TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE TIJUANA

SUBDIRECCIÓN ACADÉMICA
DEPARTAMENTO DE SISTEMAS Y COMPUTACIÓN

SEMESTRE:
Agosto - Diciembre 2025

CARRERA:
Ingeniería Informática

MATERIA:
Patrones de diseño de software

TÍTULO ACTIVIDAD:
Examen unidad 2

UNIDAD A EVALUAR:
Unidad 2

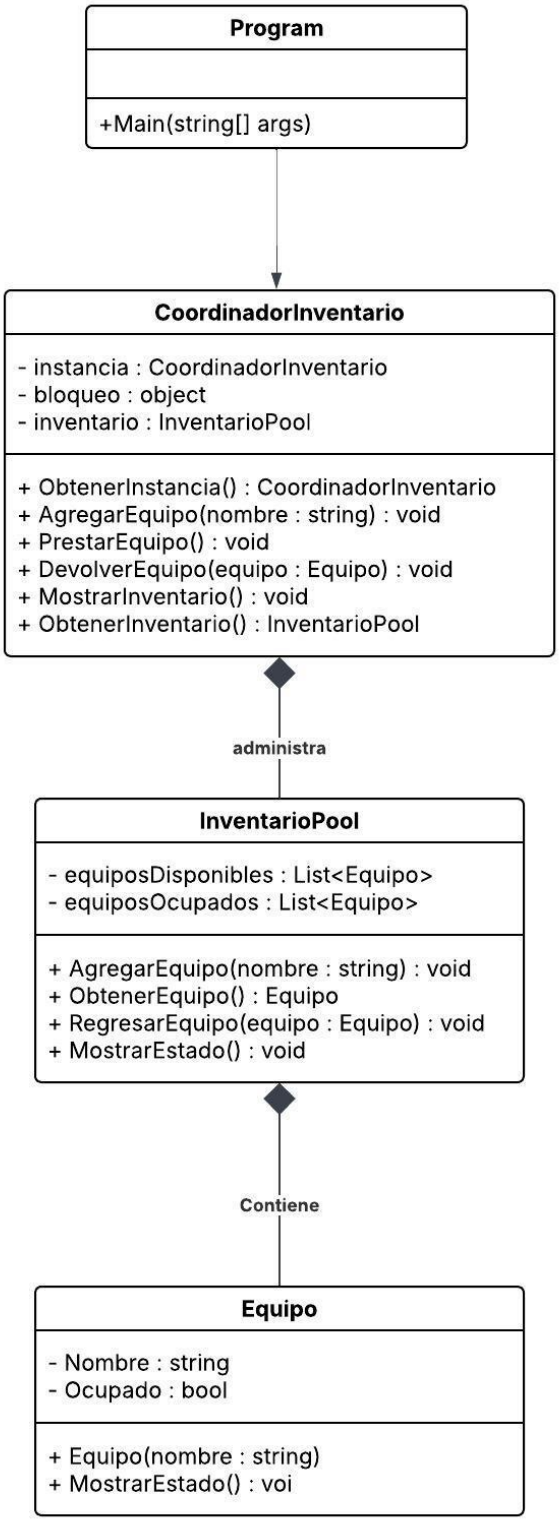
NOMBRE Y NÚMERO DE CONTROL DEL ALUMNO:
Molina Hernández Kevin Alexander (21212351)

NOMBRE DEL MAESTRO (A):
MARIBEL GUERRERO LUIS

Documentación

Gestión de inventario y recursos de una escuela

- **Diagrama UML**



- **Program**

```
namespace ExamenUnidad2
{
    0 referencias
    class Program
    {
        0 referencias
        static void Main(string[] args)
        {
            // Singleton: solo una instancia
            CoordinadorInventario coordinador1 = CoordinadorInventario.ObtenerInstancia();
            CoordinadorInventario coordinador2 = CoordinadorInventario.ObtenerInstancia();

            Console.WriteLine("\nComprobación del Singleton:");
            Console.WriteLine(ReferenceEquals(coordinador1, coordinador2));

            //El coordinador agrega materiales al inventario
            Console.WriteLine();
            coordinador1.AgregarEquipo("Protoboard");
            coordinador1.AgregarEquipo("Multímetro");
            coordinador1.AgregarEquipo("Fuente de poder");

            //Mostrar inventario
            coordinador1.MostrarInventario();

            //Préstamo
            Console.WriteLine("\nUn alumno solicita un equipo:");
            var inventario = coordinador1.ObtenerInventario();
            Equipo equipoPrestado = inventario.ObtenerEquipo();
            Console.WriteLine($"Equipo prestado: {equipoPrestado.Nombre}");

            coordinador1.MostrarInventario();

            //Devolución
            Console.WriteLine("\nEl alumno devuelve el equipo:");
            coordinador1.DevolverEquipo(equipoPrestado);
            coordinador1.MostrarInventario();
            Console.ReadKey();
        }
    }
}
```

- **CoordinadorInventario**

```
namespace ExamenUnidad2
{
    8 referencias
    class CoordinadorInventario
    {
        private static CoordinadorInventario instancia;
        private static readonly object bloqueo = new object();

        // Aquí se conecta el Singleton con el Object Pool
        private InventarioPool inventario;

        1 referencia
        private CoordinadorInventario()
        {
            inventario = new InventarioPool();
            Console.WriteLine("Coordinador del inventario creado.");
        }

        2 referencias
        public static CoordinadorInventario ObtenerInstancia()
        {
            lock (bloqueo)
            {
                if (instancia == null)
                    instancia = new CoordinadorInventario();
                else
                    Console.WriteLine("Ya existe un coordinador, no se puede crear otro.");
            }
            return instancia;
        }

        //Método que conectan al coordinador con el pool
        3 referencias
        public void AgregarEquipo(string nombre)
        {
            inventario.AgregarEquipo(nombre);
        }

        0 referencias
        public void PrestarEquipo()
        {
            Equipo equipo = inventario.ObtenerEquipo();
            if (equipo != null)
                Console.WriteLine($"El coordinador prestó: {equipo.Nombre}");
        }
```

```
        1 referencia
        public void DevolverEquipo(Equipo equipo)
        {
            inventario.RegresarEquipo(equipo);
            Console.WriteLine($"El coordinador recibió de vuelta: {equipo.Nombre}");
        }

        3 referencias
        public void MostrarInventario()
        {
            inventario.MostrarEstado();
        }

        //Permite acceder al inventario
        1 referencia
        public InventarioPool ObtenerInventario()
        {
            return inventario;
        }
    }
}
```

- **InventarioPool**

```
namespace ExamenUnidad2
{
    3 referencias
    class InventarioPool
    {
        private List<Equipo> equiposDisponibles = new List<Equipo>();
        private List<Equipo> equiposOcupados = new List<Equipo>();

        1 referencia
        public void AgregarEquipo(string nombre)
        {
            equiposDisponibles.Add(new Equipo(nombre));
            Console.WriteLine($"Equipo agregado: {nombre}");
        }

        //Asigna un equipo
        2 referencias
        public Equipo ObtenerEquipo()
        {
            if (equiposDisponibles.Count > 0)
            {
                Equipo equipo = equiposDisponibles[0];
                equipo.Ocupado = true;
                equiposDisponibles.Remove(equipo);
                equiposOcupados.Add(equipo);
                return equipo;
            }
            else
            {
                Console.WriteLine("No hay equipos disponibles.");
                return null;
            }
        }

        //Recibe el equipo devuelto
        1 referencia
        public void RegresarEquipo(Equipo equipo)
        {
            if (equipo != null)
            {
                equipo.Ocupado = false;
                equiposOcupados.Remove(equipo);
                equiposDisponibles.Add(equipo);
            }
        }
```

```
        //Muestra el estado general del inventario
        1 referencia
        public void MostrarEstado()
        {
            Console.WriteLine("\n--- Equipos disponibles ---");
            foreach (var e in equiposDisponibles)
                e.MostrarEstado();

            Console.WriteLine("\n--- Equipos ocupados ---");
            foreach (var e in equiposOcupados)
                e.MostrarEstado();
        }
    }
}
```

- **Equipo**

```
namespace ExamenUnidad2
{
    12 referencias
    class Equipo
    {
        5 referencias
        public string Nombre { get; set; }
        5 referencias
        public bool Ocupado { get; set; }

        1 referencia
        public Equipo(string nombre)
        {
            Nombre = nombre;
            Ocupado = false;
        }

        2 referencias
        public void MostrarEstado()
        {

            if (Ocupado)
            Console.ForegroundColor = ConsoleColor.Red;    // Rojo es de color ocupado
            else
            Console.ForegroundColor = ConsoleColor.Green; // Verde es de color libre

            Console.WriteLine($"{Nombre} - Estado: {(Ocupado ? "Ocupado" : "Libre")}");
            Console.ResetColor();
        }
    }
}
```

- **Conclusión**

El programa realizado se incorpora los patrones de diseño Singleton y Object Pool aplicados a la gestión del inventario escolar.

Por un lado, el patrón Singleton garantiza que exista un único coordinador encargado de administrar el inventario, evitando duplicaciones al controlar los materiales. Por otro lado, el patrón Object Pool permite reutilizar los equipos como protoboards o multímetros, asignándolos temporalmente a los alumnos y devolviéndolos al inventario una vez que se liberan.

La comunicación entre ambos patrones asegura una administración centralizada y eficiente el coordinador que es el Singleton es el único que puede interactuar con el pool, agregando, prestando o devolviendo equipos según su disponibilidad.

De esta forma, se optimizan los recursos escolares, se reduce la creación innecesaria de objetos y se mantiene un control claro sobre el estado de cada material.