

# Prueba 1 - POO Invierno

4 de agosto, 2025

Nombre: \_\_\_\_\_

- Duración: 3 horas
- Está permitido el uso de calculadora, pero de ningún otro artefacto
- Puntaje total: 100
- La nota 4.0 se obtiene con 60 puntos

## 1. (40 puntos) Evento en auditorio

La EIC hará un evento de Programación Orientada a Objetos en el auditorio, acá llegarán exponentes de todo Chile a presentar sus conocimientos y avances sobre este paradigma de programación. Para controlar el público, la EIC creó una plataforma para vender entradas.

El auditorio de la EIC es bastante pequeño, tiene 2 secciones de 6 filas con 4 asientos cada una. Todos los asientos tienen el mismo valor: \$10.000 pesos.

Se le solicitó realizar una simulación del sistema y su lógica durante el proceso de venta de entradas.

Un usuario puede comprar de 1 a 5 entradas. Esto lo hace mediante peticiones, cada petición se identifica por un número de petición, y las N posiciones que desea comprar. Una posición tiene: Sección (1 o 2), Fila (1 a 6), Asiento (1 a 4)

Si en una solicitud hay alguna posición que esté ocupada, en primer lugar se rechaza, pero el sistema verifica si hay otras ubicaciones para ofrecerle, se las ofrece y el usuario decide si las toma o no, si no la toma, se rechaza y pasa a la solicitud del siguiente usuario.

Es importante señalar, que en este caso, solo se sugieren las que no pueden ser vendidas, las posiciones libres que solicitó el usuario se conservan. Por ejemplo: Un usuario pide 3 asientos en particular, uno de esos está libre, por lo tanto, el sistema ofrece otros 2 asientos libres, el usuario decide si los toma o no. Si los toma, se vende el asiento que estaba libre inicialmente y los 2 sugeridos por el sistema, si no lo toma, no se vende ninguno

Para simular esto no usaremos inputs. Como los usuarios son impredecibles, usaremos valores aleatorios para generar las peticiones y la decisión que tome en caso de ofrecerle una alternativa.

```
public static void main(String[] args) {  
    Random random = new Random();  
    int numero = 1 + random.nextInt(5); //Numero entre 1 y 5  
    int numero2 = random.nextInt(5); //Numero entre 0 y 4  
    boolean decision = random.nextBoolean(); /*Booleano aleatorio (True o False)*/  
}
```

Se pide:

- Hacer la simulación para N peticiones, donde N es un random entre 20 y 30 (incluye extremos)
- Una vez que termine, mostrar el estado del auditorio y el monto recaudado
- Debe usar POO

## 2. (60 puntos) The Social Network

Usted se encuentra en el año 2004 y ha tenido una brillante idea: crear una red social universitaria que conecte estudiantes entre sí, permitiendo que compartan publicaciones y se hagan amigos.

La aplicación debe cumplir con los siguientes requisitos:

- Cada **Usuario** tiene un nombre, correo electrónico y puede agregar amigos.
- Los usuarios pueden **crear publicaciones** (título y texto) que quedan asociadas a su perfil.
- Es posible listar las publicaciones de un usuario y sus amigos.
- Un usuario puede reaccionar ("Me gusta") a publicaciones de sus amigos. También puede comentarlas (texto).

Se solicita:

1. Dibujar el **diagrama de clases**
2. Implementar el **código en Java** de las clases diseñadas.
3. Funcionalidad agregarAmigo(), implica que 2 usuarios sean amigos mutuamente
4. Funcionalidad eliminarAmigo(), implica que 2 usuarios dejen de ser amigos mutuamente
5. Funcionalidad publicar(titulo, texto), le permite a un usuario hacer una publicación
6. Funcionalidad feed(), permite ver todo lo que han publicado los amigos de ese usuario. Esta incluye todos los detalles: Publicación, comentarios y cantidad de Me Gusta. Finalmente, se agregan las publicaciones del usuario
7. Funcionalidad meGusta(publicacion), funciona como un botón de 'Me Gusta', si no le ha dado me gusta a la publicación, le da me gusta. Si es que ya le dio me gusta, le quita el me gusta. En otras palabras, funciona como un interruptor.
8. Funcionalidad sugerirAmigo(usuario), dado un usuario, muestra otro usuario con el que tenga algún amigo en común. En el caso de no haber ningún candidato, retorna null. Cabe destacar, que este usuario sugerido no puede ser amigo del usuario que está buscando sugerencias.
9. En el **Main** simule la siguiente interacción:
  - Crear 7 usuarios.
  - Agregar amistades entre ellos, dibuje las conexiones en papel (requisito). Le servirá para comprobar la funcionalidad `sugerirAmigo()`
  - Eliminar 2 amigos
  - Hacer 5 publicaciones
  - Hacer 5 me gusta
  - Hacer 5 comentarios
  - Mostrar el feed de un usuario donde se vean algunas publicaciones, me gusta y comentarios
  - Demostrar que funciona `sugerirAmigo()`