**Министерство науки и высшего образования российской федерации**

**ФГАОУ ВО «Пермский государственный национальный исследовательский университет»**

Физический факультет

Технологии .NET Framework

Лабораторная работа №9 на тему

**«****Файловый ввод-вывод. Работа с XML файлами»**

Выполнил:

студент 2 курса физического факультета

группа ПМИ-2-2021

направление «01.03.02 Прикладная математика и информатика»

профиль «Инженерия программного обеспечения»

Макарова Полина Фёдоровна

Пермь, 2023 г.

1. Цель работы

• научиться использовать механизмы файлового ввода-вывода;

• научиться применять классы для работы с файлами;

• научиться применять классы для сериализации/десериализации;

• научиться использовать потоки ввода-вывода.

1. Постановка задачи

Выполните индивидуальное задание, согласно предложенному варианту.

Общее для всех заданий:

1. Задайте путь (диск:\имя каталога\имя файла.xml) к файлу \*.xml любым из известных вам способов;
2. Для сериализации/десериализации используйте «сложный» объект;
3. Вывод в консоль с использованием управляющих последовательностей (табуляция, перенос строки и т.п.) и, если уместно, форматного вывода;
4. В качестве объекта для сериализации взять массив;
5. В массиве минимум 2 члена (т.е., например, в варианте 1. это 2 «Библиотеки» - район Ленинский, район Дзержинский и т.д.);
6. В каждый дочерний (усложненный) элемент, также входит 2-3 элемента (в варианте 1. это 2-3 «Книги» и т.д.).

Формулировка индивидуального задания:

Спроектируйте и сериализуйте класс «Склад» с элементом «Корпус» с усложненными элементами «Инструменты», «Материалы», «Электрические компоненты». В каждый элемент входит элемент «наименование» с элементами масса, габариты, количество.

Объекты и явления предметной области:

1. Класс «Склад»;
2. Класс «Наименование»;
3. Загрузка и выгрузка из файла \*.xml, вывод на экран.
4. **Диаграмма классов**

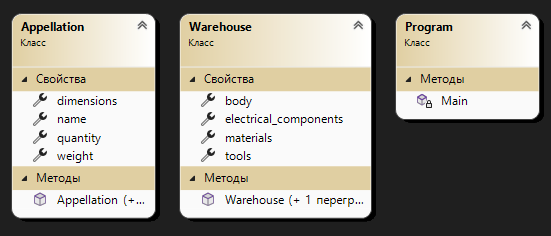


Рисунок — Диаграмма классов

1. Листинг программы

|  |
| --- |
| *Листинг 1 программы* |
| /\* Макарова Полина ПМИ-2  \* Вариант 10  \* Лабораторная работа 9  \*/  /\* Спроектируйте и сериализуйте класс «Склад» с элементом  \* «Корпус» с усложненными элементами «Инструменты», «Материалы»,  \* «Электрические компоненты». В каждый элемент входит элемент  \* «наименование» с элементами масса, габариты, количество.  \*/  using System.Xml.Serialization;  // Класс "Склад"  public class Warehouse  {  // Корпус  public string body { set; get; } = "Untitled";  // Инструменты  public Appellation[] tools { set; get; } = new Appellation[0];  // Материалы  public Appellation[] materials { set; get; } = new Appellation[0];  // Электрические компоненты  public Appellation[] electrical\_components { set; get; } = new Appellation[0];  // Конструктор по умолчанию  public Warehouse() { }  // Конструктор с параметрами  public Warehouse(string body, Appellation[] tools, Appellation[] materials, Appellation[] electrical\_components)  {  this.body = body;  this.tools = tools;  this.materials = materials;  this.electrical\_components = electrical\_components;  }  }  // Класс Наименования  public class Appellation  {  // Название (Логически будет непонятно, что за масса или габариты без названия)  public string name { set; get; } = "Not specified";  // Масса  public int weight { set; get; } = 1;  // Габариты (Высота, ширина, длина)  public string dimensions { set; get; } = "1x1x1";  // Количество  public int quantity { set; get; } = 1;  // Конструктор по умолчанию  public Appellation() { }  // Конструктор с параметрами  public Appellation(string name, int weight, string dimensions, int quantity)  {  this.name = name;  this.weight = weight;  this.dimensions = dimensions;  this.quantity = quantity;  }  }  class Program  {  static void Main(string[] args)  {  Console.SetWindowSize(90, 35);  Console.BackgroundColor = ConsoleColor.White;  Console.Clear();  Console.ForegroundColor = ConsoleColor.Black;  // Создание объектов класса "Наименование"  var tool1 = new Appellation("Лопата штыковая", 2, "1500х230х170", 2);  var tool2 = new Appellation("Лопата совковая", 2, "1400х230х170", 4);  var tool3 = new Appellation("Топор", 1, "178x92x49", 11);  var material1 = new Appellation("Алюминий волнистый", 940, "32x6x0,3", 110);  var material2 = new Appellation("Сталь волнистая", 9600, "1x100x30", 74);  var electrical\_component1 = new Appellation("Кабель на основе витой пары", 1000, "390x230x410", 305);  var electrical\_component2 = new Appellation("Коммутационный шнур", 7600, "400x400x260 ", 200);  var electrical\_component3 = new Appellation("Коммутационная панель", 1090, "520x410x240 ", 12);  // Создание объектов класса "Склад"  Warehouse[] WareHouse = new Warehouse[]  {  new Warehouse("А", new Appellation[]{tool1,tool2}, new Appellation[]{material1,material2}, new Appellation[]{electrical\_component1}),  new Warehouse("Б1", new Appellation[]{tool2}, new Appellation[]{material1}, new Appellation[]{electrical\_component1}),  new Warehouse("Б2", new Appellation[]{tool3}, new Appellation[]{material2}, new Appellation[]{electrical\_component2,electrical\_component3})  };  XmlSerializer formatter = new XmlSerializer(typeof(Warehouse[]));  using (FileStream fs = new FileStream("warehouses.xml", FileMode.OpenOrCreate))  {  formatter.Serialize(fs, WareHouse);  }  using (FileStream fs = new FileStream("warehouses.xml", FileMode.OpenOrCreate))  {  Warehouse[] newWarehouse = formatter.Deserialize(fs) as Warehouse[];  if (newWarehouse != null)  {  foreach (Warehouse warehouse in newWarehouse)  {  Console.WriteLine($"\nКорпус: {warehouse.body}");  Console.WriteLine("Инструменты:");  foreach (Appellation Appel in warehouse.tools)  {  Console.WriteLine($"\"{Appel.name}\", Вес: {Appel.weight} кг, Габариты: {Appel.dimensions} мм, Кол-во: {Appel.quantity} шт.");  }  Console.WriteLine("Материалы:");  foreach (Appellation Appel in warehouse.materials)  {  Console.WriteLine($"\"{Appel.name}\", Вес: {Appel.weight} гр/м, Габариты: {Appel.dimensions} мм, Кол-во: {Appel.quantity} шт.");  }  Console.WriteLine("Электрические компоненты:");  foreach (Appellation Appel in warehouse.electrical\_components)  {  Console.WriteLine($"\"{Appel.name}\", Вес: {Appel.weight} гр, Габариты: {Appel.dimensions} мм, Кол-во: {Appel.quantity} шт.");  }  }  }  }  Console.ReadKey();  }  } |

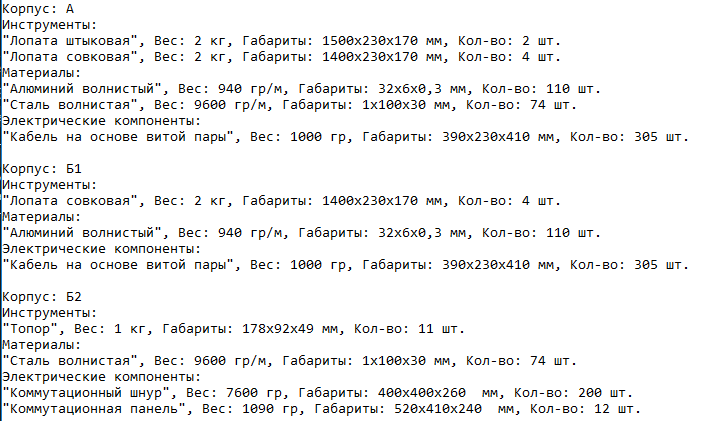


Рисунок — Результат выполнения программы в консоли

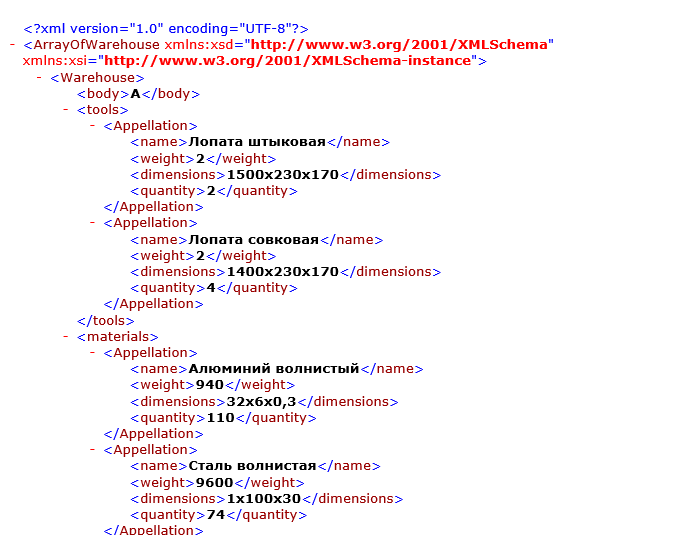


Рисунок — Результат выполнения программы в файле \*.xml, часть кода

1. Ответы на вопросы

***1. Что такое двоичный поток? Для работы с какими типами файлов предназначен?***

Двоичный поток состоит из одного или нескольких байтов произвольных данных. Можно записать значение, хранящееся в произвольном объекте, в двоичный (ориентированный на байты) поток и считать в точности те данные, которые хранились в объекте при его записи. Функции библиотеки не изменяют байты, передаваемые между программой и двоичным потоком.

Предназначен для работы с двоичными файлами.

***2. Основные классы для работы с двоичным потоком?***

BinaryReader и BinaryWriter

***3. Пространство имен и классы для работы с файлами типа XML?***

Пространство имен: System.Xml

Классы: XmlNode, XmlDocument, XmlElement, XmlAttribute, XmlText, XmlComment, XmlNodeList

***4. Что такое сериализация, десериализация?***

Сериализация — это процесс сохранения состояния объекта в последовательность байт.

Десериализация — это процесс восстановления объекта из этих байт.

***5. Для каких типов файлов реализована сериализация/десериализация в .NET? Приведите классы предназначенные для этого в C#.***

.json(JsonSerializer) и .xml (XmlSerializer)

***6. На примере вашего файла XML, опишите его структуру (элементы). (можно скриншот и подписи к элементам)***

XML документ должен содержать корневой элемент. Этот элемент является «родительским» для всех других элементов. Все элементы в XML документе формируют иерархическое дерево. Это дерево начинается с корневого элемента и разветвляется на более низкие уровни элементов.

Все элементы могут иметь подэлементы (дочерние элементы):

<корневой>

<потомок>

<подпотомок>.....</подпотомок>

</потомок>

</корневой>

***7. В пункте 3.1 представлены Диаграмма и Листинг программы с использованием для сериализации/десериализации сложного по составу объекта. В чем заключается его усложнение в сравнении ранее рассмотренным на лекции (NET Framework\_2022\_Лекция\_8\_2.mp4, слайд 49) сериализованным объектом. Как называется этот вид взаимодействия классов.***

В объекте есть массив объектов;

Вложенный объект/ массив объектов.