# Ejercicio 1. Gestión de Estudiantes y sus Calificaciones

**Narrativa:**

Una escuela desea gestionar la información de sus estudiantes y sus calificaciones en diferentes materias. Cada estudiante tiene un nombre y una lista de calificaciones. El sistema debe permitir agregar estudiantes, asignar calificaciones y mostrar un reporte de calificaciones por estudiante.

**Requerimientos:**

- Crear una clase Estudiante con atributos para el nombre y una lista de calificaciones.
- Crear una clase Escuela con una lista de estudiantes.
- Implementar métodos para agregar, actualizar y eliminar estudiantes, asignar calificaciones y mostrar un reporte de calificaciones.
- Mostrar el promedio de calificaciones de cada estudiante.
- Mostrar el promedio de calificaciones de todos los estudiantes.

# Ejercicio 2. Explique el siguiente código en términos generales y en términos particulares.

Hágase preguntas como las siguientes:

- ¿Qué hace cada clase?
- ¿qué hace cada método?
- ¿Para qué funciona el programa en términos generales?
- ¿Qué hacen los atributos?

```
using System;
using System.Collections.Generic;
using System.Linq;

public class Materia
{
    private string nombre;
    private List<int> calificaciones;

    public Materia(string nombre)
    {
```

```
        this.nombre = nombre;
        this.calificaciones = new List<int>();
    }

    public string GetNombre()
    {
        return nombre;
    }

    public void SetNombre(string nombre)
    {
        this.nombre = nombre;
    }

    public List<int> GetCalificaciones()
    {
        return calificaciones;
    }

    public void AgregarCalificacion(int calificacion)
    {
        calificaciones.Add(calificacion);
    }

    public double CalcularPromedio()
    {
        if (calificaciones.Count == 0)
            return 0;
        return calificaciones.Average();
    }

    public override string ToString()
    {
        return $"{nombre} - Promedio: {CalcularPromedio():F2}";
    }
}

public class Estudiante
{
    private string nombre;
    private Dictionary<string, Materia> materias;

    public Estudiante(string nombre)
    {
        this.nombre = nombre;
        this.materias = new Dictionary<string, Materia>();
    }
```

```
    public string GetNombre()
    {
        return nombre;
    }

    public void SetNombre(string nombre)
    {
        this.nombre = nombre;
    }

    public Dictionary<string, Materia> GetMaterias()
    {
        return materias;
    }

    public void AgregarMateria(string nombreMateria)
    {
        if (!materias.ContainsKey(nombreMateria))
        {
            materias[nombreMateria] = new Materia(nombreMateria);
            Console.WriteLine($"Materia {nombreMateria} agregada a
{nombre}");
        }
        else
        {
            Console.WriteLine($"La materia {nombreMateria} ya existe
para {nombre}");
        }
    }

    public void EliminarMateria(string nombreMateria)
    {
        if (materias.ContainsKey(nombreMateria))
        {
            materias.Remove(nombreMateria);
            Console.WriteLine($"Materia {nombreMateria} eliminada de
{nombre}");
        }
        else
        {
            Console.WriteLine($"La materia {nombreMateria} no existe
para {nombre}");
        }
    }
```

```
    public void AsignarCalificacion(string nombreMateria, int
calificacion)
    {
        if (materias.ContainsKey(nombreMateria))
        {
            materias[nombreMateria].AgregarCalificacion(calificacion);
            Console.WriteLine($"Calificación {calificacion} asignada a
{nombre} en {nombreMateria}");
        }
        else
        {
            Console.WriteLine($"La materia {nombreMateria} no existe
para {nombre}");
        }
    }

    public double CalcularPromedioGeneral()
    {
        if (materias.Count == 0)
            return 0;
        return materias.Values.Average(m => m.CalcularPromedio());
    }

    public override string ToString()
    {
        return $"{nombre} - Promedio General:
{CalcularPromedioGeneral():F2}";
    }
}

public class Escuela
{
    private List<Estudiante> estudiantes = new List<Estudiante>();

    public void AgregarEstudiante(Estudiante estudiante)
    {
        estudiantes.Add(estudiante);
        Console.WriteLine($"Estudiante agregado:
{estudiante.GetNombre()}");
    }

    public void AsignarCalificacion(string nombreEstudiante, string
nombreMateria, int calificacion)
    {
        Estudiante estudiante = estudiantes.Find(e => e.GetNombre() ==
nombreEstudiante);
        if (estudiante != null)
```

```
        {
            estudiante.AsignarCalificacion(nombreMateria,
calificacion);
        }
        else
        {
            Console.WriteLine($"El estudiante '{nombreEstudiante}' no
existe.");
        }
    }

    public void MostrarReporte()
    {
        Console.WriteLine("Reporte de calificaciones:");
        foreach (Estudiante estudiante in estudiantes)
        {
            Console.WriteLine(estudiante);
            foreach (Materia materia in
estudiante.GetMaterias().Values)
            {
                Console.WriteLine(materia);
                Console.WriteLine("Calificaciones: " + string.Join(",
", materia.GetCalificaciones()));
            }
            Console.WriteLine();
        }
    }
}

public class Program
{
    public static void Main()
    {
        Escuela escuela = new Escuela();

        Estudiante maria = new Estudiante("María");
        maria.AgregarMateria("Matemáticas");
        maria.AgregarMateria("Historia");

        Estudiante pedro = new Estudiante("Pedro");
        pedro.AgregarMateria("Ciencias");
        pedro.AgregarMateria("Literatura");

        escuela.AgregarEstudiante(maria);
        escuela.AgregarEstudiante(pedro);

        escuela.AsignarCalificacion("María", "Matemáticas", 90);
```

```
        escuela.AsignarCalificacion("María", "Matemáticas", 85);
        escuela.AsignarCalificacion("María", "Historia", 88);

        escuela.AsignarCalificacion("Pedro", "Ciencias", 78);
        escuela.AsignarCalificacion("Pedro", "Literatura", 92);

        escuela.MostrarReporte();
    }
}
```

## Ejercicio 3. Investiga el término encapsulamiento en POO.