



Sistema Estadístico Aplicado a Equipos de Fútbol

Algoritmos y Estructuras de Datos II

Romero Agustín Ezequiel

Silguero Luis Ernesto

Informe del Proyecto

1. Tema y Contexto

Tema: Sistema de gestión y análisis estadístico para equipos de fútbol.

Contexto: Este proyecto se desarrolla como parte de la materia Algoritmos y Estructuras de Datos II. El sistema permite cargar equipos de fútbol con sus jugadores, modificar datos, almacenar la información de forma persistente y generar reportes estadísticos mediante corte de control sobre archivos.

2. Objetivo

Desarrollar un sistema que permita:

- Gestionar equipos de fútbol (conjuntos registros de jugadores) mediante una cola que almacena los mismos de manera temporal antes del almacenamiento persistente.
- Almacenar la información en un archivo binario.
- Generar reportes estadísticos por equipo (valor total, edad promedio, goleador, etc.) y generales (equipo de mayor valor, máximos goleadores y asistidores) mediante corte de control.

3. Decisiones de Diseño

3.1. Estructuras de Datos

3.1.1. Cola (FIFO) para jugadores

- **Justificación:** Se utilizó una cola para almacenar temporalmente los jugadores de un equipo durante la carga. La cola es adecuada porque los jugadores se cargan en un orden específico y se procesan (guardan en archivo) en ese mismo orden. Además, la cola permite una gestión sencilla de los jugadores en el orden de llegada.
- **Alternativas consideradas:** Se consideró usar una lista enlazada simple, pero se optó por una cola porque no se necesitaba acceso aleatorio, solo procesamiento secuencial.

3.1.2. Archivos binarios para persistencia

- **Justificación:** Se eligió un archivo binario para almacenar los equipos y jugadores porque permite guardar y recuperar la información de forma eficiente y directa, manteniendo la estructura de los datos.
- **Alternativas consideradas:** Archivos de texto. Se descartaron porque aunque son legibles, son menos eficientes en espacio y tiempo para datos estructurados.

3.2. Técnicas de Programación

3.2.1. Corte de Control

- **Justificación:** Se aplicó corte de control sobre el archivo binario para generar reportes por equipo. Esta técnica es adecuada porque los registros en el archivo están agrupados por equipo, lo que permite procesar cada equipo en un solo barrido del archivo.

3.3. TADs (Tipos Abstractos de Datos)

- **colaJugadores.h:** Define la estructura de la cola (tColaJugadores) y los nodos (tNodoJugadores) que contienen los jugadores (tJugador). Implementa las operaciones básicas de una cola: iniciarCola, encolar, desencolar, y además incluye la función cambioJugador que permite modificar los datos de un jugador en la cola.
- **cargajugadores.h:** Se encarga de la persistencia de datos. La función iniciar abre el archivo "Equipos.dat" en modo "a+b" (lectura y escritura binaria). La función cargarEquipos recorre la cola y escribe cada jugador en el archivo. La función finalizar cierra el archivo.
- **corteControlEquipos.h:** Implementa el algoritmo de corte de control. La función inicio prepara el archivo y las variables globales para el corte. La función procesoCorte recorre el archivo y para cada equipo (agrupación por club) calcula las estadísticas. Las funciones principioCorte, unJugador y finCorte se utilizan para el procesamiento de cada equipo.

4. Desarrollo Obtenido

4.1. Funcionalidades Implementadas

- **Carga de equipos y jugadores:** Permite cargar jugadores en una cola por equipo.
- **Modificación de jugadores:** Permite modificar los datos de un jugador en la cola antes de guardar.
- **Almacenamiento en archivo binario:** Guarda los jugadores de la cola en un archivo binario.
- **Visualización de equipos existentes:** Muestra todos los equipos y jugadores almacenados en el archivo.
- **Reporte por corte de control:** Genera estadísticas por equipo y generales.

4.2. Estructura del Código

- **colaJugadores.h:** Define la cola y sus operaciones.
- **cargajugadores.h:** Maneja la apertura, cierre y escritura en el archivo binario.
- **corteControlEquipos.h:** Implementa el corte de control para generar reportes.
- **sistemaFutbol.c:** Contiene la función main y la lógica de la interfaz de usuario.

4.3. Compilación y Ejecución

- **Compilación:** gcc -o sistemaFutbol sistemaFutbol.c
- **Ejecución:** ./sistemaFutbol

5. Reflexión Crítica

5.1. Proceso de Desarrollo

- Se rotaron los roles para que todos los integrantes participaran en diferentes partes.
- Se fijaron objetivos a cumplir en lapsos de tiempo.
- Uso de Inteligencia Artificial (IA):
 - Revisar y sugerir mejoras en la estructura del código.
 - Ayudar en la depuración de errores y a un mejor entendimiento del comportamiento del cursor en archivos binarios.
 - Generar ideas para la organización del informe y la justificación de decisiones de diseño.
 - Todo el código obtenido con IA fue **comprendido, adaptado y probado** por el equipo para garantizar su correcto funcionamiento e integración en el proyecto.

5.2. Aprendizajes

- Se reforzó el uso de colas y la manipulación de archivos binarios.
- Se perfeccionó la aplicación corte de control en archivos.
- Se aprendió a tomar decisiones de diseño considerando alternativas.
- Adaptación al trabajo en equipo de forma remota.

5.3. Dificultades y Soluciones

- **Dificultad:** Abundancia de posibilidades de datos por jugador.
- **Solución:** Priorización de estadísticas más relevantes o de mayor interés para evitar complejizar el sistema con variables irrelevantes o de poco interés.
- **Dificultad:** Interfaz de usuario lineal y poco vistosa.
- **Solución:** Implementar un concepto de diseño de impresiones por pantalla minimalista, pero priorizando la claridad para que sea lo mas amigable posible con el usuario.

6.1 Conclusiones

- El sistema cumple con los objetivos planteados: gestión de equipos, almacenamiento y reportes.
- Las estructuras de datos y técnicas utilizadas fueron adecuadas para el problema.