```
// Car.cs
class Car : Vehicle, IRegistrable
{
    public static int Counter = 0;
    public string Model { get; set; } = string.Empty;
    public string LicensePlate { get; set; } = string.Empty;
    public int Year { get; set; }

    public Car() => Counter++;

    public override string GetInfo() =>
        $"{Brand} {Model} ({LicensePlate}), évjárat: {Year}";

    public string GetLicensePlate() => LicensePlate;
}

// Vehicle.cs
class Vehicle
{
    public string Brand { get; set; } = string.Empty;

    public virtual string GetInfo() =>
        $"Brand: {Brand}";
}

// IRegistrable.cs
interface IRegistrable
{
    string GetLicensePlate();
}

// Document.cs
abstract class Document
{
    public string Details { get; protected set; } = string.Empty;
    public abstract void Print();
}

// CarInfo.cs
class CarInfo : Document
{
    private Car car;

    public CarInfo(Car car)
    {
        this.car = car;
        Details = $"Részletek: {car.GetInfo()}";
    }

    public override void Print()
    {
        Console.WriteLine(Details);
        Console.WriteLine($"Évjárat: {car.Year}");
```

```
    }
}

// Repository.cs
class Repository<T>
{
    private List<T> items = new List<T>();

    public void Add(T item) => items.Add(item);

    public T? FindByLicensePlate(string licensePlate)
    {
        foreach (T item in items)
        {
            if (item is Car car && car.LicensePlate == licensePlate)
                return item;
        }
        return default;
    }

    public List<T> GetAll() => items;
}

// Program.cs
using System.Text.Json;

internal class Program
{
    static void Main()
    {
        List<Car> cars = LoadCars();

        var repo = new Repository<Car>();
        foreach (var car in cars) repo.Add(car);

        var found = repo.FindByLicensePlate("XYZ-789");
        if (found != null)
        {
            Console.WriteLine("\nTalált autó:");
            Console.WriteLine(found.GetInfo());
        }

        var filteredCars = cars
            .Where(c => c.Year > 2019)
            .OrderBy(c => c.Brand)
            .ThenBy(c => c.Year);

        foreach (var car in filteredCars)
            Console.WriteLine($"{car.Brand} {car.Model}-{car.Year}");
    }

    static List<Car> LoadCars()
    {
```

```
        string json = File.ReadAllText("cars.json");
        List<Car>? cars = JsonSerializer.Deserialize<List<Car>>(json);

        if (cars != null)
        {
            foreach (var car in cars)
                Console.WriteLine($"{car.Brand} {car.Model} ({car.LicensePlate})");

            Console.WriteLine($"Counter: {Car.Counter}");
            return cars;
        }

        Console.WriteLine("A JSON beolvasása nem sikerült vagy üres.");
        return new List<Car>();
    }
}
```

cars.json

```
[
  {
    "Brand": "Toyota",
    "Model": "Corolla",
    "LicensePlate": "ABC-123",
    "Year": 2020
  },
  {
    "Brand": "Ford",
    "Model": "Focus",
    "LicensePlate": "XYZ-789",
    "Year": 2019
  },
  {
    "Brand": "Mazda",
    "Model": "3",
    "LicensePlate": "DEF-456",
    "Year": 2021
  }
]
```