Práctica 8. Uso de ficheros

1. Introducción

En esta práctica continuaremos con el desarrollo del sistema en el que venimos trabajando durante el curso. El objetivo en este caso es ofrecer la funcionalidad para que toda la información de los usuarios que hasta ahora teníamos que cargar una y otra vez en el arranque de la aplicación, se guarde permanentemente. Con esto, daremos un gran paso para que la aplicación tenga una funcionalidad muy cercana a un uso real.

Para conseguir ese objetivo, necesitamos hacer uso de ficheros (que se guardan en disco duro) que nos sirvan de soporte para almacenar toda la información. Recordemos que, sin estos, los datos cargados hasta ahora en nuestra aplicación están almacenados en memoria RAM, que es volátil; esto es, se pierden una vez cerremos la aplicación (o se interrumpa la alimentación eléctrica).

Para poder manejar estos ficheros, nos apoyaremos en las bibliotecas de gestión de gestión de entrada/salida (*input/output*) que nos ofrece C++, y que hemos visto en clase. Y, dado que vamos a leer/escribir objetos, nos apoyaremos en la modalidad de ficheros binarios (`ios::binary`).

Ejercicio

Deberás almacenar en fichero, al menos, la información de todos los usuarios. Para ello, crea un fichero `users.dat`, donde se guardará la información de estos. También puedes optar por tener distintos ficheros para los distintos tipos de usuarios que tengas contemplados en

tu sistema.

Además de los ficheros, deberás —evidentemente— implementar la funcionalidad necesaria para que toda la información que tenemos cargada en la colección (la que sea que usemos) de usuarios pueda guardarse y/o cargarse en/desde el disco duro. Recuerda, muy importante, como ya vimos en clase, la dinámica a seguir es la misma que, por ejemplo, la partida de un videojuego; esto es, primero se carga la partida y al final, antes de salir, se guarda la partida. Lo que, en nuestro contexto, se traduciría en:

1. Al inicio, **cargamos** datos del fichero a la colección de usuarios. Esto es: fichero/s $\Longrightarrow$ coleccion.

2. Ejecución de la aplicación normal.

3. Antes de salir, **guardamos** los datos de la colección al fichero, en modalidad `ios::trunc`, para conseguir tener siempre una *copia espejo* del contenido de lal colección en el fichero. Esto es: colección $\Longrightarrow$ fichero/s.