PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

PROGRAMACIÓN 2

5ta 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**). **NO PUEDE UTILIZAR LA CLASE string**. 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.

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_Lab05_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 3 del curso: "Programación Orientada a Objetos". En este laboratorio se trabajará con arreglos dinámicos de objetos..

Deberá elaborar un proyecto denominado "StreamersPOO" 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**

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

streamers.csv

```
xQcOW,6196161750,27716.75,3246298,Minecraft
summit1g,6091677300,25610.33,5310163,JustChatting
Gaules,5644590915,10976.87,1767635,Valorant
…
```

cuenta, tiempo total, promedio espectadores, seguidores y categoría

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 contrata para ayudar a un pequeño estudio a analizar métricas de creadores de contenido (streamers). Debes construir una **aplicación de consola en C++** que **cargue** información desde un **archivo CSV**, la almacene en un **arreglo dinámico de objetos**, y permita **ordenar** y **filtrar** la información en una serie de reportes solicitados para gerencia.

PARTE 1 (4 puntos)

Construye una clase **Streamer,** que soporte los atributos de cada streamer.

Para ello añade los siguientes atributos:

1) **cuenta**, es un char* (cadena en memoria dinámica) que almacena el nombre/cuenta del streamer y se usa para mostrar y ordenar por nombre.
2) **tiempo_total**, es un long long que representa el tiempo total transmitido (en horas) por el streamer dentro del sistema.
3) **promedio_espectadores**, es un valor de punto flotante, que guarda el promedio de espectadores simultáneos del streamer en el periodo considerado.
4) **n_seguidores**, es un entero que contiene la cantidad total de seguidores del streamer registrada en la data.
5) **categoria**, es un char* (cadena en memoria dinámica) que indica la categoría principal del streamer (p. ej., "Minecraft", "Valorant", "JustChatting").

No olvides crear **todos** los constructores, getters y setters para esta clase.

Además, crea los siguientes métodos:

1) **leer_streamer**, un método que permite leer la información de un streamer de un archivo csv y alojarla en la clase.
2) **mostrar_streamer**, un método que permita mostrar la información de un streamer, ya sea en un archivo o en consola, en una línea bien definida y con los espacios requeridos para entender correctamente la información.
3) **copiar**(Streamer s), un método que permite copiar la información de un streamer brindado como parámetro de este método, hacia la clase Streamer.

PARTE 2 (8 puntos)

Construye una clase ***GestorStreamers***, Esta clase te permitirá cargar la información necesaria y gestionar el menú mostrado al usuario. Debe implementar el constructor por defecto y los getters y setters correspondientes.

Para ello añade los siguientes atributos:

1) data, es un arreglo dinámico de **Streamer,** con la totalidad de la información del sistema.
2) dataVista, es un arreglo dinámico de **Streamer,** es un subconjunto de la **data** principal, aplicando los filtros para un reporte determinado.
3) cantidad_datos, un entero que lleve la cantidad total de datos.
4) cantidad_datos_vista, un entero que lleve la cantidad de datos del arreglo dataVista.

Además crea los siguientes métodos:

1) **cargar_datos**, un método que recibe la ruta del archivos de streamers, y carga en data la totalidad de los streamers del sistema.
2) **mostrar_menu**, un método que por consola, muestra la información del menú de opciones que el usuario tiene en el sistema. Este Menú sólo debe parar cuando el usuario elija la opción "Terminar". Las posibles opciones para el menú son las siguientes:
    a) Cargar Datos
    b) Mostrar Reporte
        i) Reporte Top10 streamers por número de seguidores.
        ii) Reporte Bottom10 streamers por tiempo total transmitido.
        iii) Reporte Top5 categorías con mayor likes.
        iv) Reporte de Categoría
        v) Reporte de Influencia
    c) Generar Reporte
        i) Reporte Top10 streamers por número de seguidores.
        ii) Reporte Bottom10 streamers por tiempo total transmitido.
        iii) Reporte Top5 categorías con mayor promedio de espectadores..
        iv) Reporte de Categoría
        v) Reporte de Influencia
    d) Generar Todos los reportes
    e) Terminar
3) **copiar_datos**(), un método que permite copiar la información del arreglo data al arreglo dataVista.
4) **cortar_datos**(int), un método que permita cortar el arreglo dataVista, dada la cantidad de elementos definida como parámetro.

PARTE 3 (8 puntos)

En la clase GestorStreamers, debe realizar lo siguiente:

- Debe generar un método por cada reporte mostrado en el menú y este debe soportar ser impreso en consola (**Mostrar**) o en un archivo (**Generar**), cada método debe usar el arreglo **data** y filtrar la información requerida y almacenarla en el arreglo **dataVista.**
  Solo se evaluarán estos métodos si han sido filtrados correctamente y con la memoria correctamente reservada en dataVista, de lo contrario el método recibirá nota 0.
- El único método de ordenamiento permitido es Quicksort (O(nlogn)) y solo puede usar el provisto en la libreria <cstdlib>:

```
qsort (myArray, size, sizeof(myArray[0]), compare);
```

Los reportes se describen a continuación:

1. **Reporte Top10 streamers por número de seguidores**: El arreglo debe estar ordenado por número de seguidores de manera descendentemente y sólo debe contener los 10 primeros streamers.

2. **Reporte Bottom10 streamers por tiempo total transmitido**: El arreglo debe estar ordenado por el tiempo total transmitido ascendentemente y sólo debe contener los 10 primeros streamers.
3. **Reporte Top5 categorías con mayor promedio de espectadores**: El arreglo debe estar ordenado por promedio de espectadores descendentemente y por categorías alfabéticamente y sólo debe contener los 5 primeros streamers.
4. **Reporte de Categoría**: El arreglo debe estar ordenado por categorías alfabéticamente.
5. **Reporte de Influencia**: El arreglo debe estar ordenado por Influencia. Para calcular la influencia puede seguir la siguiente fórmula:

$$\left(\frac{promedio\_espectadores * tiempo\_total}{log(seguidores+1)}\right).$$

La salida de cada reporte debe ser organizada y correcta para que pueda ser visualizada por un gerente. Queda en ud, proponer la MEJOR OPCIÓN para mostrar y generar sus reportes.

La función main de su programa debe permitir mostrar el menú de opciones de su sistema.

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, Rony Cueva, Erasmo Gómez, Andrés Melgar, Eric Huiza

San Miguel, 10 de octubre del 2025.