**Цель работы –** приобрести умение работать с документами XML с помощью технологии LINQ to XML.

Задание: разработать консольное приложение на языке C#, в котором с помощью классов System.Xml.Linq “с нуля” создаётся одна ветвь XML-документа.

Код метода для создания ветви XML-документа:

**private** **static** **void** **CreateXmlDocument**()

{

**var** doc = **new** XDocument(

**new** **XDeclaration**("1.0", "utf-8", "yes"),

**new** **XElement**("автодилер",

**new** **XElement**("машина", **new** XAttribute("номер", **1**),

**new** **XElement**("модель", "Mark2"),

**new** **XElement**("фирма", "Toyota"),

**new** **XElement**("цена", **800000**),

**new** **XElement**("кузов", "седан"),

**new** **XElement**("ответственный\_работник", "Анатолий"),

**new** **XElement**("мощность", "200")

),

**new** **XElement**("машина", **new** XAttribute("номер", **2**),

**new** **XElement**("модель", "A80"),

**new** **XElement**("фирма", "Toyota"),

**new** **XElement**("цена", **2400000**),

**new** **XElement**("кузов", "купе"),

**new** **XElement**("ответственный\_работник", "Дмитрий"),

**new** **XElement**("мощность", "300")

),

**new** **XElement**("машина", **new** XAttribute("номер", **3**),

**new** **XElement**("модель", "GT-R"),

**new** **XElement**("фирма", "Nissan"),

**new** **XElement**("цена", **4600000**),

**new** **XElement**("кузов", "купе"),

**new** **XElement**("ответственный\_работник", "Анатолий"),

**new** **XElement**("мощность", "550")

),

**new** **XElement**("машина", **new** XAttribute("номер", **4**),

**new** **XElement**("модель", "Civic"),

**new** **XElement**("фирма", "Honda"),

**new** **XElement**("цена", **900000**),

**new** **XElement**("кузов", "хетчбек"),

**new** **XElement**("ответственный\_работник", "Дмитрий"),

**new** **XElement**("мощность", "150")

),

**new** **XElement**("машина", **new** XAttribute("номер", **5**),

**new** **XElement**("модель", "RX-7"),

**new** **XElement**("фирма", "Mazda"),

**new** **XElement**("цена", **2400000**),

**new** **XElement**("кузов", "купе"),

**new** **XElement**("ответственный\_работник", "Вячеслав"),

**new** **XElement**("мощность", "300")

)

)

);

doc.Save("CarDealer.xml");

}

Результат выполнения:

<?xml version="1.0" encoding="utf-8" standalone="yes"?>

<автодилер>

<машина номер="1">

<модель>Mark2</модель>

<фирма>Toyota</фирма>

<цена>800000</цена>

<кузов>седан</кузов>

<ответственный\_работник>Анатолий</ответственный\_работник>

<мощность>200</мощность>

</машина>

<машина номер="2">

<модель>A80</модель>

<фирма>Toyota</фирма>

<цена>2400000</цена>

<кузов>купе</кузов>

<ответственный\_работник>Дмитрий</ответственный\_работник>

<мощность>300</мощность>

</машина>

<машина номер="3">

<модель>GT-R</модель>

<фирма>Nissan</фирма>

<цена>4600000</цена>

<кузов>купе</кузов>

<ответственный\_работник>Анатолий</ответственный\_работник>

<мощность>550</мощность>

</машина>

<машина номер="4">

<модель>Civic</модель>

<фирма>Honda</фирма>

<цена>900000</цена>

<кузов>хетчбек</кузов>

<ответственный\_работник>Дмитрий</ответственный\_работник>

<мощность>150</мощность>

</машина>

<машина номер="5">

<модель>RX-7</модель>

<фирма>Mazda</фирма>

<цена>2400000</цена>

<кузов>купе</кузов>

<ответственный\_работник>Вячеслав</ответственный\_работник>

<мощность>300</мощность>

</машина>

</автодилер>

Задание: добавить метод в приложение, который с помощью запроса LINQ преобразовывает данные из массива объектов в XML.

Для этого задания был взят класс Laptop из лабораторной работы № 1.

Код преобразования объектов в XML:

**private** **static** **void** **GenerXML**(Laptop[] laptops)

{

**var** xmlData = **from** laptop **in** laptops

**select**

**new** **XElement**("Laptop",

**new** **XAttribute**("Id", laptop.Id),

**new** **XAttribute**("StoreId", laptop.StoreId),

**new** **XElement**("Company", laptop.Company),

**new** **XElement**("Model", laptop.Model),

**new** **XElement**("CPU", laptop.CPU),

**new** **XElement**("RAM", laptop.RAM),

**new** **XElement**("Price", laptop.Price)

);

XElement rootElement = **new** XElement("LaptopStock", xmlData);

XDocument xmlDoc = **new** XDocument();

xmlDoc.Add(rootElement);

xmlDoc.Save("laptops.xml");

}

Результат:

<?xml version="1.0" encoding="utf-8"?>

<LaptopStock>

<Laptop Id="0" StoreId="0">

<Company>ASUS</Company>

<Model>Laptop 14 F415EA-EB736</Model>

<CPU>Intel Pentium Gold 7505</CPU>

<RAM>8</RAM>

<Price>30999</Price>

</Laptop>

<Laptop Id="1" StoreId="1">

<Company>HP</Company>

<Model>15s-eq1322ur</Model>

<CPU>AMD 3020e</CPU>

<RAM>8</RAM>

<Price>32999</Price>

</Laptop>

<Laptop Id="2" StoreId="2">

<Company>Acer</Company>

<Model>Aspire 3 A315-56-34Q8</Model>

<CPU>Intel Core i3-1005G1</CPU>

<RAM>4</RAM>

<Price>34999</Price>

</Laptop>

<Laptop Id="3" StoreId="0">

<Company>HP</Company>

<Model>Laptop 15s-eq1142ur</Model>

<CPU>AMD Athlon Silver 3050U</CPU>

<RAM>8</RAM>

<Price>36999</Price>

</Laptop>

<Laptop Id="4" StoreId="1">

<Company>Acer</Company>

<Model>Swift 3 SF314-43</Model>

<CPU>AMD Ryzen 3 5300U</CPU>

<RAM>8</RAM>

<Price>42999</Price>

</Laptop>

<Laptop Id="5" StoreId="2">

<Company>ASUS</Company>

<Model>Laptop 14 D415DA-EK614T</Model>

<CPU>AMD Ryzen 3 3250U</CPU>

<RAM>8</RAM>

<Price>44999</Price>

</Laptop>

<Laptop Id="6" StoreId="0">

<Company>HP</Company>

<Model>15s-fq2018ur</Model>

<CPU>Intel Core i3-1115G4</CPU>

<RAM>8</RAM>

<Price>47999</Price>

</Laptop>

<Laptop Id="7" StoreId="1">

<Company>ASUS</Company>

<Model>VivoBook Flip 14 TM420UA-EC063T</Model>

<CPU>AMD Ryzen 3 5300U</CPU>

<RAM>4</RAM>

<Price>49999</Price>

</Laptop>

<Laptop Id="8" StoreId="2">

<Company>Lenovo</Company>

<Model>IdeaPad Flex 5 14ALC05</Model>

<CPU>AMD Ryzen 3 5300U</CPU>

<RAM>8</RAM>

<Price>51999</Price>

</Laptop>

<Laptop Id="9" StoreId="0">

<Company>HP</Company>

<Model>Pavilion Aero 13-be0050ur</Model>

<CPU>AMD Ryzen 5 5600U</CPU>

<RAM>8</RAM>

<Price>55999</Price>

</Laptop>

<Laptop Id="10" StoreId="1">

<Company>ASUS</Company>

<Model>VivoBook 15 X513EA-BQ2370W</Model>

<CPU>Intel Core i3-1115G4</CPU>

<RAM>8</RAM>

<Price>58699</Price>

</Laptop>

<Laptop Id="11" StoreId="2">

<Company>Acer</Company>

<Model>Aspire 3 A315-56-71MM</Model>

<CPU>Intel Core i7-1065G7</CPU>

<RAM>8</RAM>

<Price>61999</Price>

</Laptop>

<Laptop Id="12" StoreId="0">

<Company>Lenovo</Company>

<Model>Yoga Slim 7 14ARE05</Model>

<CPU>AMD Ryzen 5 4500U</CPU>

<RAM>8</RAM>

<Price>64999</Price>

</Laptop>

<Laptop Id="13" StoreId="1">

<Company>Dell</Company>

<Model>Inspiron 5515-0363</Model>

<CPU>AMD Ryzen 7 5700U</CPU>

<RAM>8</RAM>

<Price>65999</Price>

</Laptop>

<Laptop Id="14" StoreId="2">

<Company>HP</Company>

<Model>Pavilion Aero 13-be0005ur</Model>

<CPU>AMD Ryzen 5 5600U</CPU>

<RAM>16</RAM>

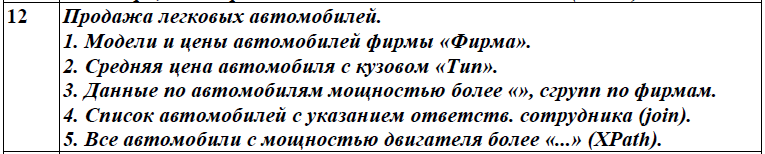
<Price>69999</Price>

</Laptop>

</LaptopStock>

Задание: разработать консольное приложение на языке C#, которое обеспечивает выполнение заданных запросов к XML-документу.

Вариант 12.

****

Для выполнения этого задания был взят XML-документ полученный в первом задании (создание ветви XML-документа “с нуля”).

Метод Main:

Console.Title = "Выполнение запросов LINQ к XML-документу";

XDocument xmldoc = XDocument.Load("CarDealer.xml");

Console.WriteLine("........Результаты запросов LINQ к XML-документу........");

**string** manufacturer = "Toyota";

//1. Модели и цены автомобилей фирмы «Фирма».

GetByManufacturer(xmldoc, manufacturer);

**string** bodyType = "купе";

//2. Средняя цена автомобиля с кузовом «Тип».

GetAveragePriceByBodyType(xmldoc, bodyType);

**int** power = **200**;

//3. Данные по автомобилям мощностью более «», сгрупп по фирмам.

GetByPowerGroupByManufacturer(xmldoc, power);

//4. Список автомобилей с указанием ответств. сотрудника (join).

GetByResponsibleEmployee(xmldoc);

**int** power2 = **200**;

//5. Все автомобили с мощностью двигателя более «...» (XPath).

GetByPowerXPath(xmldoc, power2);

1. Модели и цены автомобилей фирмы “Toyota”

Код метода:

**private** **static** **void** **GetByManufacturer**(XDocument xmldoc, **string** manufacturer)

{

**var** result = **from** car **in** xmldoc.Descendants("машина")

**where** (**string**)car.Element("фирма") == manufacturer

**select** car;

Console.WriteLine($"........Машины фирмы {manufacturer}........");

**foreach** (**var** car **in** result)

{

Console.WriteLine((**string**)car.Element("модель") + " - " + (**string**)car.Element("цена"));

}

}

Результат выполнения:



1. Средняя цена автомобиля с кузовом купе

Код метода:

**private** **static** **void** **GetAveragePriceByBodyType**(XDocument xmldoc, **string** bodyType)

{

**var** result = (**from** car **in** xmldoc.Descendants("машина")

**where** (**string**)car.Element("кузов") == bodyType

**select** Convert.ToDouble(car.Element("цена").Value)).Average();

Console.WriteLine($"........Средняя стоимость машин с кузовом типа {bodyType}........");

Console.WriteLine(result);

}

Результат выполнения:



1. Данные по автомобилям мощность более 200, сгруппированные по фирмам

Код метода:

**private** **static** **void** **GetByPowerGroupByManufacturer**(XDocument xmldoc, **int** power)

{

**var** groups = **from** car **in** xmldoc.Descendants("машина")

**where** (**int**)car.Element("мощность") > power

**group** car by car.Element("фирма");

Console.WriteLine($"........Машины мощности выше {power}........");

**foreach** (**var** **group** **in** groups)

{

Console.WriteLine($"Машины {(string)group.Key} мощностью выше {power}");

**foreach** (**var** car **in** **group**)

{

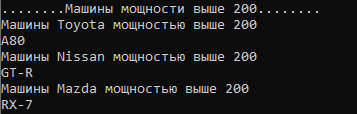
Console.WriteLine((**string**)car.Element("модель"));

}

}

}

Результат выполнения:



1. Список автомобилей с указанием ответственного сотрудника

Код метода:

**private** **static** **void** **GetByResponsibleEmployee**(XDocument xmldoc)

{

XElement employees = **new** XElement("работники\_салона",

**new** **XElement**("работник", **new** XAttribute("код", "а111"),

**new** **XAttribute**("номер\_машины", **1**),

**new** **XElement**("фио", "Буров Анатолий Григорьевич")

),

**new** **XElement**("работник", **new** XAttribute("код", "а112"),

**new** **XAttribute**("номер\_машины", **2**),

**new** **XElement**("фио", "Петров Дмитрий Александрович")

),

**new** **XElement**("работник", **new** XAttribute("код", "а113"),

**new** **XAttribute**("номер\_машины", **3**),

**new** **XElement**("фио", "Кудрова Татьяна Михайловна")

),

**new** **XElement**("работник", **new** XAttribute("код", "а114"),

**new** **XAttribute**("номер\_машины", **4**),

**new** **XElement**("фио", "Самойлов Константин Михайлович")

),

**new** **XElement**("работник", **new** XAttribute("код", "а115"),

**new** **XAttribute**("номер\_машины", **5**),

**new** **XElement**("фио", "Иванов Михаил Дмитриевич")

)

);

**var** result = **from** car **in** xmldoc.Descendants("машина")

**from** model **in** car.Elements("модель")

**join** emp **in** employees.Elements("работник")

on car.Attribute("номер").Value **equals** emp.Attribute("номер\_машины").Value

**select** **new** {

name = emp.Element("фио").Value,

carMan = car.Element("фирма").Value,

carMod = car.Element("модель").Value

};

Console.WriteLine($"........Работники автодилера и машины, за которые они ответственны........");

**foreach** (**var** r **in** result)

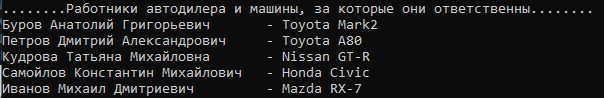
{

Console.WriteLine($"{r.name}\t - {r.carMan} {r.carMod}");

}

}

Результат выполнения:



1. Все автомобили с мощностью двигателя более 200. (XPath)

Код метода:

**private** **static** **void** **GetByPowerXPath**(XDocument xmldoc, **int** power)

{

**var** result = xmldoc.XPathSelectElements($"//машина[мощность>{power}]");

Console.WriteLine($"........Машины мощности выше {power}(XPath)........");

**foreach** (**var** car **in** result)

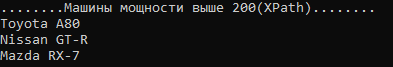
{

Console.WriteLine($"{car.Element("фирма").Value} {car.Element("модель").Value}");

}

}

Результат выполнения:



**Вывод:** я приобрёл навыки работы с XML-документами посредством технологии LINQ to XML.