Introducción

En este documento se detalla la implementación del proyecto de la Aseguradora, donde se mostrará con ejemplos cómo utilizar todas las funcionalidades realizadas.
La implementación de los repositorios no se desarrollará más allá de la introducción dado que son soluciones temporales de la entrega parcial.
Para poder persistir las diversas entidades se ha implementado en cada repositorio una propiedad ID la cual a su vez persiste la última ID utilizada y de este modo no perder el contador si el programa debe reiniciarse, para lograr esto se dispone de un archivo de texto donde cada repositorio almacena con su clave el valor de su última ID, dentro de dicho archivo podemos encontrar líneas como estas:

```
Titular:3
Vehiculo:7
```

Al instanciar los repositorios, el contador recupera la última ID para poder seguir persistiendo entidades de manera coherente.
Las entidades, interfaces y casos de uso se encuentran en la biblioteca de clases Aseguradora.Aplicacion, a su vez, las interfaces de los repositorios se encuentran implementadas en Aseguradora.Repositorios.

Cómo usar los métodos implementados

En la cabecera del archivo *program.cs* que se encuentra en Aseguradora.Consola (o donde usemos las clases creadas), se debe especificar el namespace al que pertenecen dichas clases y repositorios, siendo en este caso Aseguradora.Aplicacion y Aseguradora.Repositorios.

```
using Aseguradora.Repositorio;
using Aseguradora.Aplicacion;
```

Instanciar Titulares

Para instanciar titulares es necesario pasar al constructor el DNI, Apellido y Nombre, luego opcionalmente por medio de propiedades se puede agregar Telefono, Email, Lista de ID's de sus vehiculos e incluso la direccion del titular.

```
Titular seniora = new Titular(45356789,"Loyola","Nicole")
{Telefono="2214567890",
Email="nickyNicole@ayuda.com",
Direccion="Nadie sabe"};
Titular senior = new Titular(44130359,"Macias","Luciano")
{Telefono="221 345-1234"};
Titular generico = new Titular(35130300,"Leoncio","Jose");
```

Casos de Uso para Titulares

```
using Aseguradora.Repositorio;
using Aseguradora.Aplicacion;
// instanciamos caso de uso e inyectamos la dependencia
RepoTitularTXT repoTitu = new RepoTitularTXT();
AgregarTitularUseCase agregarTitular = new AgregarTitularUseCase(repoTitu);
ListarTitularesUseCase listarTitulares = new ListarTitularesUseCase(repoTitu);
ModificarTitularUseCase modificarTitular = new ModificarTitularUseCase(repoTitu);
EliminarTitularUseCase eliminarTitular = new EliminarTitularUseCase(repoTitu);

// instanciamos un titular
Titular T = new Titular(27354200,"Suster","Laura"){Email="layu208@hotmail.com"};
Console.WriteLine("ID del titular recien instaciado: {0}",T.ID);

// persistimos el titular
agregarTitular.Ejecutar(T);
Console.WriteLine("ID del titular luego de persistirlo: {0}",T.ID);

// agregamos unos titulares más
agregarTitular.Ejecutar(new Titular(44130300,"Macias","Luciano"){Direccion="15 e/ 54 y
55"});
agregarTitular.Ejecutar(new Titular(44130303,"Loyola","Nicole"));

// listamos los titulares
List<Titular> listaTitulares = listarTitulares.Ejecutar();
foreach(Titular ti in listaTitulares){
    Console.WriteLine(ti);
}

// tratamos de agregar un titular con dni ya existente
T = new Titular(44130300,"Picapiedra","Pedro");
Console.WriteLine("Intentando agregar titular con DNI: {0}",T.DNI);
try{
    // si ya existe retorna una excepcion
    agregarTitular.Ejecutar(T);
}catch (Exception e){
    // si la excepcion es que ya existe, lo modificamos
    if(e.Message == $"Ya existe Titular con DNI: {T.DNI}"){
        Console.WriteLine(e.Message);
        Console.WriteLine($"Modificando Titular con DNI: {T.DNI}");
        modificarTitular.Ejecutar(T);
    }
}

// vemos los cambios de modificar el titular
```

```
listaTitulares = listarTitulares.Ejecutar();
foreach (Titular ti in listaTitulares){
    Console.WriteLine(ti);
}

// eliminamos un titular
Console.WriteLine("Eliminando titular con DNI: {0}",listaTitulares[1].DNI);
eliminarTitular.Ejecutar(listaTitulares[1].ID);

// vemos los titulares sin el eliminado
listaTitulares = listarTitulares.Ejecutar();
foreach (Titular ti in listaTitulares){
    Console.WriteLine(ti);
}
```

La salida por consola de este código de ejemplo es la siguiente:

```
ID del titular recien instaciado: -1
ID del titular luego de persistirlo: 1
1: DNI:27354200 Apellido:Suster Nombre:Laura Email:layu208@hotmail.com
2: DNI:44130300 Apellido:Macias Nombre:Luciano Direccion:15 e/ 54 y 55
3: DNI:44130303 Apellido:Loyola Nombre:Nicole
Intentando agregar titular con DNI: 44130300
Ya existe Titular con DNI: 44130300
Modificando Titular con DNI: 44130300
1: DNI:27354200 Apellido:Suster Nombre:Laura Email:layu208@hotmail.com
2: DNI:44130300 Apellido:Picapiedra Nombre:Pedro
3: DNI:44130303 Apellido:Loyola Nombre:Nicole
Eliminando titular con DNI: 44130300
1: DNI:27354200 Apellido:Suster Nombre:Laura Email:layu208@hotmail.com
3: DNI:44130303 Apellido:Loyola Nombre:Nicole
```

Si se quisiera eliminar un Titular con un DNI determinado en nuestra implementacion

Instanciar Vehiculos

Los vehiculos son sencillos de instanciar, reciben todas sus propiedades a traves del constructo, por ejemplo:

```
Vehiculo V = new Vehiculo("Patente/Dominio","Marca",anoFabricacion:2020,titular:1);
```

(“anoFabricacion:” y “titular:” es opcional ponerlos, en este caso es visible para mostrar en el ejemplo a que corresponden esos parametros enteros)

Casos de Uso para Vehiculos

```
using Aseguradora.Repositorio;
using Aseguradora.Aplicacion;
// instanciamos casos de uso e inyectamos dependencia
RepoVehiculoTXT repoVehi = new RepoVehiculoTXT();
```

```
AgregarVehiculoUseCase agregarVehiculo = new AgregarVehiculoUseCase(repoVehi);
ModificarVehiculoUseCase modificarVehiculo = new
ModificarVehiculoUseCase(repoVehi);
EliminarVehiculoUseCase eliminarVehiculo = new
EliminarVehiculoUseCase(repoVehi);
ListarVehiculosUseCase listarVehiculos = new ListarVehiculosUseCase(repoVehi);

// instanciamos un vehiculo
Vehiculo V = new Vehiculo("AB123EF","Mercedes",2023,1);
Console.WriteLine("ID vehiculo antes de persistir: {0}",V.ID);
// persistimos el vehiculo
agregarVehiculo.Ejecutar(V);
Console.WriteLine("ID vehiculo luego de persistir: {0}",V.ID);

// agregamos unos vehiculos mas
agregarVehiculo.Ejecutar(new Vehiculo("EF320DE","Ford",2023,1));
agregarVehiculo.Ejecutar(new Vehiculo("DRO 345","Ford",2010,2));

// listamos los vehiculos
List<Vehiculo> listaV = listarVehiculos.Ejecutar();
foreach(Vehiculo ve in listaV){
    Console.WriteLine(ve);
}

// tratamos de persistir un vehiculo con un dominio existente
V = new Vehiculo("AB123EF","Renault",2022,2);
try{
    agregarVehiculo.Ejecutar(V);
}catch(Exception e){
    if(e.Message == $"Ya existe Vehiculo con Dominio: {V.Dominio}"){
        Console.WriteLine(e.Message);
        Console.WriteLine("Modificando Vehiculo con Dominio {0}",V.Dominio);
        modificarVehiculo.Ejecutar(V);
    }
}

// listamos vehiculos para comprobar la modificacion
listaV = listarVehiculos.Ejecutar();
foreach (Vehiculo ve in listaV){
    Console.WriteLine(ve);
}
```

```
// eliminaremos vehiculo por ID
Console.WriteLine("Eliminando vehiculo con ID: 1");
eliminarVehiculo.Ejecutar(1);
// nota: para eliminar el vehiculo de la lista de un titular se debe hacer de
manera manual

// listamos vehiculos para comprobar la modificacion
listaV = listarVehiculos.Ejecutar();
foreach (Vehiculo ve in listaV){
    Console.WriteLine(ve);
}
```

La salida por consola de este ejemplo es la siguiente:

```
ID vehiculo antes de persistir: -1
ID vehiculo luego de persistir: 1
1: Dominio:AB123EF  Marca:Mercedes  AnoFabricacion:2023 Titular:1
2: Dominio:EF320DE  Marca:Ford  AnoFabricacion:2023 Titular:1
3: Dominio:DRO 345  Marca:Ford  AnoFabricacion:2010 Titular:2
Ya existe Vehiculo con Dominio: AB123EF
Modificando Vehiculo con Dominio AB123EF
1: Dominio:AB123EF  Marca:Renault  AnoFabricacion:2022 Titular:2
2: Dominio:EF320DE  Marca:Ford  AnoFabricacion:2023 Titular:1
3: Dominio:DRO 345  Marca:Ford  AnoFabricacion:2010 Titular:2
Eliminando vehiculo con ID: 1
2: Dominio:EF320DE  Marca:Ford  AnoFabricacion:2023 Titular:1
3: Dominio:DRO 345  Marca:Ford  AnoFabricacion:2010 Titular:2
```

Instanciar Polizas

Las polizas al igual que los vehiculos son sencillos de instanciar ya que reciben todas sus propiedades a traves del constructor, por ejemplo:

```
// id_asegurado, valor_asegurado, franquicia, cobertura, vigenteDesde, vigenteHasta
Poliza P = new Poliza(1,20000,"","",new DateTime(2020,3,14),new DateTime(2024,3,14));
```

Casos de Uso para Polizas

```
using Aseguradora.Repositorio;
using Aseguradora.Aplicacion;
// instanciamos casos de uso e inyectamos las dependencias
RepoPolizaTXT repoPoli = new RepoPolizaTXT();
AgregarPolizaUseCase agregarPoliza = new AgregarPolizaUseCase(repoPoli);
ModificarPolizaUseCase modificarPoliza = new ModificarPolizaUseCase(repoPoli);
EliminarPolizaUseCase eliminarPoliza = new EliminarPolizaUseCase(repoPoli);
ListarPolizasUseCase listarPolizas = new ListarPolizasUseCase(repoPoli);
```

```
// instanciamos una poliza
// id_asegurado, valor_asegurado, franquicia, cobertura, vigenteDesde, vigenteHasta
Poliza P = new Poliza(1,20000,"0","Todo Riesgo",new DateTime(2020,3,14),new
DateTime(2024,3,14));
Console.WriteLine("ID poliza antes de persistir: {0}", P.ID);

// persistimos la poliza
agregarPoliza.Ejecutar(P);
Console.WriteLine("ID poliza luego de persistir: {0}", P.ID);

// agregamos unas polizas mas
agregarPoliza.Ejecutar(new Poliza(2, 10000, "2500", "Todo Riesgo", new DateTime(2021,
3, 14), new DateTime(2025, 3, 14)));
agregarPoliza.Ejecutar(new Poliza(3, 15000, "15000", "Responsabilidad Civil", new
DateTime(2019, 7, 14), new DateTime(2023, 7, 14)));

// listamos polizas
List<Poliza> polizas = listarPolizas.Ejecutar();
foreach(Poliza po in polizas){
    Console.WriteLine(po);
}

// modificamos una poliza existente
Console.WriteLine("Modificando la poliza de ID: "+polizas[2].ID);
polizas[2].Cobertura = "Todo Riesgo";
polizas[2].ValorAsegurado = 50000;
// la modificacion usa la ID de la poliza
modificarPoliza.Ejecutar(polizas[2]);

// listamos para ver la modificacion
polizas = listarPolizas.Ejecutar();
foreach (Poliza po in polizas){
    Console.WriteLine(po);
}

// eliminamos una poliza existente por medio de su ID
Console.WriteLine("Eliminando poliza con ID: 1");
eliminarPoliza.Ejecutar(1);

// listamos para ver la modificacion
polizas = listarPolizas.Ejecutar();
foreach (Poliza po in polizas){
    Console.WriteLine(po);
}
```

La salida por consola del ejemplo anterior es la siguiente:

```
ID poliza antes de persistir: -1
ID poliza luego de persistir: 1
1: VehiculoAsegurado:1 ValorAsegurado:20000 Franquicia:0  Cobertura:Todo Riesgo
VigenteDesde:14/3/2020 VigenteHasta:14/3/2024
2: VehiculoAsegurado:2 ValorAsegurado:10000 Franquicia:2500  Cobertura:Todo Riesgo
VigenteDesde:14/3/2021 VigenteHasta:14/3/2025
3: VehiculoAsegurado:3 ValorAsegurado:15000 Franquicia:15000
Cobertura:Responsabilidad Civil  VigenteDesde:14/7/2019 VigenteHasta:14/7/2023
Modificando la poliza de ID: 3
1: VehiculoAsegurado:1 ValorAsegurado:20000 Franquicia:0  Cobertura:Todo Riesgo
VigenteDesde:14/3/2020 VigenteHasta:14/3/2024
2: VehiculoAsegurado:2 ValorAsegurado:10000 Franquicia:2500  Cobertura:Todo Riesgo
VigenteDesde:14/3/2021 VigenteHasta:14/3/2025
3: VehiculoAsegurado:3 ValorAsegurado:50000 Franquicia:15000   Cobertura:Todo Riesgo
VigenteDesde:14/7/2019 VigenteHasta:14/7/2023
Eliminando poliza con ID: 1
2: VehiculoAsegurado:2 ValorAsegurado:10000 Franquicia:2500  Cobertura:Todo Riesgo
VigenteDesde:14/3/2021 VigenteHasta:14/3/2025
3: VehiculoAsegurado:3 ValorAsegurado:50000 Franquicia:15000   Cobertura:Todo Riesgo
VigenteDesde:14/7/2019 VigenteHasta:14/7/2023
```

Algo a destacar es que las pólizas no tienen un identificador único por lo que podrian existir varias polizas similares.

Caso de Uso Listar Titulares con sus Vehiculos

Este es un caso de uso especial ya que recibe 2 inyecciones de dependencia para poder funcionar, este caso de uso se encarga de completar la lista de vehiculos de un titular a partir de la lista de las ID's.
En titular tenemos:

```
public List<int> ListaVehiculos {get;set;} = new List<int>();
public List<Vehiculo> ListaVehiculosInfo {get;set;} = new List<Vehiculo>();
```

ListaVehiculos es la que lleva las ID's para tenerlas cuando persistimos los datos, luego si quisieramos una lista de Titulares con la informacion completa de sus vehiculos, usamos ListarTitularesConSusVehiculosUseCase que se encarga de completar esta segunda ListaVehiculosInfo.

```
using Aseguradora.Repositorio;
using Aseguradora.Aplicacion;

// caso de uso con su inyeccion de dependencia
RepoVehiculoTXT repoV = new RepoVehiculoTXT();
RepoTitularTXT repoT = new RepoTitularTXT();
ListarTitularesConSusVehiculosUseCase listarTconV = new
ListarTitularesConSusVehiculosUseCase(repoV,repoT);
```

Los archivos que persisten los vehículos y los titulares se encuentran de este modo para el ejemplo:

Titulares:

```
1: DNI: 27354200 Apellido: Suster Nombre:Laura Email:layu208 @hotmail.com Vehiculos:2,3
3: DNI: 44130303 Apellido: Loyola Nombre:Nicole
```

Vehículos:

```
2: Dominio: EF320DE Marca:Ford AnoFabricacion:2023 Titular: 1
3: Dominio: DRO 345 Marca: Ford AnoFabricacion:2010 Titular: 1
4: Dominio: ION 115 Marca: Mercedes AnoFabricacion:2014 Titular: 3
```

Al usar el siguiente código:

```
List<Titular> listaT = listarTconV.Ejecutar();
foreach(Titular T in listaT){
    Console.WriteLine($"Titular {T.Apellido}, {T.Nombre} tiene
{T.ListaVehiculosInfo.Count} vehiculos.");
    if(T.ListaVehiculosInfo.Count > 0){
        foreach(Vehiculo V in T.ListaVehiculosInfo){
            Console.WriteLine("└──> "+V);
        }
    }
}
```

La salida por consola queda:

```
Titular Suster, Laura tiene 2 vehiculos.
└──> 2: Dominio:EF320DE  Marca:Ford  AnoFabricacion:2023 Titular:1
└──> 3: Dominio:DRO 345  Marca:Ford  AnoFabricacion:2010 Titular:1
3: DNI:44130303 Apellido:Loyola Nombre:Nicole
Titular Loyola, Nicole tiene 1 vehiculos.
└──> 4: Dominio:ION 115  Marca:Mercedes  AnoFabricacion:2014 Titular:3
```