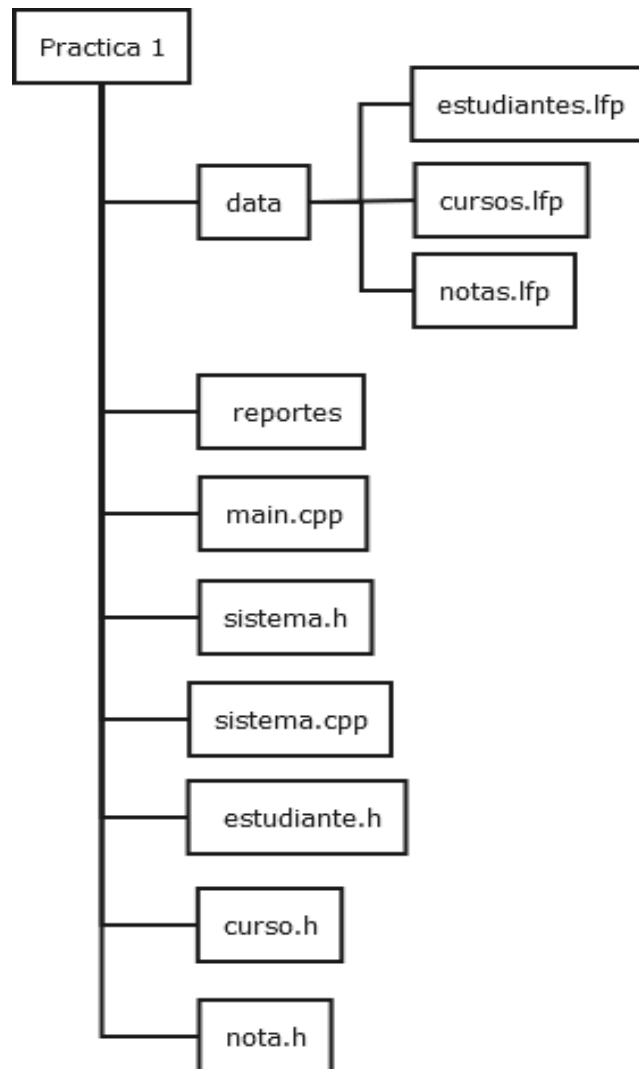


Manual Técnico

Estructura General del Programa:

El sistema fue hecho en visual estudio code con el lenguaje de **C++** utilizando programación orientada a objetos(POO).

El proyecto está desarrollado de una forma modular, separando declaraciones de clases (.h) e implementaciones (.Cpp).



Clases Utilizadas:

El sistema tiene cuatro clases principales:

1. Clase Sistema

- Clase central que administra: Carga de archivos
 - Procesamiento de datos
 - Generación de reportes
 - Cálculos estadísticos
- Atributos principales:
 - `vector<Estudiante> estudiantes`
 - `vector<Curso> cursos`
 - `vector<Nota> notas`
 - `bool estudiantesCargados`
 - `bool cursosCargados`
 - `bool notasCargadas`
 - `string rutaBase`
 - `string rutaReportes`
- Métodos principales:
 - `cargarEstudiantes()`
 - `cargarCursos()`
 - `cargarNotas()`
 - `reporteEstadisticasCursoHTML()`
 - `reporteRendimientoEstudianteHTML()`
 - `reporteTop10HTML()`
 - `reporteIndiceReprobacionHTML()`
 - `reporteAnalisisCarreraHTML()`

- Métodos privados de apoyo:
 - calcularPromedio()
 - calcularMediana()
 - calcularDesviacionEstandar()
 - calcularPercentil()

2. Clase Estudiante

- Atributos privados:
 - carnet
 - nombre
 - apellido
 - carrera
 - semestre

3. Clase Curso

- Atributos privados:
 - codigo
 - nombre
 - creditos
 - semestre
 - carrera

4. Clase Nota

- Atributos privados:
 - carnet
 - codigoCurso
 - nota
 - ciclo
 - anio

Funcionamiento General:

- Cargar archivos desde la carpeta data/
- Leer línea por línea
- Separar campos usando stringstream
- Convertir datos con stoi() y stod()
- Crear objetos y almacenarlos en vectores
- Procesar datos mediante recorridos y cálculos estadísticos
- Generar reportes en formato HTML dentro de la carpeta reportes/

Requerimientos Técnicos:

- Lenguaje: C++
- Compilador compatible con C++11 o superior
- Uso de librerías:
 - <vector>
 - <string>
 - <fstream>
 - <sstream>
 - <algorithm>
 - <map>
 - <cmath>
- Estructura de carpetas obligatoria (data/ y reportes/)

Observaciones del Desarrollo:

Se implementó encapsulamiento completo.

Se evitaron duplicados mediante control de banderas booleanas.

Se validó apertura de archivos antes de procesarlos.

Se manejaron errores de conversión con bloques try-catch.

Se modularizó el código para facilitar mantenimiento.

Diagrama de Clases:

