PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

PROGRAMACIÓN 2

8va práctica (tipo b)
Segundo Semestre 2025

Indicaciones Generales:

- Duración: 110 minutos.

NO SE PERMITE EL USO DE APUNTES DE CLASE, FOTOCOPIAS NI MATERIAL IMPRESO

- No se pueden emplear **variables globales, NI OBJETOS** (con excepción de los elementos de **iostream, iomanip y fstream**). Tampoco se podrán emplear las funciones malloc, realloc, memset, strtok o strdup, igualmente no se puede emplear cualquier función contenida en las bibliotecas stdio.h, cstdio o similares y que puedan estar también definidas en otras bibliotecas. **NO PODRÁ EMPLEAR PLANTILLAS EN ESTE LABORATORIO**
- Deberá modular correctamente el proyecto en archivos independientes. LAS SOLUCIONES DEBERÁN DESARROLLARSE BAJO UN ESTRICTO DISEÑO DESCENDENTE. **Cada función NO debe sobrepasar las 20 líneas de código aproximadamente.** El archivo main.cpp solo podrá contener la función main de cada proyecto y el código contenido en él solo podrá estar conformado por tareas implementadas como funciones. En el archivo main.cpp deberá colocar un comentario en el que coloque claramente su nombre y código, **de no hacerlo se le descontará 0.5 puntos en la nota final.**
- El código comentado NO SE CALIFICARÁ. De igual manera NO SE CALIFICARÁ el código de una función si esta función no es llamada en ninguna parte del proyecto o su llamado está comentado.
- Los programas que presenten errores de sintaxis o de concepto se calificarán en base al 40% de puntaje de la pregunta. Los que no muestres resultados o que estos no sean coherentes en base al 60%.
- Se tomará en cuenta en la calificación el uso de comentarios relevantes.
- **TAMPOCO SE PODRÁ EMPLEAR LA CLÁUSULA protected NI LA CLÁUSULA friend, SALVO EN LAS CLASES AUTOREFERENCIADAS PARA DECLARAR AMIGA A LA CLASE QUE CONTENGA A LA LISTA, SE PROHÍBE QUE LOS NODOS DECLAREN friend A OTRAS CLASES QUE NO TENGAN QUE VER CON LA LISTA, DE HACERLO SE NO SE LE CALIFICARÁN LA PREGUNTA.**

SE LES RECUERDA QUE, DE ACUERDO AL REGLAMENTO DISCIPLINARIO DE NUESTRA INSTITUCIÓN, CONSTITUYE UNA FALTA GRAVE COPIAR DEL TRABAJO REALIZADO POR OTRA PERSONA O COMETER PLAGIO.

NO SE HARÁN EXCEPCIONES ANTE CUALQUIER TRASGRESIÓN DE LAS INDICACIONES DADAS EN LA PRUEBA

- **Puntaje total**: 20 puntos.

INDICACIONES INICIALES

Cree un proyecto de C++ en CLion siguiendo estrictamente las indicaciones que a continuación se detallan:

- La unidad de trabajo será **t:** (Si lo coloca en otra unidad, no se calificará su laboratorio y se le asignará como nota cero)
- Cree allí una carpeta con el nombre **"CO_PA_PN_Lab07_2025_2"** donde **CO** indica: Código del alumno, **PA** indica: Primer Apellido del alumno y **PN** primer nombre (de no colocar este requerimiento se le descontará 3 puntos de la nota final). **Allí colocará el proyecto solicitado en la prueba.**

Cuestionario:

La finalidad principal de este laboratorio es la de reforzar los conceptos contenidos en el capítulo 6 punteros a objetos.

Deberá elaborar un proyecto denominado "StreamersPolimorfismoYListas" y en él desarrollará el programa que dé solución al problema planteado. DE NO COLOCAR ESTE REQUERIMIENTO SE LE DESCONTARÁ 3 PUNTOS DE LA NOTA FINAL. **NO SE HARÁN EXCEPCIONES**

NO PODRÁ EMPLEAR ARREGLOS DE MÁS DE UNA DIMENSIÓN
NO PUEDE MANIPULAR UN PUNTERO CON MÁS DE UN ÍNDICE
LOS ARCHIVOS SOLO SE PUEDEN LEER UNA VEZ

Se te ha contratado para apoyar a un pequeño estudio en el análisis de **métricas de creadores de contenido (streamers)**. El objetivo es desarrollar un **sistema de gestión de streamers** aplicando los principios de **herencia, encapsulamiento, polimorfismo y clases autoreferenciadas** en programación orientada a objetos.

En este sistema, deberá manejar distintos tipos de **métricas**, cada una representando un aspecto diferente del desempeño de los streamers:

- **Métrica Básica**: mide la actividad general del canal, como las horas transmitidas y el promedio de espectadores.
- **Métrica Engagement**: evalúa la interacción del público, considerando la cantidad de mensajes en el chat, usuarios únicos activos y clips creados durante las transmisiones.
- **Métrica Calidad**: analiza el rendimiento técnico del stream, como el bitrate promedio, la proporción de cuadros perdidos y los cuadros por segundo (FPS) alcanzados.

Tu tarea consistirá en diseñar las clases necesarias, leer los datos desde un **archivo de texto** (o CSV) y generar un reporte que muestre las estadísticas correspondientes según el tipo de métrica. Este ejercicio busca evaluar tu dominio de **herencia, polimorfismo, encapsulamiento, sobrecarga, y manejo de listas ligadas** en C++, manteniendo una arquitectura limpia, reutilizable y con responsabilidad clara por tipo de objeto.

Se tiene un archivo del tipo CSV, el cual se describe a continuación:

metricas.csv

```
ENGAGEMENT,573984,161,Q&A largo,2025-08-07,2025-12-31,1586,571,27
BASIC,263905,44,Sesión Games,2025-11-26,2025-12-31,6.5,1116
QUALITY,314875,3033,Bitrate consistente,2025-07-27,2025-12-31,7586,0.063,48
…
```

tipo, id_stream, id_metrica, descripcion, fecha_calculo, fecha_expiracion.
Luego puede venir, dependiendo del tipo:
BASIC : horas_transmitidas, espectadores_promedio
ENGAGEMENT : mensajes_chat, usuarios_unicos_chat, clips_generados
QUALITY : bitrate_promedio_kbps, porcentaje_frames_perdidos, fps_promedio

PARTE 1 (6 puntos)

Construye una clase **Metrica**, que soporte los atributos generales de cualquier métrica asociada a un streamer. Para ello, añade los siguientes atributos privados:

- id: entero (int) que almacena el id de la métrica.
- descripcion: cadena (char* o string usted elija solo entre las dos) que almacena la descripción de la métrica.
- fecha_calculo : entero (int) que almacena la fecha en la que se calculó la métrica (formato AAAAMMDD).
- fecha_expiracion: entero (int) que almacena la fecha límite de validez de la métrica (formato AAAAMMDD).
- estado: booleano (bool) que indica si la métrica se encuentra **activa** (true) o **expirada** (false).

Deberás implementar **todos los constructores, destructores, getters y setters** necesarios para esta clase. Además, implementa los siguientes métodos virtuales:

- leer(ifstream&) : permite **leer la información de una métrica desde un archivo CSV** y almacenarla en la clase.
- imprimir(ofstream&) const: muestra los datos de la métrica en una sola línea bien formateada (ya sea en archivo o en consola), de modo que la información sea clara y fácil de interpretar.

Extensión de la clase Metrica

Crea tres clases derivadas de Metrica, cada una representando un tipo específico de métrica, con sus propios atributos de dominio:

1. **MetricaBasica:** double horas_transmitidas, int espectadores_promedio. Representa la actividad general del canal (tiempo de transmisión y promedio de audiencia).

2. **MetricaEngagement:** int mensajes_chat, int usuarios_unicos_chat, int clips_generados.Evalúa el nivel de interacción e interés de la comunidad durante las transmisiones.
3. **MetricaCalidad:** i nt bitrate_promedio_kbps, double porcentaje_frames_perdidos, int fps_promedio. Mide la calidad técnica del stream, su estabilidad y fluidez.

Cada clase hija deberá:

- Incluir **constructores, destructores y getters/setters propios**.
- Reutilizar los métodos leer e imprimir, extendiéndolos o sobrecargándolos para incluir los nuevos atributos. Los métodos solo leerán o imprimirán propios de la clase derivada, los atributos de la clase base se leerán o imprimirán obligatoriamente en la clase base.
- Mantener la coherencia con el atributo estado: al leer o actualizar una métrica, si la fecha actual es mayor que fecha_expiracion, el estado debe marcarse automáticamente como inactivo (false) caso contrario activo (true).

Podrá definir más métodos a las clases que usted considere conveniente pero no podrá agregar nuevos atributos ni modificar los ya existentes, esto incluye los nombres dados a los atributos y métodos dados.

PARTE 2 (10 puntos)

Construya una **lista ordenada doblemente ligada** que permita almacenar los diferentes datos del archivo "métricas.csv. la lista estará ordenada ascendentemente por el tipo métrica (Básica, Calidad y Engagement) y en caso de igualdad ascendentemente por la descripción de la métrica. Las operaciones deberán manejarse de manera polimórfica.

La lista estará conformada por las siguientes clases:

1. **Nodo:** class Metrica *métrica, class Nodo *anterior, class Nodo*siguiente;
2. **Lista:** class Nodo *listaDoblementeLigada;

Deberá respetar obligatoriamente los tipos de datos y nombres de los atributos.

La lista debe definir métodos que permita: crear la lista, insertar un nodo de manera ordenada de acuerdo a los criterios dados, determinar el reporte solicitado y eliminar, completa o parcialmente, la lista.

PARTE 3 (4 puntos)

Construye una clase AdministrarMetricas, Esta clase te permitirá cargar la información necesaria y gestionar los reportes necesarios del sistema.

La clase contendrá como único atributo: class Lista listaDeMetricas y los siguientes métodos:

1) **cargarArchivo,** que recibirá el nombre del archivo y cargará los datos del archivo en la lista ordenada.
2) **generarRepote**, recibirá el nombre del archivo de reporte y mostrará un reporte similar al que se muestra a continuación.
3) **eliminarNodos**, eliminar de la lista, las métricas cuyo estado sea EXPIRADA.
4) **generarRepote**, vuelva a generar el reporte y mostrará un reporte en otro archivo cuyo nombre sea recibido también.
5) **eliminarLista**, eliminar completamente la lista.

```
================================================================================
                              REPORTE DE METRICAS
================================================================================
[METIRCAS BASIC]
    Código   Descripción      Fecha Calc.   Expira      Estado     Horas  Viewers
    MB-0003  Sesión Ant.      20250620      20251231    ACTIVA     4.20   515m
    MB-0004  Sesión Juegos    20250625      20251231    ACTIVA     3.50   450m
    MB-0005  Sesión Música    20250415      20251001    ACTIVA     5.00   410m
--------------------------------------------------------------------------------
[METIRCAS ENGAGE]
    Código   Descripción      Fecha Calc.   Expira      Estado     Msgs   Chatters  Clips
    ME-0102  Chat activo      20250620      20251231    ACTIVA     1800   360       31
    ME-0103  Reacción evento  20250701      20251231    ACTIVA     1800   320       24
--------------------------------------------------------------------------------
[METIRCAS QUALITY]
    Código   Descripción      Fecha Calc.   Expira      Estado     Bitrate  Drop%  FPS
    MQ-3001  Calidad estable  20250620      20251231    ACTIVA     6200     0.02   60
    MQ-3002  Picos leves      20250701      20251231    ACTIVA     6200     0.06   59
--------------------------------------------------------------------------------
[TOTAL] METRICAS ACTIVAS:  6 | MÉTRICAS EXPIRADAS:  1 | FECHA DE REPORTE: 2025-11-14
================================================================================
```

Al finalizar la práctica, comprima la carpeta dada en las indicaciones iniciales empleando el programa Zip que viene por defecto en el Windows, no se aceptarán los trabajos compactados con otros programas como RAR, WinRAR, 7zip o similares.

Profesores del curso: Miguel Guanira, Andrés Melgar, Rony Cueva, Eric Huiza, Erasmo Gómez

San Miguel, 14 de noviembre del 2025.