

# Programación 4

## EXAMEN FEBRERO 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 la prueba.

### **Problema 1 (30 puntos)**

Para facilitar la integración entre los estudiantes de la FING, se le ha propuesto a su Equipo de Desarrollo la construcción de una aplicación para conocer gente llamada “Fing-Connect”. La misma cuenta con un perfil por estudiante en el cual aparece el nombre de pila (que puede repetirse), la edad, el género del estudiante (por simplificación se comenzará con Masculino y Femenino) y un conjunto de fotos (por lo menos una es obligatorio colocar). Además, cada perfil cuenta con un mail (único) y una contraseña para habilitar el ingreso a la aplicación.

Con el objetivo de conectar gente que compartan ciertas cosas, cada perfil de estudiante está asociado al conjunto de asignaturas que él/ella está cursando, de modo que la aplicación sólo le sugerirá perfiles de otros estudiantes que también estén cursando al menos una asignatura en común y que respeten el rango de edades así como el género de interés especificado. Mientras que la información de las asignaturas se tomará de Bedelía (nombre de la asignatura y fecha de inicio y de fin), la información sobre las preferencias de edad y género deben ser ingresadas por el estudiante.

Una vez que la aplicación identifica dos estudiantes cuyas preferencias coincidan y que además comparten al menos una asignatura, se desplegará una sugerencia de conexión (o *match*) para ese par de estudiantes. A cada sugerencia, el estudiante puede o bien ignorar o bien dar aprobación (o *like*).

Sólo si ambas personas dan like, entonces se genera un chat entre ambos, el cual estará compuesto de mensajes solo de texto, todo gestionado dentro de la aplicación. Interesa saber la fecha y hora exacta de cada mensaje (llegando al segundo de precisión), así como la fecha en la cual comenzaron a chatear.

Además, se consideran los siguientes Casos de Uso (CU):

Caso de Uso:	Enviar Mensaje
Actor:	Estudiante
Descripción:	El CU comienza cuando el Estudiante ingresa a la aplicación, indicando su mail (que lo identifica) y la contraseña correspondiente. Si el ingreso es satisfactorio, entonces el Estudiante solicita una lista de todos sus matches para luego elegir (mediante el nombre) aquel con el cual desea chatear. Finalmente se puede enviar el mensaje (solo texto) o bien cancelar el CU.

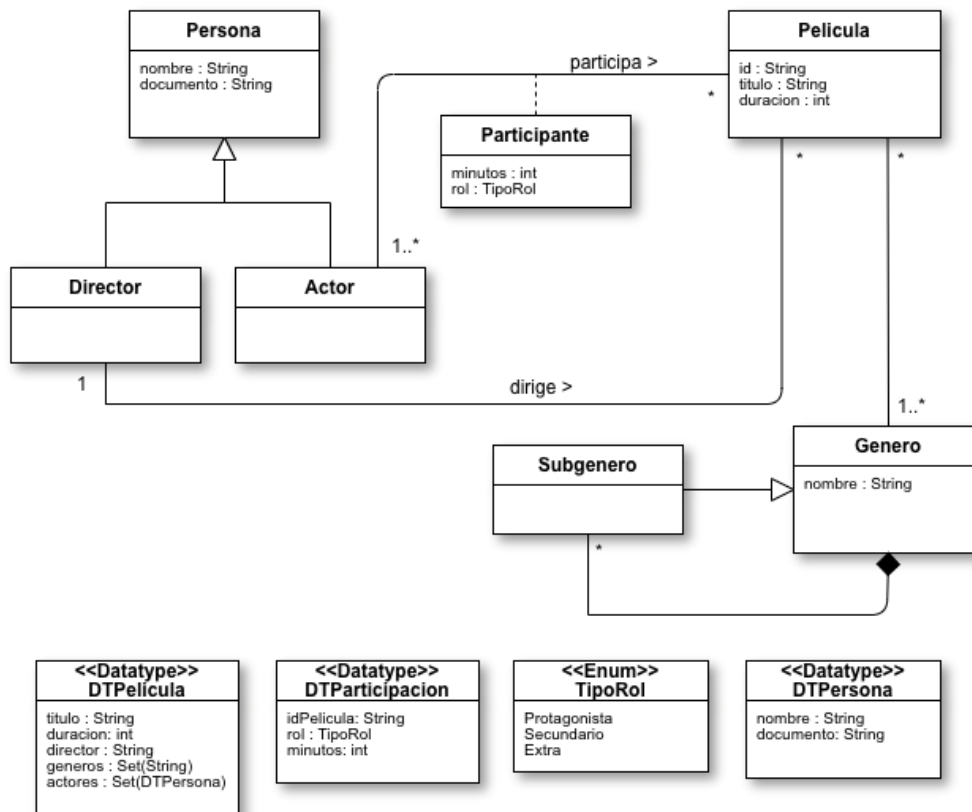
Caso de Uso:	Revisar Sugerencias
Actor:	Estudiante
Descripción:	El CU comienza cuando el Estudiante ingresa a la aplicación, indicando su mail (que lo identifica) y la contraseña correspondiente. Si el ingreso es satisfactorio, entonces el Estudiante solicita una lista de las sugerencias preparada por el Sistema. Luego, para cada perfil sugerido, el Estudiante puede o bien dar like o bien ignorar. Esto se repite para cada perfil sugerido o bien hasta que el Estudiante cancele el CU.

**Se pide:**

- Realizar el Modelo de Dominio de la realidad anterior, incluyendo restricciones en lenguaje natural.
- Realizar los Diagramas de Secuencia de Sistema correspondientes a los CU Enviar Mensaje y Revisar Sugerencias, incluyendo manejo de memoria del Sistema y datatypes si fuera necesario.

**Problema 2 (35 puntos)**

Considere el siguiente Modelo de Dominio para un sistema de registro de películas de cine con sus respectivos actores, director y género al que pertenecen.



**Restricciones:**

- El atributo id identifica a la Película.
- El atributo nombre identifica al Género.
- El atributo documento identifica a la Persona.
- No existe un género que sea subgénero de sí mismo.
- Si un género es subgénero de otro, este no puede ser subgénero del anterior.

**Se pide:**

- i) Realizar los Diagramas de Comunicación correspondientes a las siguientes operaciones del sistema, **incluyendo visibilidades:**

<code>registrarActor(dtp: DtPersona, participaciones: Set&lt;DTParticipacion&gt;)</code>	
Descripción:	Se registra un nuevo actor en el sistema.
Parámetros:	<code>dtp</code> : Contiene los datos del actor. <code>participaciones</code> : Colección de <code>DTParticipacion</code> que indica las películas en que la persona participó, junto con el rol y la duración de dicha participación.
Precondiciones:	Existe en el sistema una <code>Pelicula</code> con <code>id</code> igual a <code>idPelicula</code> para cada uno de los elementos de <code>participaciones</code> .
Postcondiciones:	Se crea una instancia de <code>Actor</code> . Para cada elemento de <code>participaciones</code> se crea una instancia de <code>Participante</code> con su correspondiente rol y minutos y se asocia con la instancia de <code>Actor</code> creada y la de <code>Película</code> correspondiente.

<code>obtenerDatosPelicula(id: string):DTPelicula</code>	
Descripción:	Se retorna la información relacionada a la película.
Parámetros:	<code>id</code> : El identificador en el sistema de la película.
Precondiciones:	Existe en el sistema una película con identificador <code>id</code> .
Postcondiciones:	Se retorna un <code>DTPelicula</code> con el título y duración de la <code>Película</code> , los géneros a la que pertenece, el nombre del director y los nombres de los actores protagonistas que participaron en la película.

- ii) Realizar el Diagrama de Clases de Diseño (DCD) resultante.

Notas:

- No incorporar setters ni getters al DCD.
- Incorporar en el DCD sólo los constructores o destructores que se utilicen en los diagramas de comunicación.
- No incorporar al DCD las operaciones de colecciones.

**Problema 3 (35 puntos)**

Se desea construir un sistema para la enseñanza de matemática en las escuelas del país. Existe un equipo de desarrollo trabajando sobre el módulo de operaciones geométricas, que presentó el Diagrama de Clases de Diseño de la figura en la siguiente página (los signos de + y – en las clases representan si el respectivo miembro es público o privado respectivamente). El sistema permite trabajar con algunas figuras básicas y otras denominadas “raras”, que se construyen a partir de otras figuras. El comportamiento de las operaciones de cada clase se describe a continuación:

`ControladorOperacionesGeometricas:`

- `calcularAreaFigura(i: int):` Devuelve el área de la figura cuyo atributo `id` vale `i`.

`Circulo:`

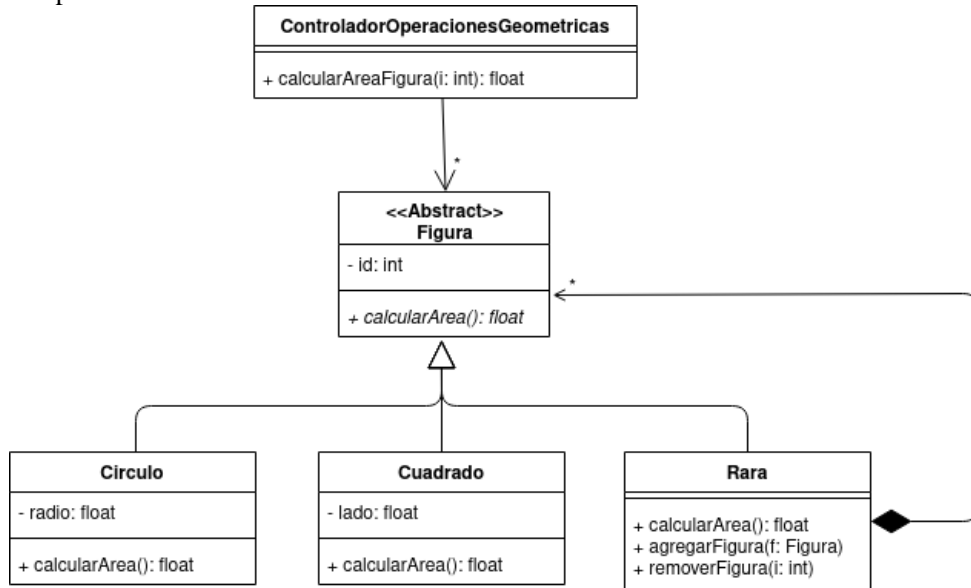
- `calcularArea():` Devuelve el área de la figura, calculada como  $\pi * \text{radio}^2$ .

`Cuadrado:`

- `calcularArea():` Devuelve el área de la figura, calculada como  $\text{lado}^2$ .

Rara:

- `calcularArea()`: Devuelve el área de la figura, calculada como la suma del área de las figuras que la constituyen.
- `agregarFigura(f: Figura)`: Agrega `f` a la figura implícita.
- `removeFigura(i: int)`: Remueve la figura cuyo atributo `id` vale `i` de la figura implícita.



**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, el nombre del patrón y qué rol cumple cada clase en el mismo.
- b) Implementar los archivos `.h` y `.cpp` de las clases `ControladorOperacionesGeometricas`, `Figura`, `Cuadrado` y `Rara`.
- c) En diálogo con los restantes equipos de desarrollo se determinó que era necesario que desde la clase `Diagramacion`, que pertenece a otro módulo, se pueda acceder a las operaciones provistas por `ControladorOperacionesGeometricas`. A la vez, se está pensando para futuras ediciones trabajar en la reimplementación del módulo completo, manteniendo sin cambiar la firma de las operaciones y sin generar un vínculo muy rígido entre ambos módulos. Teniendo en cuenta lo anterior, diseñe e implemente la solución propuesta, explicando el rol que cumplen las clases agregadas y cómo interactúan. Si utilizó algún patrón de diseño indique cuál es.

*Considerar:*

- Es posible utilizar la clase `map<K,V>` de STL.
- Puede suponer la existencia de la interface `ICollectionable` e implementaciones de `IDictionary` (clase `List`) e `Iterator` 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.