Тестовое задание №1

для претендентов на должность программиста CSharp

Необходимо реализовать два C# класса: **TagItem** и **TagStorage**.

Класс **TagItem** должен содержать:

1. Имя тэга
2. Полный пусть к тэгу. Имена родительских тэгов должны быть разделены через точку.
3. Полное имя тэга.
4. Тип данных тэга – **double**, **int**, **bool** или **none**, если не хранит значения. Тип данных тэга определяется на этапе создания экземпляра и остаётся неизменным.
5. Значение данных тэга, если применимо
6. Уровень вложенности тэга (**Level**)
7. Ссылку на родительский тэг
8. Список дочерних тэгов следующего уровня вложенности (**Level + 1**)

Класс **TagItem** должен содержать следующие методы для манипуляции со структурой тэгов:

1. Добавление дочернего тега
2. Удаление дочернего тэга
3. Поиск дочернего тэга следующего уровня по имени
4. Поиск дочернего тэга любого уровня вложенности по относительному (к себе) пути/имени тэга
5. Изменение родительского тэга (перемещение)

Пример:
Тэг с именем **tag1** содержит дочерний тэг с именем **tag2**, который содержит дочерний тэг с именем **tag3**, тогда полный путь к тэгу **tag3** будет равен **tag1.tag2**, полное имя тэга tag3 **tag1.tag2.tag3,** значение уровня вложенности для **tag3** равно трём.

Класс **TagStorage** содержит корневой тэг **Root** типа **none**, который является родителем всех тэгов, содержащихся в **TagStorage** и предоставляет методы по загрузке и выгрузке в XML-файл структуры дерева тэгов и их текущих значений.

Сохранение и загрузка структуры дерева тэгов, а также текущих значений тэгов в XML файл **не должна** производится встроенными средствами сериализации/десериализации объектов.

Возможности созданных классов должны быть продемонстрированы консольным приложением C#. Должен быть предоставлен файл XML со структурой тэгов для тестирования возможностей программы (уровень вложенности не менее 4, разные типы, не менее 20 тэгов в сумме)

Возможности меню программы:

1. Загрузка/выгрузка дерева тэгов и их значений из/в XML файл имя файла – константа)
2. Вывод построчного списка тэгов – полный путь, уровень вложенности, тип, значение
3. Удаление тэга по полному имени
4. Добавление нового тэга. Указываем полное имя родительского тэга, имя нового тэга и его тип
5. Переименование тэга. Указываем полный путь и новое имя тэга
6. Другие пункты по усмотрению, необходимые для демонстрации возможностей

Требования к реализации:

1. Версия .Net 4.5
2. Имена классов, методов, полей, переменных и т.д. должны быть на английском языке и иметь логический смысл.
3. **Должны соблюдаться** правила оформления кода приведённые в приложении №1.
4. Программа должна содержать комментарии (можно на русском языке) классов, методов и ключевых точек алгоритмов.
5. Все, что не оговорено выше остается на усмотрение претендента.

Приложение №1

Правила оформления кода

```
// Первая буква слова или аббревиатуры в имени начинается с большой буквы "UpperCamelCase" для:
// Названия пространств имен (Namespace), классов (class), Полей (Fields), Свойств (Property) и Методы классов (Method)

// Первая буква в названии с прописной буквы, все остальные с большой "lowerCamelCase" для:
// Названия переменных и параметров методов

using System;


public class ClassName
{
    // Имена неизменяемых полей пишутся в верхнем регистре,
    // в качестве разделения между словами используется нижнее подчеркивание "_"
    private const int CONST_FIELD = 18; // Верно
    private const int Const_Field = 18; // Неверно
    private const int ConstField = 18;  // Неверно

    // Приватные поля и свойства всегда начинаются с символа подчеркивания
    private bool _PrivateFieldA; // Верно
    private bool PrivateFieldB; // Неверно

    // В любых случая указывается модификатор доступа
    private bool _FieldA; // Верно
    bool _FieldB; // Неверно

    // Двухсимольные аббревиатуры:
    public bool FiledId; // Предпочтительно
    public bool FieldID; // Допускается

    // Многосимвольные аббревиатуры (GUID) - заглавный только первый символ
    public Guid FieldGuid; // Верно
    public Guid FieldGUID; // Неверно

    public void ClassMethodOne()
    {
        int localVarA = 3; // Верно
        int LocalVarB = 5; // Неверно
    }
}
```