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

TÍTULO ACTIVIDAD:
Examen unidad 2

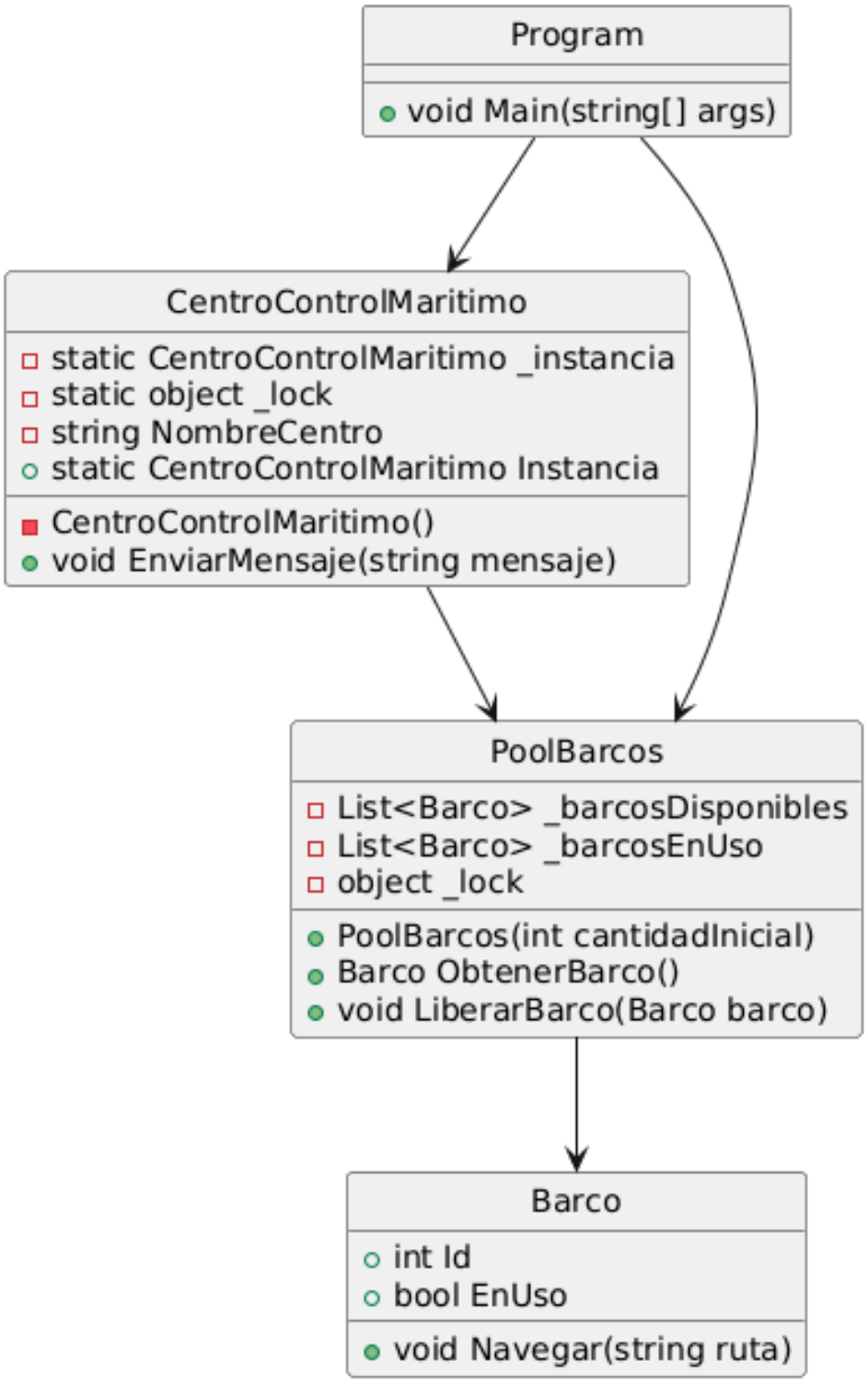
UNIDAD A EVALUAR:
2

NOMBRE Y NÚMERO DE CONTROL DEL ALUMNO:

Soriano Cabrera Edgar Manuel 21212860

NOMBRE DEL MAESTRO (A):
Maribel Guerrero Luis

Diagrama UML - Control de flotas y rutas marítimas



Class Barco

```
namespace ExamenU2
{
    12 referencias
    public class Barco
    {
        4 referencias
        public int Id { get; set; }
        3 referencias
        public bool EnUso { get; set; }

        4 referencias
        public void Navegar(string ruta)
        {
            Console.WriteLine($"Barco #{Id} navegando por la ruta: {ruta}");
        }
    }
```

Class CentroControlMaritimo

```
namespace ExamenU2
{
    5 referencias
    public class CentroControlMaritimo
    {
        // Variable estática que guarda la única instancia del centro
        private static CentroControlMaritimo _instancia;
        // Objeto para evitar conflictos si varios hilos acceden al mismo tiempo
        private static readonly object _lock = new object();

        1 referencia
        public string NombreCentro { get; private set; }

        1 referencia
        private CentroControlMaritimo()
        {
            NombreCentro = "Centro Global de Control Marítimo";
        }

        1 referencia
        public static CentroControlMaritimo Instancia
        {
            get
            {
                lock (_lock)
                {
                    if (_instancia == null)
                        _instancia = new CentroControlMaritimo();
                    return _instancia;
                }
            }
        }
        3 referencias
        public void EnviarMensaje(string mensaje)
        {
            Console.WriteLine($"\nCENTRO DE CONTROL: {mensaje}");
        }
    }
}
```

Class PoolBarco

```
namespace ExamenU2
{
    3 referencias
    public class PoolBarcos
    {
        private readonly List<Barco> _barcosDisponibles = new List<Barco>();
        private readonly List<Barco> _barcosEnUso = new List<Barco>();
        private readonly object _lock = new object();

        1 referencia
        public PoolBarcos(int cantidadInicial)
        {
            for (int i = 1; i <= cantidadInicial; i++)
                _barcosDisponibles.Add(new Barco { Id = i, EnUso = false });
        }

        4 referencias
        public Barco ObtenerBarco()
        {
            lock (_lock)
            {
                if (_barcosDisponibles.Count > 0)
                {
                    Barco barco = _barcosDisponibles[0];
                    _barcosDisponibles.RemoveAt(0);
                    barco.EnUso = true;
                    _barcosEnUso.Add(barco);
                    Console.WriteLine($"Se asignó el Barco #{barco.Id}");
                    return barco;
                }
                else
                {
                    Console.WriteLine("No hay barcos disponibles, espere...");
                    return null;
                }
            }
        }
```

```
        1 referencia
        public void LiberarBarco(Barco barco)
        {
            lock (_lock)
            {
                barco.EnUso = false;
                _barcosEnUso.Remove(barco);
                _barcosDisponibles.Add(barco);
                Console.WriteLine($"El Barco #{barco.Id} regresó al puerto y esta disponible.");
            }
        }
    }
}
```

Class Program

```
namespace ExamenU2
{
    0 referencias
    class Program
    {
        0 referencias
        static void Main(string[] args)
        {
            var centro = CentroControlMaritimo.Instancia;
            centro.EnviarMensaje("Iniciando control de flotas...");

            PoolBarcos pool = new PoolBarcos(3);

            Barco b1 = pool.ObtenerBarco();
            Barco b2 = pool.ObtenerBarco();
            Barco b3 = pool.ObtenerBarco();

            b1.Navegar("Ruta Tijuana - Ensenada");
            b2.Navegar("Ruta Mazatlan - La Paz");
            b3.Navegar("Ruta Veracruz - Cozumel");

            centro.EnviarMensaje("Monitoreando rutas maritimas...");

            // Liberar un barco y reutilizarlo
            pool.LiberarBarco(b2);
            Barco b4 = pool.ObtenerBarco();
            b4.Navegar("Ruta Manzanillo - Acapulco");

            centro.EnviarMensaje("Fin de simulacion marítima.");
            Console.ReadLine();
        }
    }
}
```

Ejecución

```
C:\Users\edgar\source\repos\

CENTRO DE CONTROL: Iniciando control de flotas...
Se asignó el Barco #1
Se asignó el Barco #2
Se asignó el Barco #3
Barco #1 navegando por la ruta: Ruta Tijuana - Ensenada
Barco #2 navegando por la ruta: Ruta Mazatlan - La Paz
Barco #3 navegando por la ruta: Ruta Veracruz - Cozumel

CENTRO DE CONTROL: Monitoreando rutas maritimas...
El Barco #2 regresó al puerto y esta disponible.
Se asignó el Barco #2
Barco #2 navegando por la ruta: Ruta Manzanillo - Acapulco

CENTRO DE CONTROL: Fin de simulacion marítima.
```

Conclusión

Este programa nos ayudó a entender cómo se pueden controlar los barcos y sus rutas usando algunos patrones de diseño, se usó uno que hace que solo haya un centro de control y otro que sirve para no estar creando y borrando barcos a cada rato, sino reutilizarlos.

La verdad es que el programa funciona bien porque muestra cómo se pueden organizar mejor los recursos y tener todo más ordenado, además ayuda a que el sistema sea más rápido y fácil de manejar. El trabajo demuestra cómo se puede aplicar la programación orientada a objetos de una forma más práctica y sencilla para resolver un problema real, como controlar una flota de barcos.