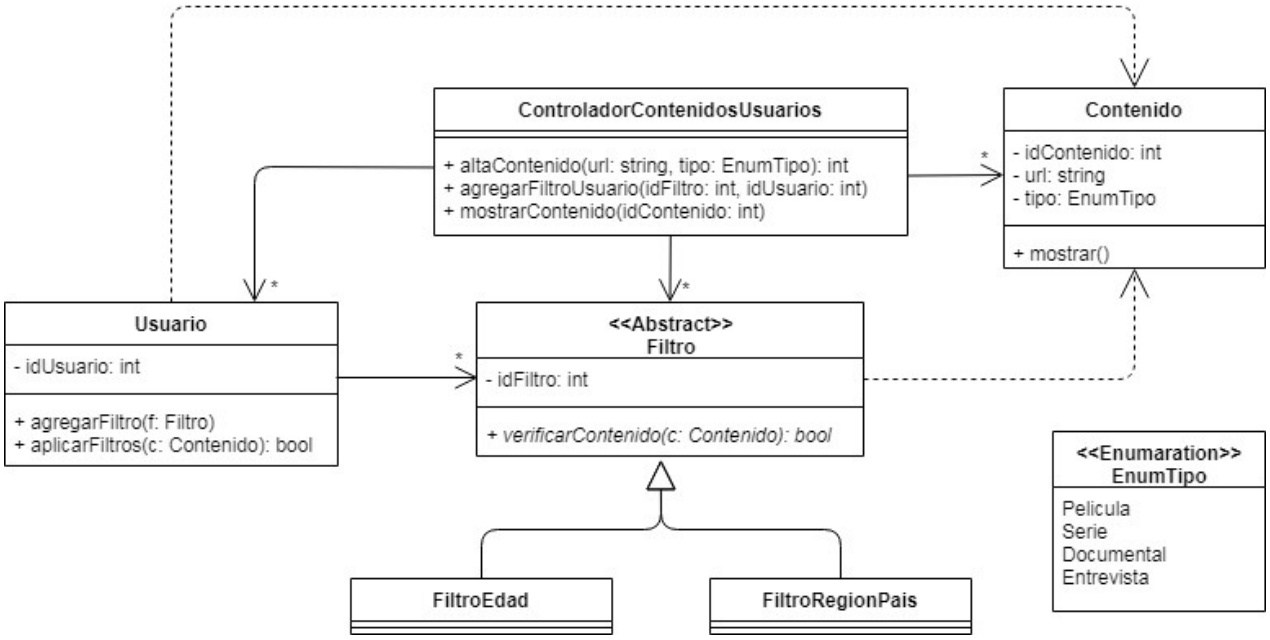
`inscribir(ciE:Integer, codA:Integer, pends:Set(Integer))`	
Descripción	Inscribe un estudiante a una asignatura, eventualmente con asignaturas pendientes de aprobación.
Parámetros	- `ciE`: Cédula de identidad del estudiante - `codA`: Código de la asignatura a inscribirse - `pends`: Conjunto de códigos de asignaturas pendientes
Precondiciones	- Existe en el sistema un estudiante con cédula `ciE` - Existe en el sistema una asignatura con código `codA` - Por cada elemento `e` de `pends`, existe una asignatura en el sistema con código `e` - No existe en el sistema un link de `Cursa` entre el estudiante con cédula `ciE` y la asignatura con código `codA`
Postcondiciones	- Existe en el sistema una nueva instancia de `Cursa`, asociada al estudiante de cédula `ciE` y a la asignatura de código `codA` - El valor del atributo `condicional` de la nueva instancia de `Cursa` es igual a `FALSE` si el conjunto `pends` es vacío, de lo contrario es igual a `TRUE`. - Si el atributo `condicional` tiene el valor `TRUE`, existe en el sistema un nuevo link entre cada asignatura correspondiente a los elementos del conjunto `pends` y la nueva instancia de `Cursa`

`horasSemDedicacion(ciE:Integer):Real`	
Descripción	Devuelve una estimación del total de horas semanales de dedicación de un estudiante a las asignaturas en que está inscripto
Parámetros	- `ciE`: Cédula de identidad del estudiante
Precondiciones	- Existe en el sistema un estudiante con cédula `ciE`
Postcondiciones	- El resultado es la suma de las dedicaciones semanales de las asignaturas a las que está inscripto el estudiante con cédula `ciE`. - La dedicación de una asignatura depende de su modalidad de dictado: o Si es presencial, es la suma de las duraciones de todas sus clases. o Si es virtual sincrónica es el producto de las horas de sesión por la cantidad de sesiones. o Si es virtual asincrónica es el producto de las intervenciones estimadas por semana por el valor en horas de la intervención.

b) Realizar el Diagrama de Clases de Diseño resultante de su solución a la parte a). El diagrama debe incluir todos los atributos y sus tipos, todas las operaciones y los tipos de sus parámetros y retornos, además de el o los controladores que surjan del diseño.

Problema 3 (35 puntos)

NetFing es una plataforma para usuarios que deseen acceder a contenidos audiovisuales. En la figura se presenta el Diagrama de Clases de Diseño (DCD) parcial del subsistema de gestión de contenidos. Los signos de + y – en las clases representan si la operación es pública o privada respectivamente. Este subsistema permite dar de alta contenidos (por el momento no se darán de baja los contenidos) y también mostrar los mismos a los usuarios; los contenidos deberán mostrarse solo si no son filtrados. Inicialmente existen 2 tipos de filtros, pero es posible que a futuro se agreguen nuevos.



El comportamiento de las operaciones se describe a continuación (siguiente página):

`ControladorContenidosUsuarios:`

- `altaContenido(url: string, tipo: EnumTipo)`: Crea un nuevo contenido con atributo `url` igual al parámetro `url`, atributo `tipo` igual al parámetro `tipo` y atributo `idContenido` autogenerado. Se retorna `idContenido`.
- `agregarFiltroUsuario(idFiltro: int, idUsuario: int)`: Agrega el filtro con atributo `idFiltro` igual al parámetro `idFiltro` al usuario con atributo `idUsuario` igual al parámetro `idUsuario`.
- `mostrarContenido(idContenido: int)`: Muestra el contenido con atributo `idContenido` igual al parámetro `idContenido` a todos los usuarios para los cuales la operación `aplicarFiltros`, aplicada a dicho contenido, retorne `false`. Para hacerlo invoca a la operación `mostrar` de la clase `Contenido`.

`Contenido:`

- `mostrar()`: Esta operación es la que se encarga efectivamente de mostrar el contenido al usuario. Se asume implementada.

`Usuario:`

- `agregarFiltro(f: Filtro)`: Agrega el filtro `f` al usuario.
- `aplicarFiltros(c: Contenido)`: Devuelve `false` si y solo si el resultado de `verificarContenido` para `c` es `false` para todos los filtros del usuario.

`Filtro:`

- `verificarContenido(c: Contenido)`: Indica si corresponde aplicar el filtro para este contenido. En caso afirmativo devuelve `true` y no se mostrará al usuario el contenido.

Se pide:

a) ¿En el DCD presentado identifica el uso de algún patrón de diseño? En caso afirmativo, indique qué tipo de problema se está resolviendo y qué clase cumple cada rol.
b) Implementar los siguientes archivos:

.cpp	.h
ControladorContenidosUsuarios	ControladorContenidosUsuarios
Contenido	Contenido
Usuario	Usuario
---	Filtro
---	FiltroRegionPais

Considerar:

- Es posible utilizar las clases `set<T>`, `map<K,V>` y `vector<T>` de STL.
- Puede suponer la existencia de la interface `ICollectible` e implementaciones de `ICollection` (clase `List`) e `IIterator` según sea necesario.
- Las implementaciones **deben** incluir constructores y destructores, liberando en estos últimos toda la memoria que sea necesaria.
- Asumir existencia y no implementar los get y set de los atributos.
- **No** incluir directivas al precompilador (#include, #define, etc).
- No implementar los datatypes.