**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«РОССИЙСКАЯ АКАДЕМИЯ НАРОДНОГО ХОЗЯЙСТВА   
И ГОСУДАРСТВЕННОЙ СЛУЖБЫ**

**ПРИ ПРЕЗИДЕНТЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»**

**НИЖЕГОРОДСКИЙ ИНСТИТУТ УПРАВЛЕНИЯ – филиал РАНХиГС**

Факультет управления

Кафедра Информатики и информационных технологий

Направление подготовки / специальность: 09.04.03 Прикладная информатика

Направленность (профиль) / специализация: Информационные технологии в ГМУ

**Отчет по лабораторным работам**

|  |  |
| --- | --- |
| по дисциплине: | Лингвистическое обеспечение web-взаимодействия |
|  |  |
|  |  |
|  |  |

|  |  |  |
| --- | --- | --- |
|  |  | **АВТОР** |
|  |  | Обучающийся 2 курса группы Ик-721 |
|  |  | заочной формы обучения |
|  |  |  |
|  |  | \_\_\_\_\_\_\_\_\_\_\_\_\_ Соколов Дмитрий Александрович |
|  |  | *(подпись) (фамилия, инициалы)* |
|  |  |  |
|  |  | **РУКОВОДИТЕЛЬ** |
|  |  | Кандидат физико-математических наук |
|  |  | *(ученая степень, ученое звание)* |
|  |  | оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  |  | «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_\_ г. |
|  |  | *(дата защиты)* |
|  |  | \_\_\_\_\_\_\_\_\_\_\_\_\_ Орлов М.Л. |
|  |  | *(подпись) (фамилия, инициалы)* |

Нижний Новгород, 2025г.

Оглавление

[Практическая работа №1 3](#_Toc196065441)

[Задание №1 3](#_Toc196065442)

[Задание №2 4](#_Toc196065443)

[Задание №3 8](#_Toc196065444)

[Практическая работа №2 12](#_Toc196065445)

[Задание №1 12](#_Toc196065446)

[Задание №2 13](#_Toc196065447)

[Практическая работа №3 16](#_Toc196065448)

[Задание №1 16](#_Toc196065449)

[Задание №2 20](#_Toc196065450)

[Практическая работа №4 23](#_Toc196065451)

[Задание №1 23](#_Toc196065452)

[Практическая работа №5 26](#_Toc196065453)

[Задание №1 26](#_Toc196065454)

[Задание №2 29](#_Toc196065455)

[Практическая работа №6 33](#_Toc196065456)

[Задание №1 33](#_Toc196065457)

[Задание №2 35](#_Toc196065458)

[Задание №3 36](#_Toc196065459)

[Задание №4 38](#_Toc196065460)

[Практическая работа №7 39](#_Toc196065461)

[Задание №1 39](#_Toc196065462)

[Практическая работа №8 46](#_Toc196065463)

[Задание №1 46](#_Toc196065464)

# Практическая работа №1

## Задание №1

Для выполнения задания создаем документ 1.xml. Для начала определяем основной тег <TOVAR>.

Затем добавил дочерние теги <NAME> и <PRACE>.

Внутрь дочерних тегов заносим информационную часть.

Указываем пролог - версию языка XML, кодировки документа и автономности. Необходимо указать кодировку UTF-8 для корректного отображения русского языка.

Результат работы - текстовое содержимое документа (рис.1):

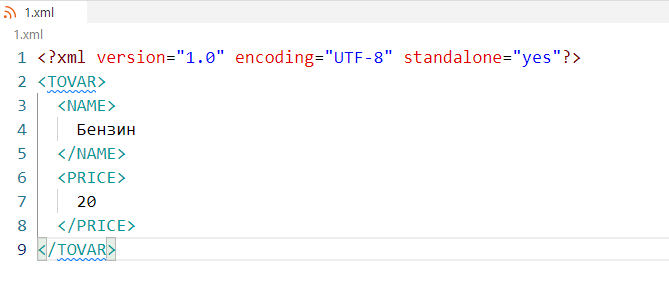


Рис. 1 Текстовое содержимое документа 1.xml

Далее проверяем структуру файла в браузере (рис.2).

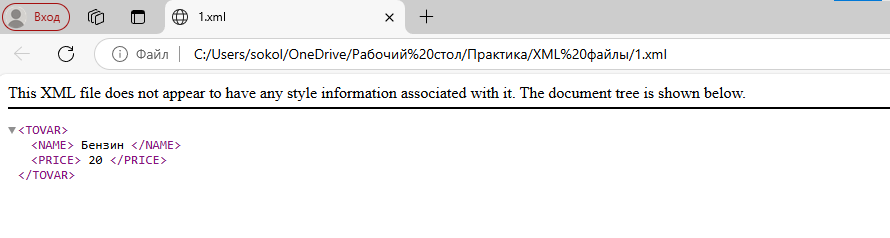


Рис. 2 Текстовое содержимое документа 1.xml

## Задание №2

Для упрощения создания xml документов устанавливаем приложение XmlPad (рис.3).

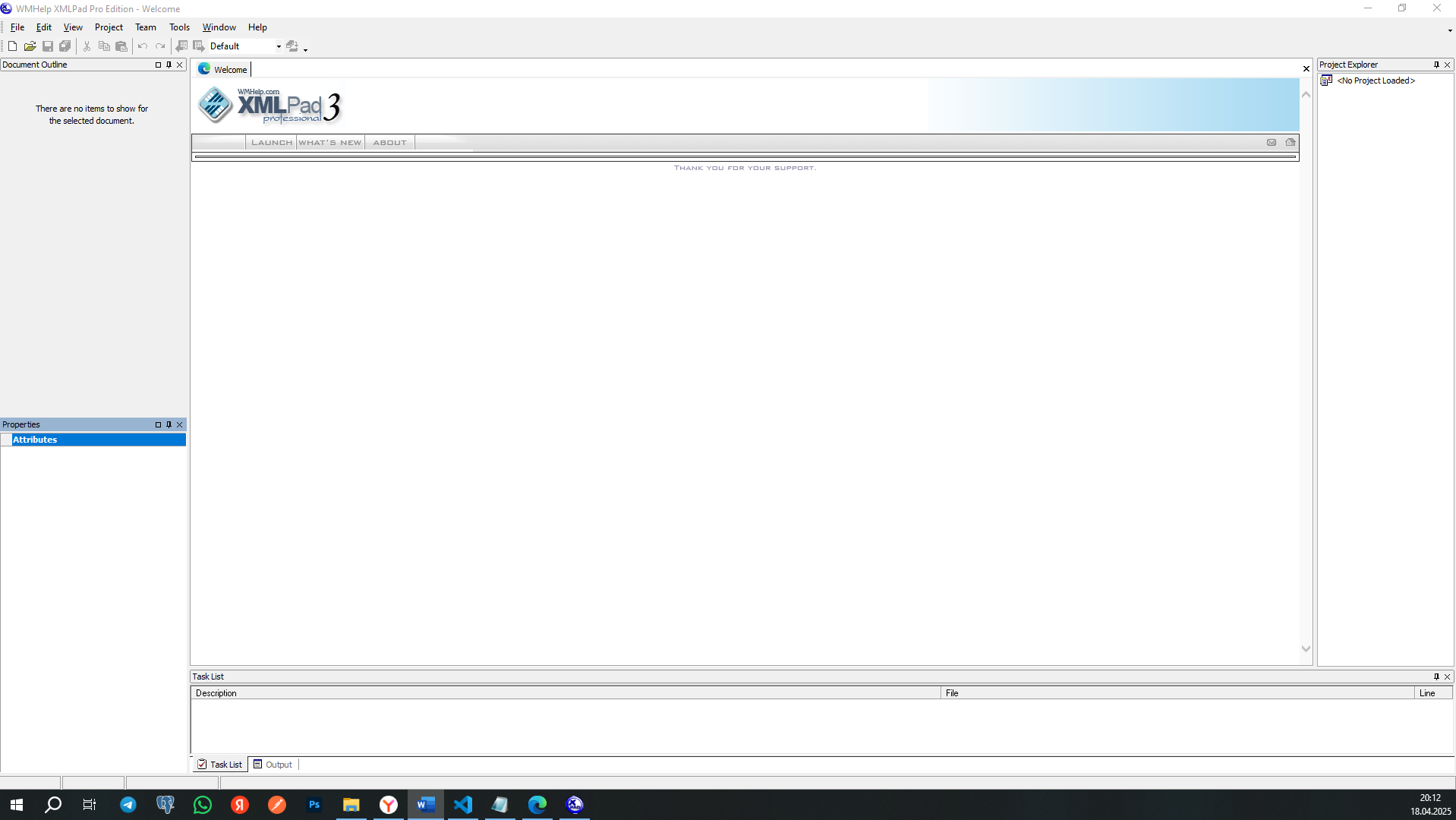


Рис. 3 Редактор XmlPad

Далее создаем новый файл через контекстное меню, выбрав категорию XML (рис.4).

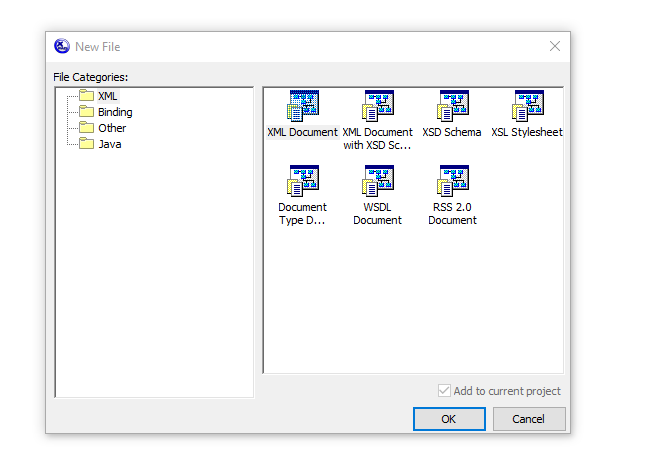


Рис. 4 Создание файла

Затем указываем дополнительные параметры и главный тег <MAIN> (рис.5).

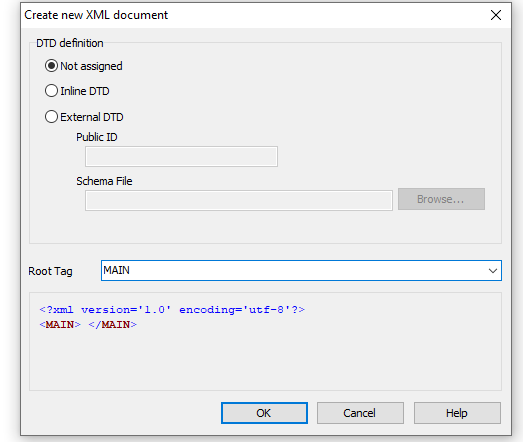


Рис. 5 Доп. параметры

Проверяем созданный документ (рис.6).

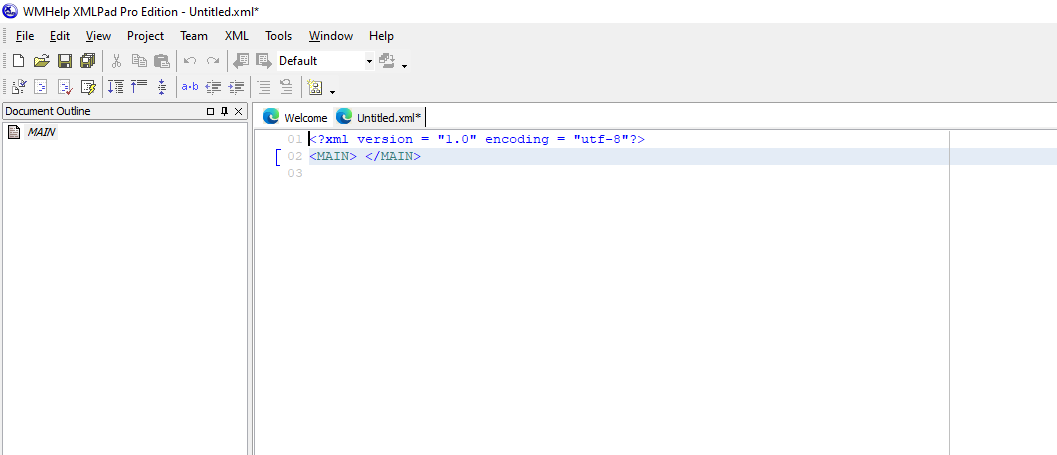


Рис. 6 Проверка документа

Меняем кодировку следуя заданию через пункт меню «Advanced Save Option» (рис.7).

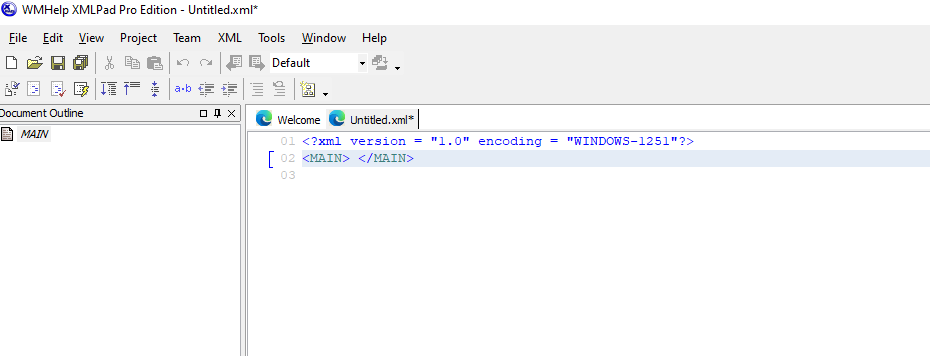


Рис. 7 Кодировка

Проверяем изменения (рис. 8).

В первой строке прописываем атрибут standalone, удаляем тег <MAIN> и добавляем разметку (рис. 9).

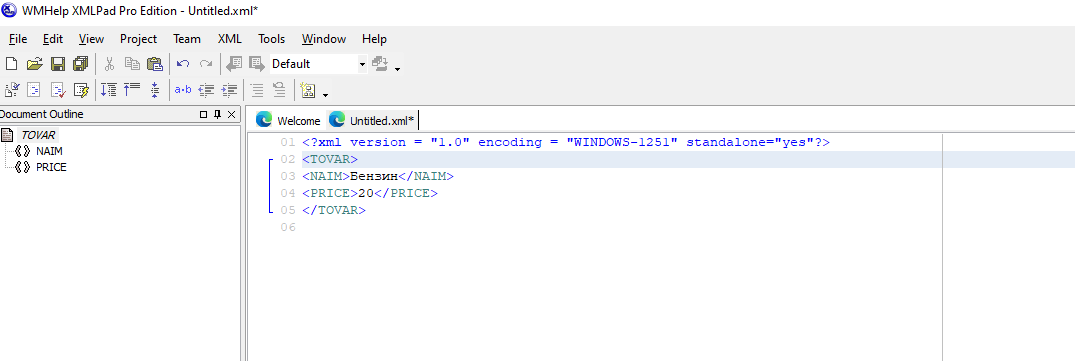


Рис. 9 Добавление атрибута

Выполняем проверку структуры документа с помощью программы XmlPad по алгоритму XML => Validate. Проверяем task list, данная надпись означает верное заполнение документа (рис. 10).

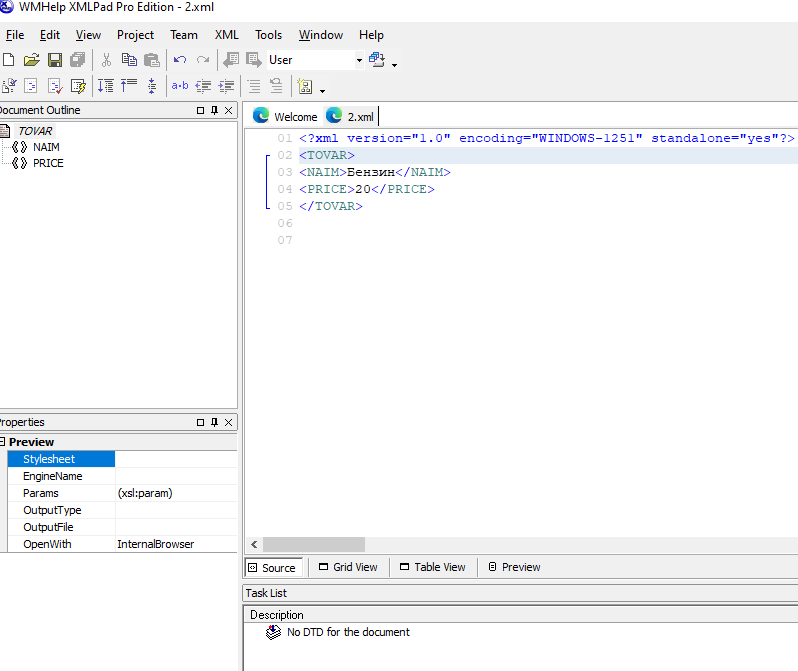


Рис. 10 Проверка

В конце сохраняем документ 2.xml в папку.

## Задание №3

Создаем новый документ и добавляем содержимое из задания (рис. 12).

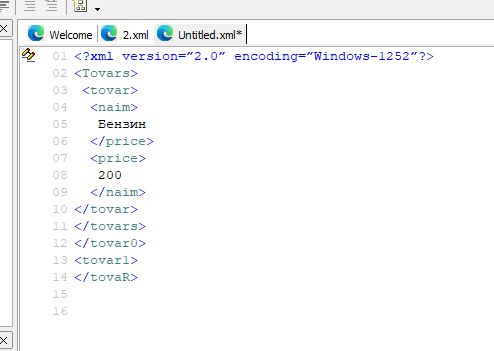


Рис. 12 Содержимое документа

Проанализируем ошибки с помощью программы XmlPad, обращаем внимание на Task List (рис. 13).

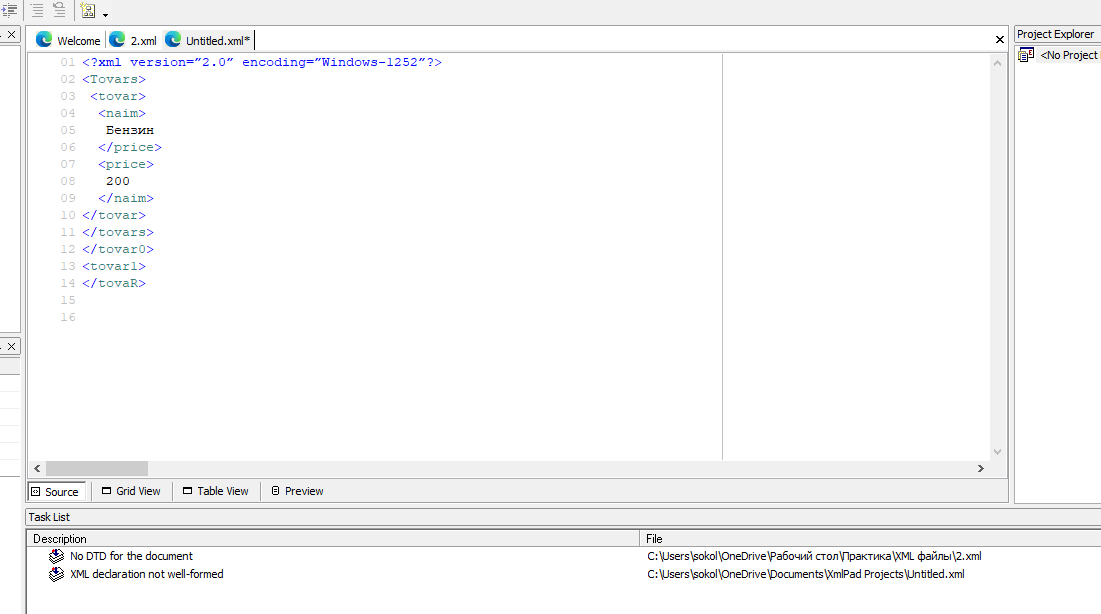


Рис. 13 Валидация с помощью XmlPad

Устраняем стандартную ошибку псевдокавычек и получаем следующую ошибку из-за несоблюдения вложенности (рис. 14).



Рис. 14 Устранение ошибок

Анализируем разметку, анализируем, учитывая правила, что есть основной тег, все теги должны закрываться и совпадать по имени (регистр тоже важен). После устранения ошибок смотрим на результат (рис. 15).

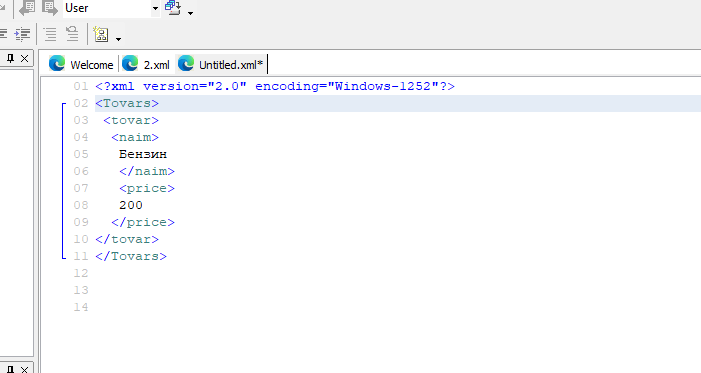


Рис. 15 Результат

Проверим вид документа с помощью вкладки Preview, предварительно изменив версию на 1.0, т.к 2.0 выдает ошибку (рис. 16).

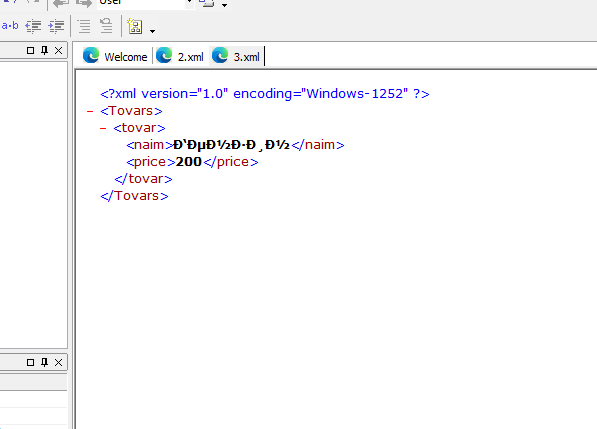


Рис. 16 Неверная кодировка

Такой результат из-за кодировки, меняем на UTF-8, поддерживающая русские символы (рис. 17).

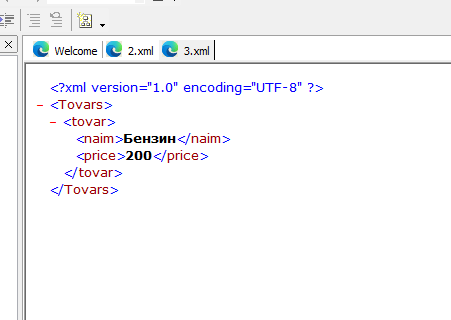


Рис. 17 UTF-8

# Практическая работа №2

## Задание №1

Создадим файл xml, который будет содержать следующие данные (рис. 18):

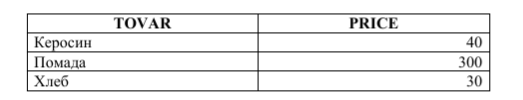


Рис. 18 Пример таблицы

Создаем базовую структуру с тегом TOVARS, меняем кодировку, добавляем тег TOVAR по кол-ву товаров (рис. 19).

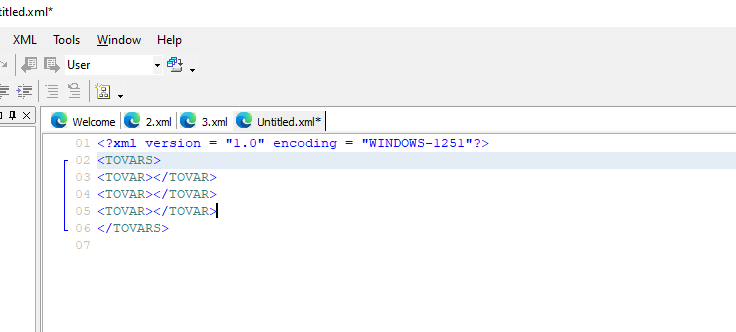


Рис. 19 Базовая структура

Затем добавляем атрибуты NAIM и PRICE в 3 товара (рис. 20).

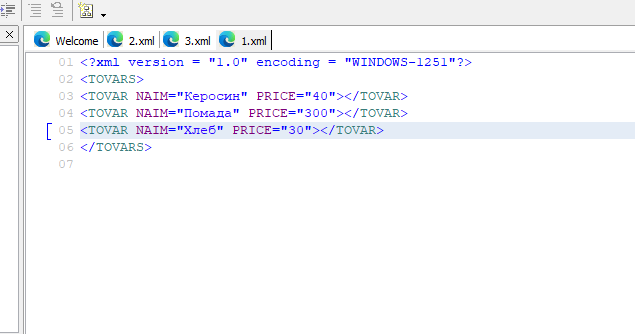


Рис. 20 Добавляем атрибуты

Не забываем провести валидацию, убеждаемся, что ошибки отсутствуют.

## Задание №2

Создаем документ следующей структуры (рис.21).

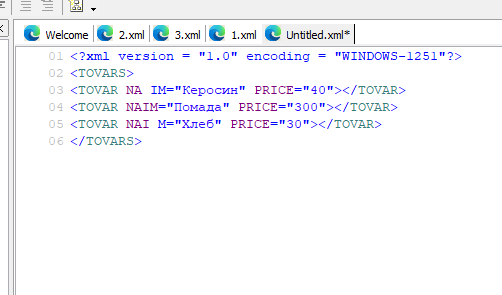


Рис. 21 Структура документа

Меняем атрибут PRICE, на вложенный тег (рис. 22):

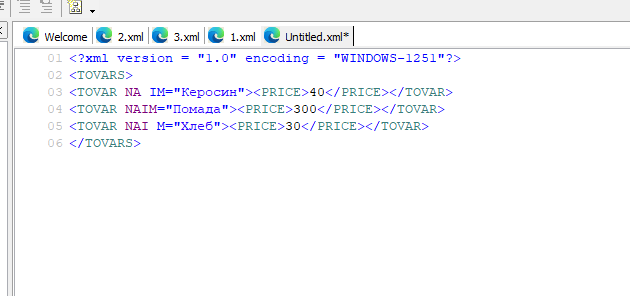


Рис. 22 Меняем атрибуты

Получаем ошибку форматирования, указывающую на 3 строку, корректируем атрибуты NAIM, убираем лишние пробелы (рис.23).

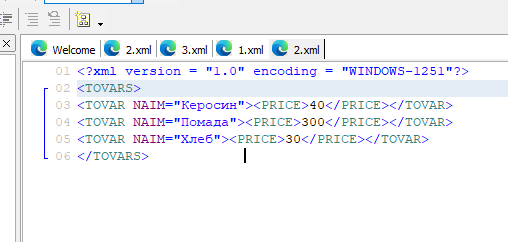


Рис. Валидация

Убеждаемся в отсутствии ошибок (рис. 24).

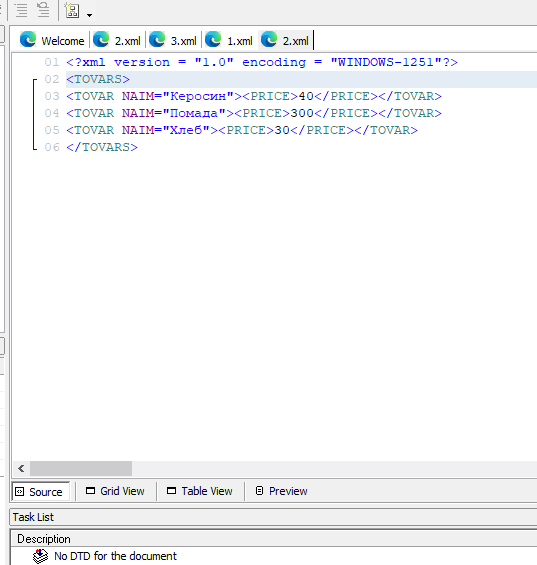


Рис. 24 Проверяем на ошибки

# Практическая работа №3

## Задание №1

Создаем новый документ и структуру из задания - TOVARS и TOVAR– задать в виде элементов, PRICE, NAIM, DESCRIPTION и VALUTA – в виде атрибутов (рис. 25).

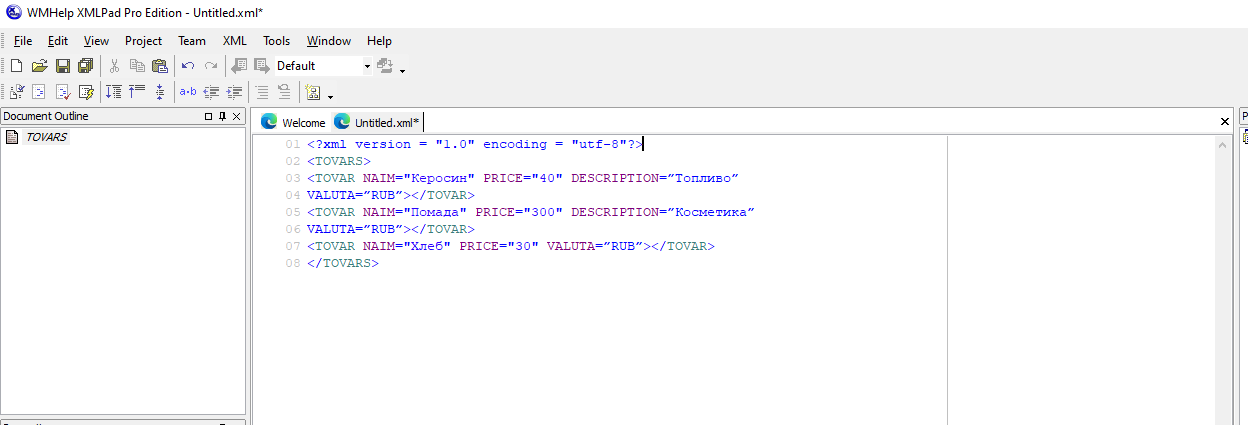


Рис. 25 Задаем атрибуты

Далее создадим схему DTD. DTD (Document Type Definition) — это способ определения структуры XML-документа. С помощью DTD можно задать правила, которым должен соответствовать XML-документ: какие элементы в нем допустимы, в каком порядке они могут встречаться, какие атрибуты допускаются у элементов и т. д.

Для создания схемы DTD в программе XMLPad требуется создать новый документ, с типом Document Type Difinition (рис. 26)

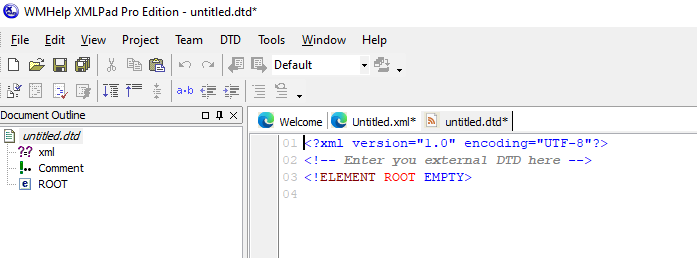


Рис. 26 Создаем DTD

Для корректной работы схемы требуется удалить всю информацию и ввести заново. Ввод информацию в схему начинается с указания корневого элемента и всех его дочерних элементов, при этом получается схема (рис. 27).

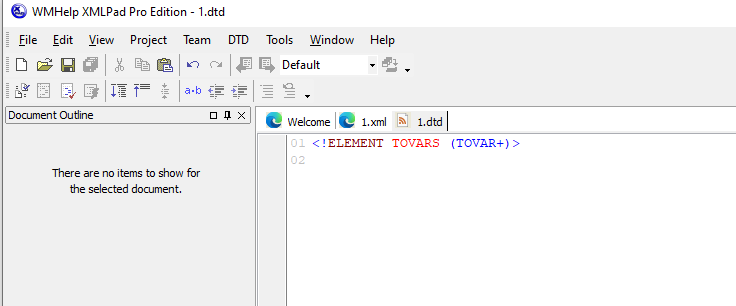


Рис. 27 Создание DTD

Было введено имя корневого тега TOVARS и имя дочернего элемента (TOVAR).

Символ + после имени дочернего элемента означает, что он может повторяться несколько раз и будет включен в документ не менее одного раза.

Затем требуется задать атрибуты NAIM, PRICE, DESCRIPTION, VALUTA

элемента TOVAR, причем атрибут DESCRIPTION являются необязательным, атрибут VALUTA является обязательным и может принимать одно из 3-х значений (RUB, USD, EUR). Вид схемы показан на рис. 28.

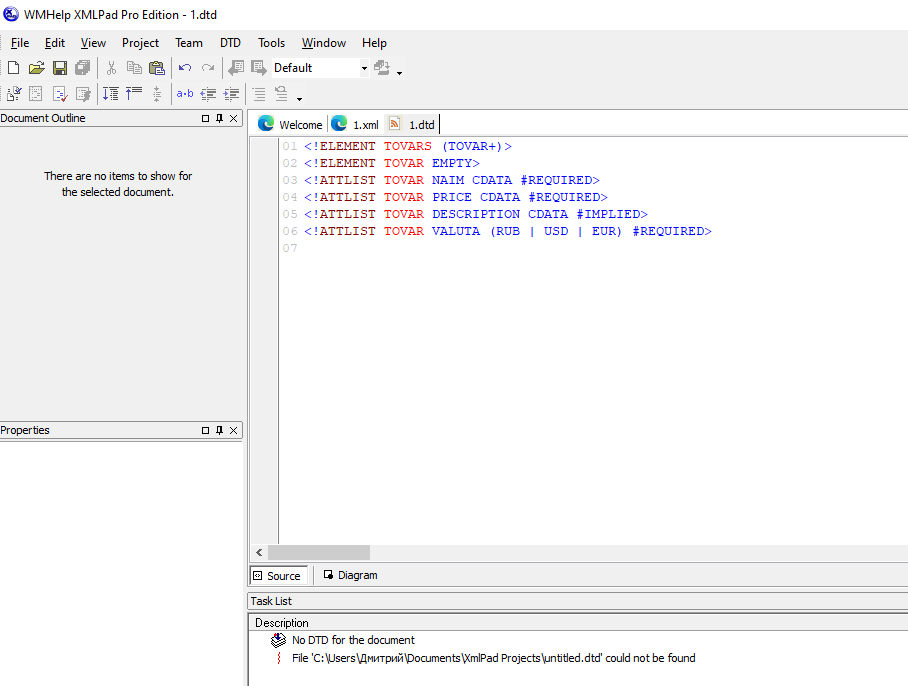


Рис. 28 DTD

Теперь необходимо подключить схему к документу 1.xml, добавляем в заголовке <!DOCTYPE TOVARS SYSTEM "1.dtd"> (рис. 29).

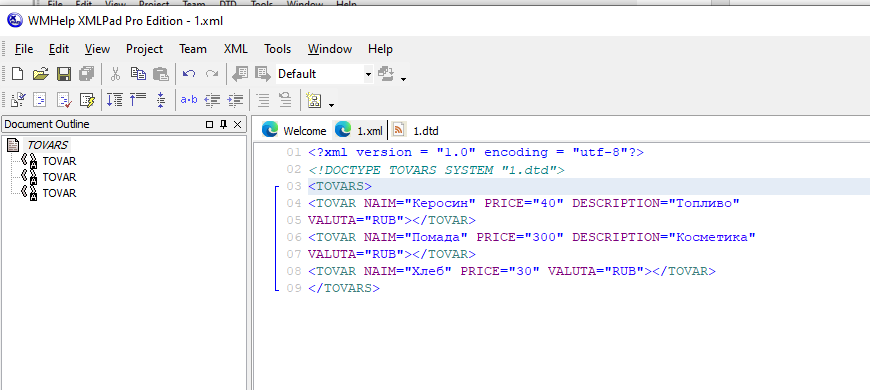


Рис. 29 Подключение схемы

После подключения делаем валидацию и получаем сообщение об отсутствии ошибок т.к разметка файла xml соответствует схеме DTD (рис. 30)

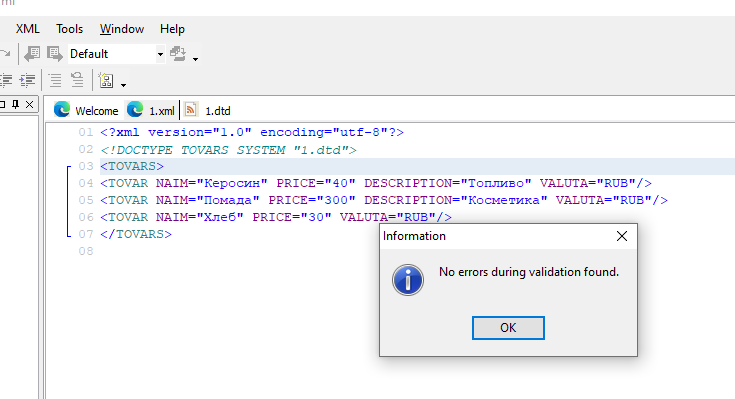


Рис. 30 Валидация

## Задание №2

Проверяем схему на рис. 31 на соответствие схеме на рис.32.

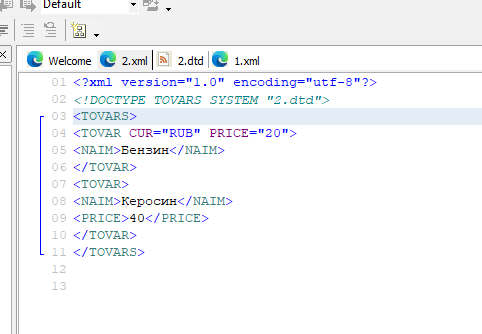


Рис. 31 Проверка xml

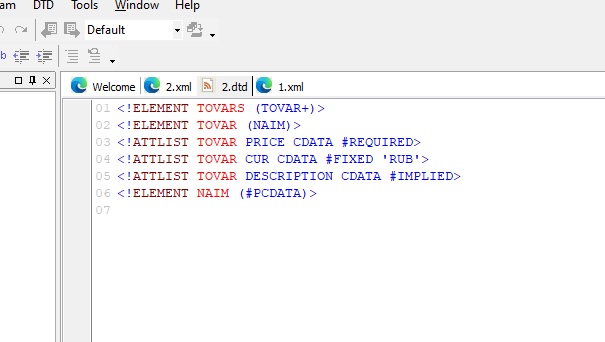


Рис. 32 Схема DTD

Выполнив валидацию, получаем множество ошибок (рис. 33).



Рис. 33 Ошибки

Видим, что требуется атрибут PRICE для товара, а дочерний элемент прайс нам не нужен, т.к. в схеме указан именно атрибут, исправляем, делаем валидацию, просматриваем документ (рис. 34-36).

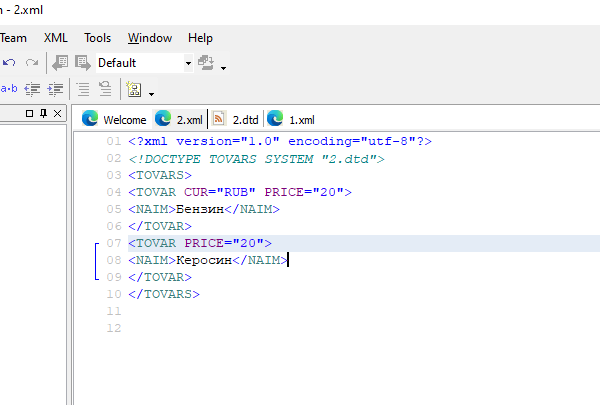


Рис. 34 Исправляем price

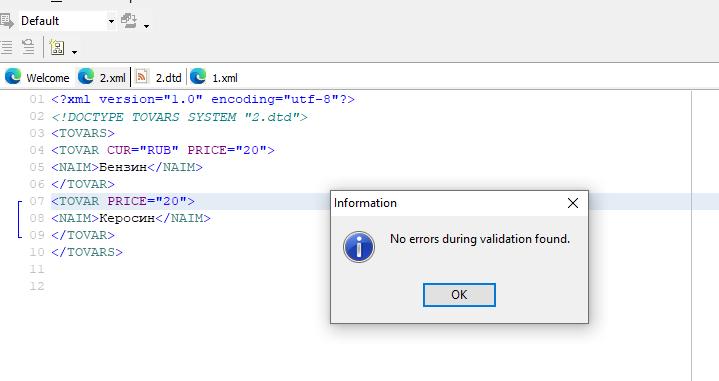


Рис. 35 Валидируем

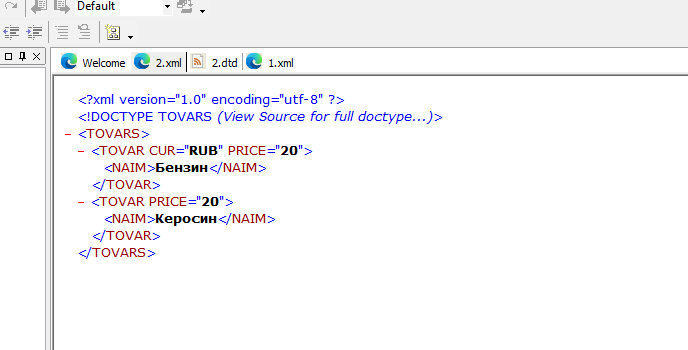


Рис. 36 Предпросмотр

# Практическая работа №4

## Задание №1

Создаем документ с применением пространства имен. В документе будет храниться информация о наличии товара на складе, каждый из товаров характеризуется наименованием (NAIM), ценой (PRICE) и упаковкой. Каждая упаковка характеризуется наименованием (NAIM) и количеством единиц товара в упаковке (KOL\_ED) и количеством упаковок в наличии (KOL\_VO). Также каждая упаковка характеризуется единицами измерения товара (ED\_IZM), в свою очередь единицы измерения товара (ED\_IZM)характеризуются наименованием (NAIM) и дополнительным описанием (DOP\_OPIS).

Дополнительное описание является необязательным элементом.

Создадим документ без пространства имен (рис. 37).

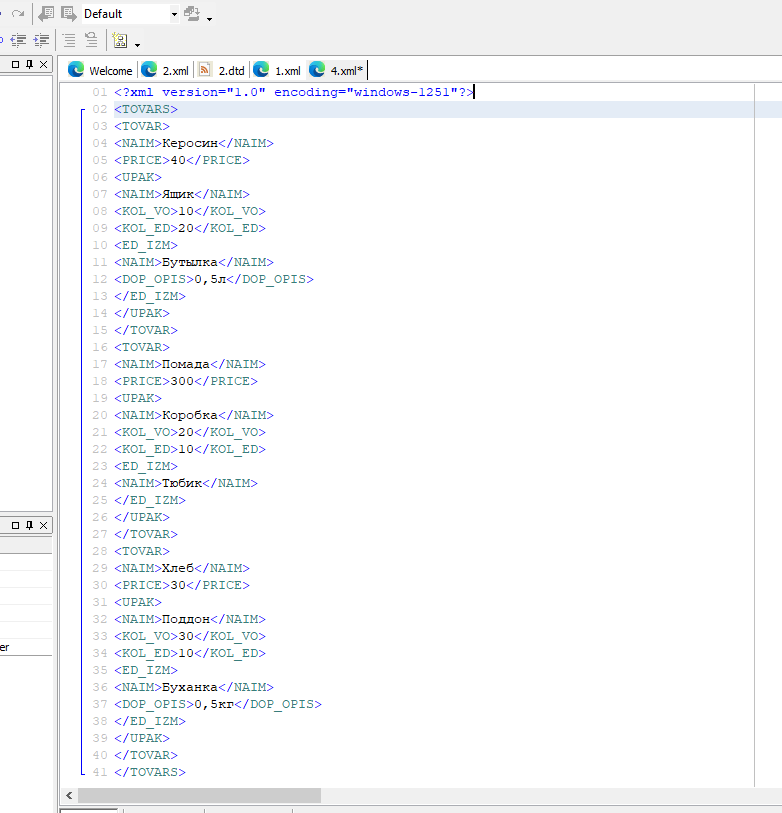


Рис. 37 Документ без пространства имен

Получается много одинаковых элементов, во избежание ошибок, добавим пространство имен, документ примет вид указанный на рисунке 38.

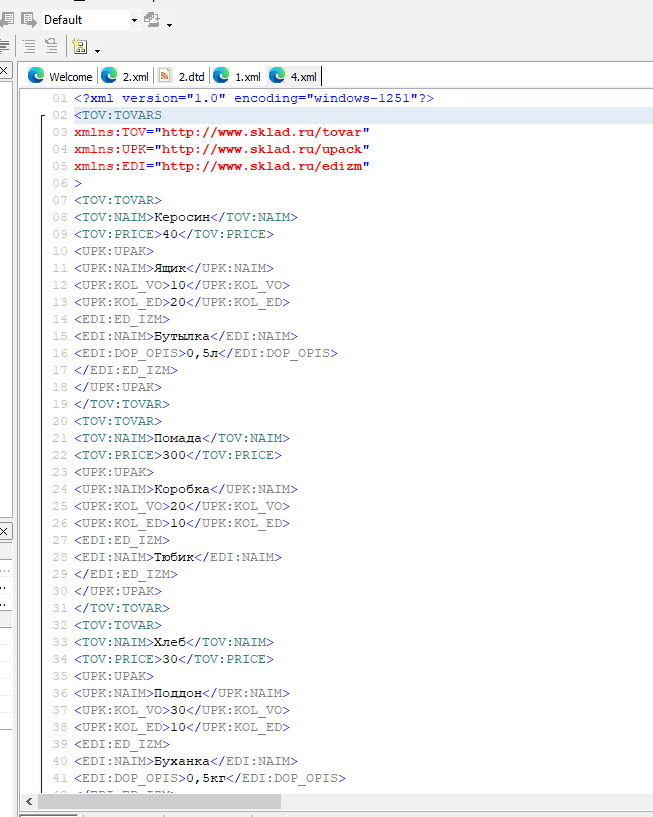


Рис. 38 Добавили пространство имен

Теперь каждый документ относится к своему пространству имен. Просмотрим документ (рис. 39)

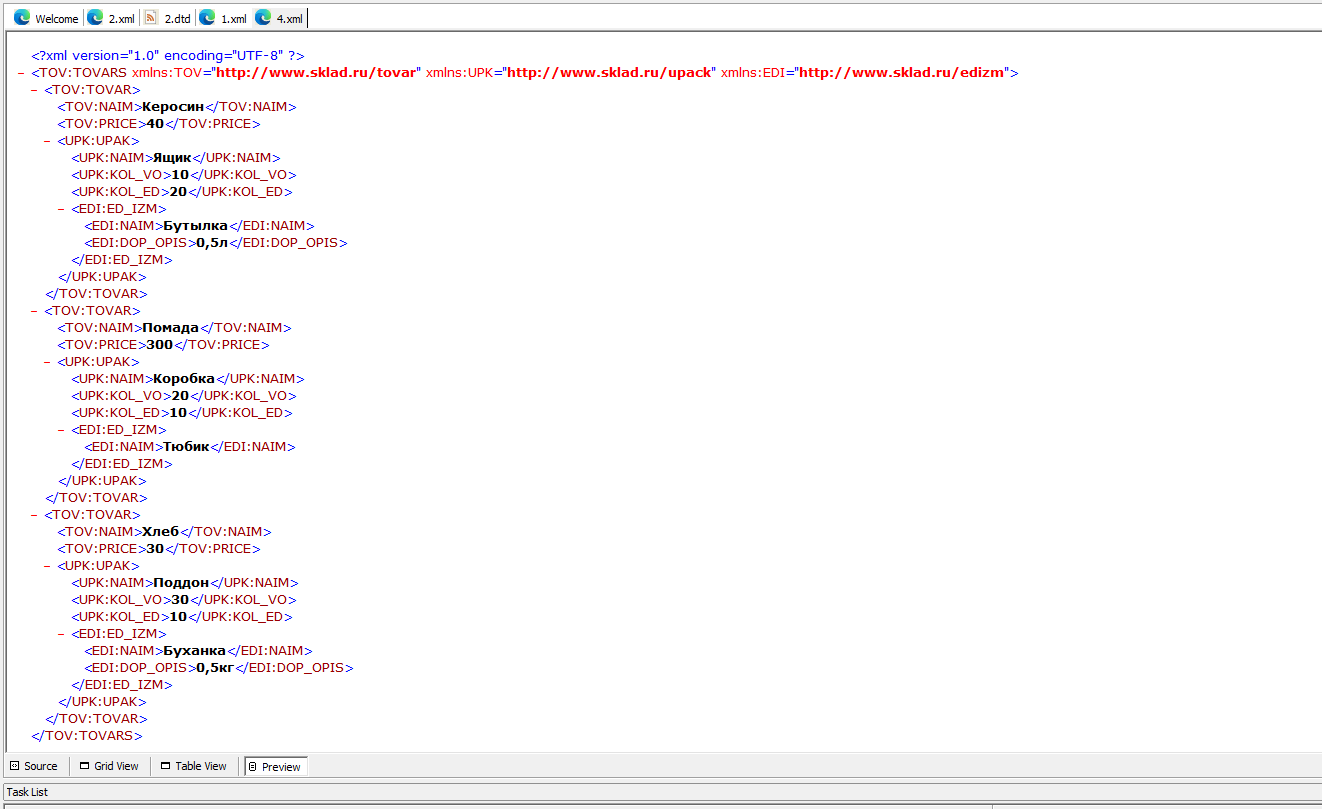


Рис. 39 Preview

# Практическая работа №5

## Задание №1

Требуется создать для опубликования на сайте прайс-листа товаров документ XML на основе схемы XSD. Схема представлена ниже (рис. 40):

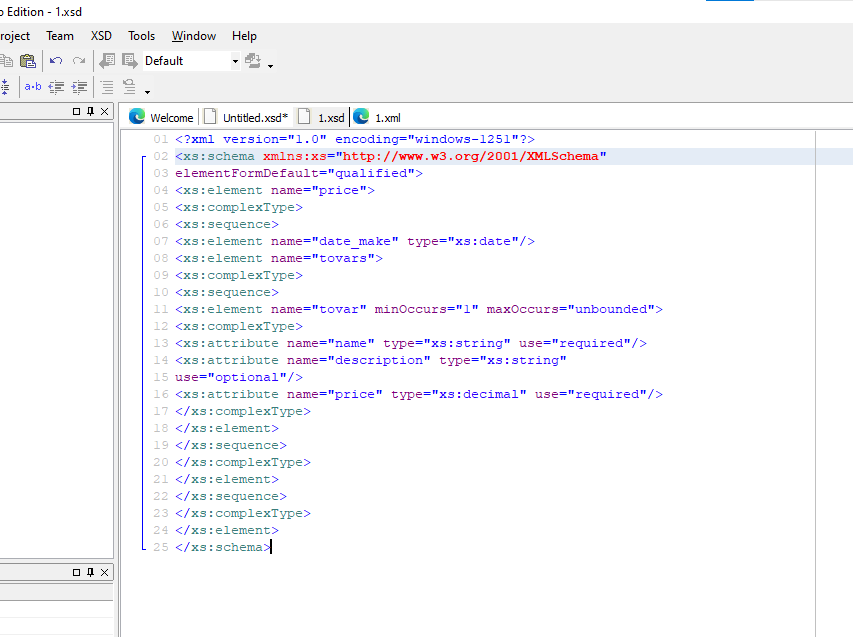


Рис. 40 Схема xsd

XSD (XML Schema Definition) — это язык, с помощью которого описывают структуру, содержимое и правила XML-документа. Проще говоря, это «инструкция», как должен выглядеть правильный XML-файл: какие элементы разрешены, в каком они порядке, какие у них типы данных, какие значения допустимы и т.д.

XSD — это более современная и мощная альтернатива DTD.

На основе данной схемы создаем XML документ (рис. 41):

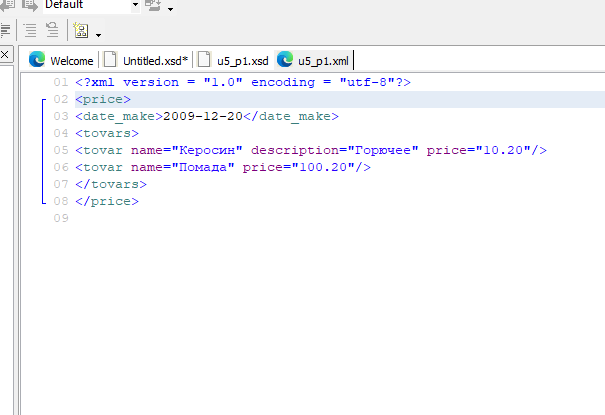


Рис.41 Файл xml

Затем подключаем схему (рис. 42)

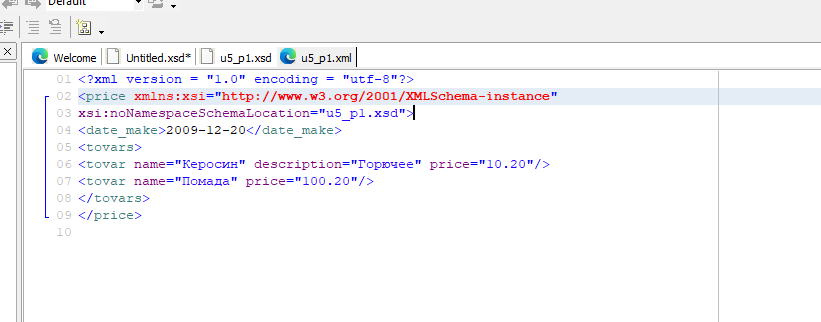


Рис. 42 Подключение схемы

Теперь можно проверить документ xml на ошибки, т.к схема подключена (рис. 43).

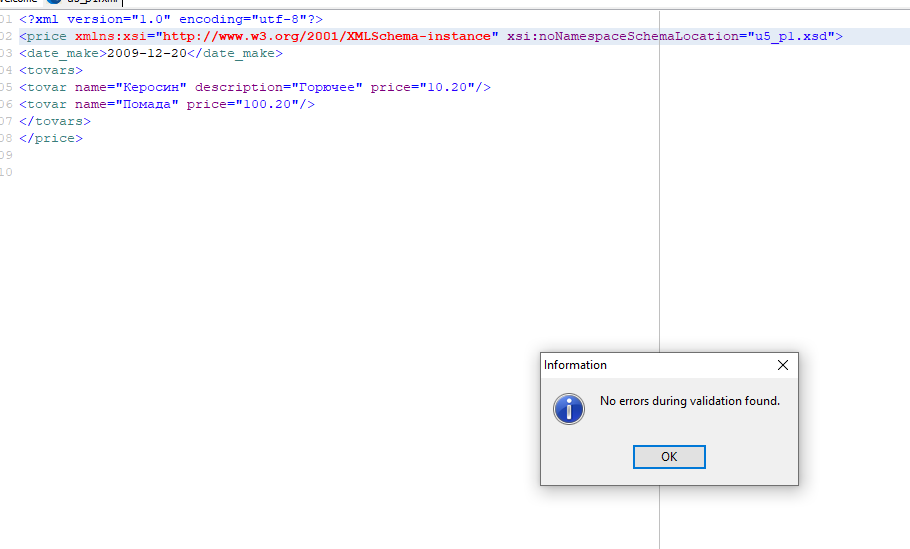


Рис. 43 Проверка ошибок

Ошибки отсутствуют, но для проверки, сознательно внесем неверные значения в PRICE, например добавим букву (рис. 44).

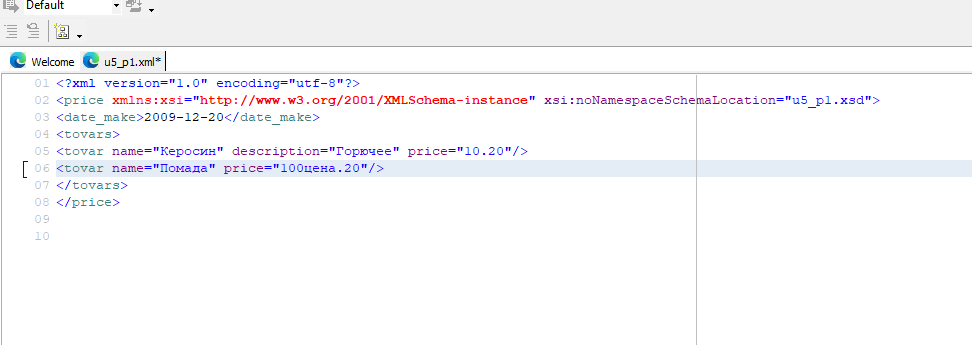


Рис. 44 Создаем ошибки

Схема работает, т.к. получаем ошибку (рис. 45) о несоответствии типов.

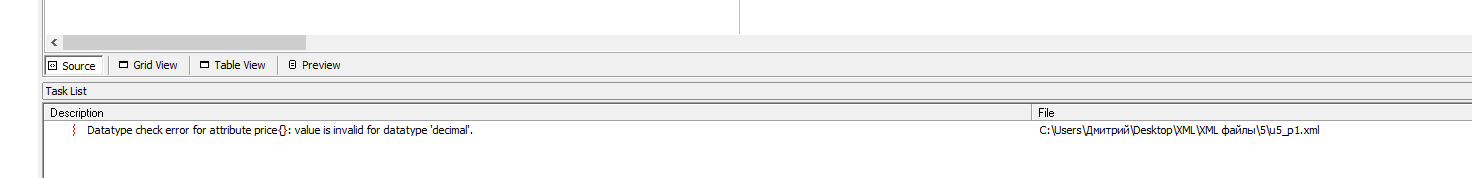


Рис.45 Работоспособность схемы

## Задание №2

Схема XML соответствует схеме задания 1, но файл xml (рис. 46) отличается и после валидации получаем много ошибок, которые необходимо исправить (рис. 47).

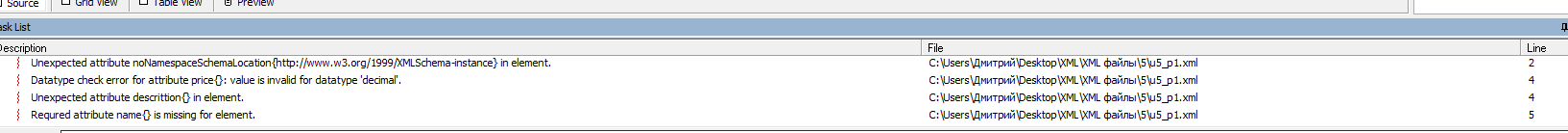


Рис. 47 Ошибки

Первая ошибка расположена на строке 2 – пространство имен указано неверно (должен быть указан 2001, а не 1999). После исправления количество ошибок не уменьшилось. Далее требуется исправить ссылку на пространство имен, правильное название - XMLSchema-instance После исправления ошибки и проверки содержимого, вид документа показан на рис. 48 Видно, что количество ошибок уменьшилось.



Рис. 48 Исправляем ошибки

Одна из ошибок вызвана тем, что на строке 5 в атрибуте price информационное содержимое не соответствует числовому типу (разделитель целой и дробной части указан –, а не точка). После исправления этой ошибки можно исправить имя атрибута descritton на description . После исправления этих ошибок и дополнительной проверки документа количество ошибок изменилось (рис. 49)



Рис. 49 Исправляем ошибки

Следующая ошибка возникла на строке 10 – нет обязательного атрибута с именем name.

Требуется добавить этот атрибут и заново проверить содержимое документа. В результате проверки осталось 2 ошибки на строке 8 (рис. 50).

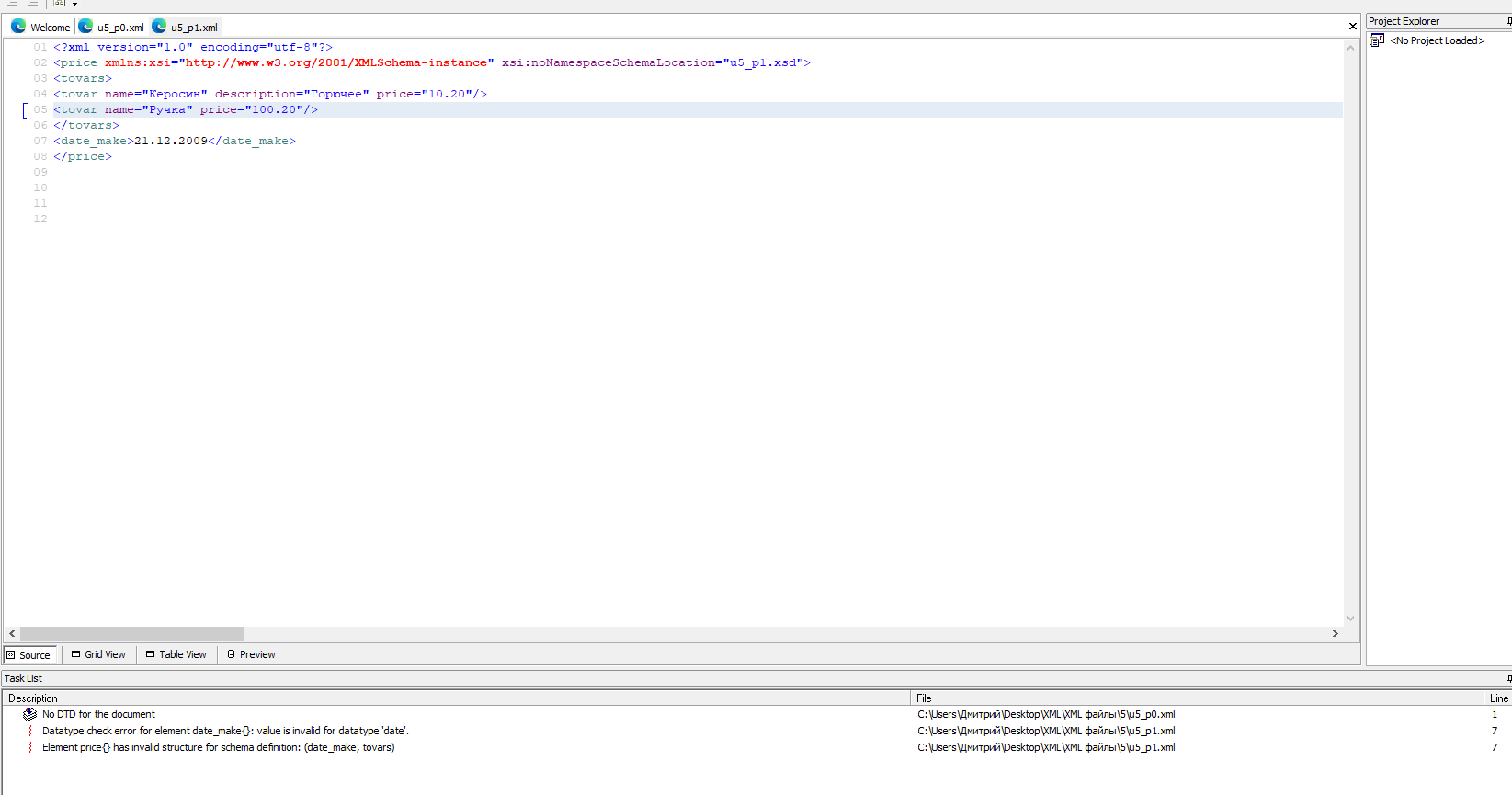


Рис. 50 Исправляем ошибки

Одна из ошибок вызвана неправильным форматом даты – дата представлена в формате дд.мм.гггг в то время как требуется – гггг-мм-дд После изменения формата даты и проверки ошибок осталась одна ошибка (рис. 51)

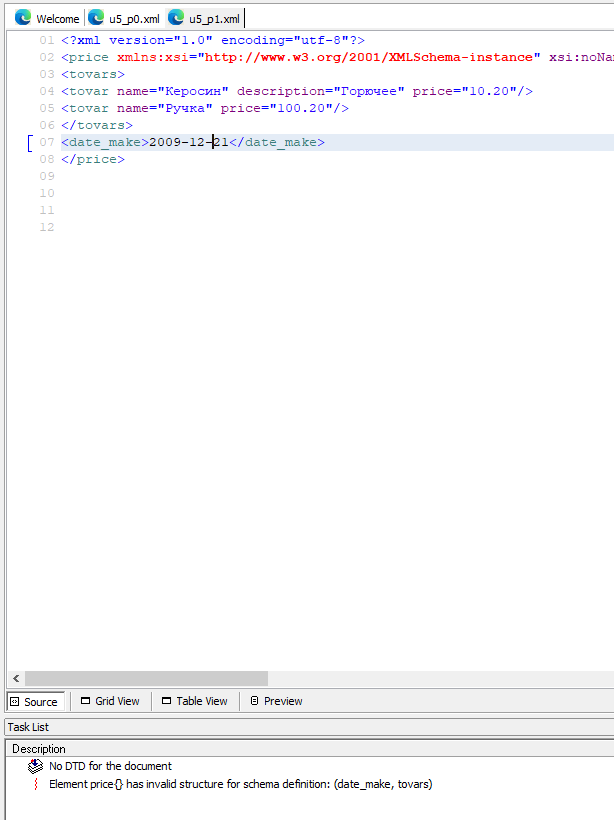


Рис. 51 Исправляем ошибки

Ошибка вызвана тем, что элемент date\_make должен быть указан «выше» по тексту документа, чем элемент tovars. После изменения положения данного элемента документ соответствует схеме (рис. 52)

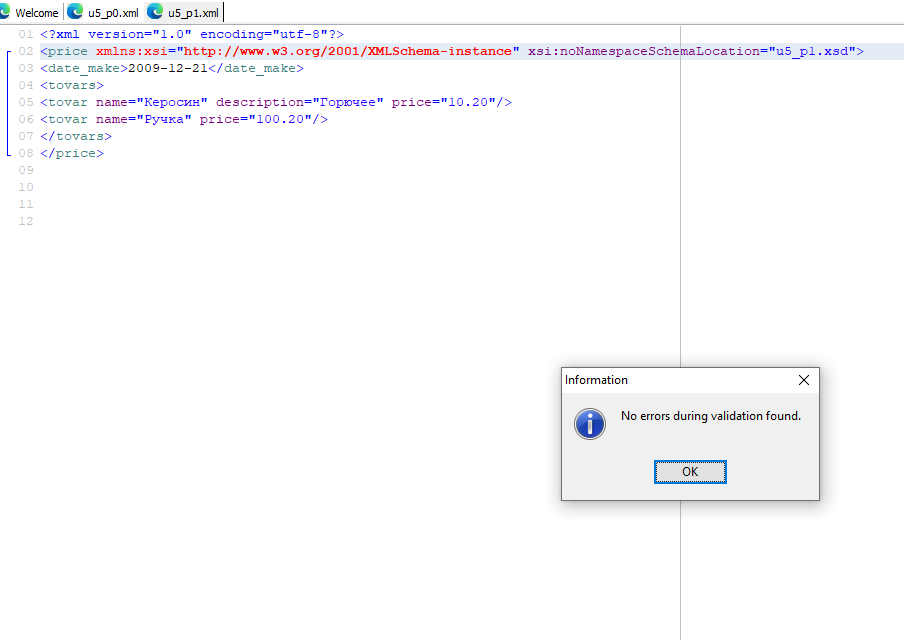


Рис. 52 Ошибки отсутствуют

# Практическая работа №6

## Задание №1

Требуется найти все элементы с именем naim в документе 1 (рис. 53):

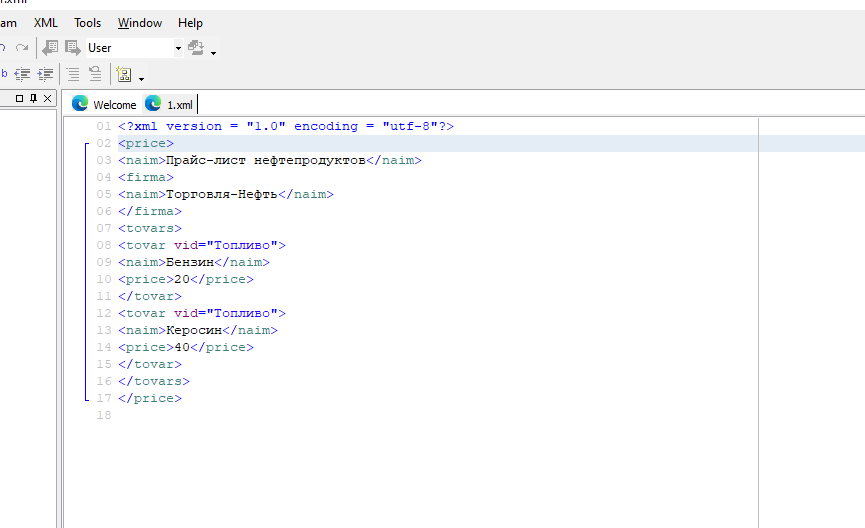


Рис. 53 Документ xml

Для поиска в документе с применением XPATH выражения требуется выполнить пункт Evaluate XPATH меню XML. При этом откроется окно для ввода XPATH выражения, показанное на рис. 54

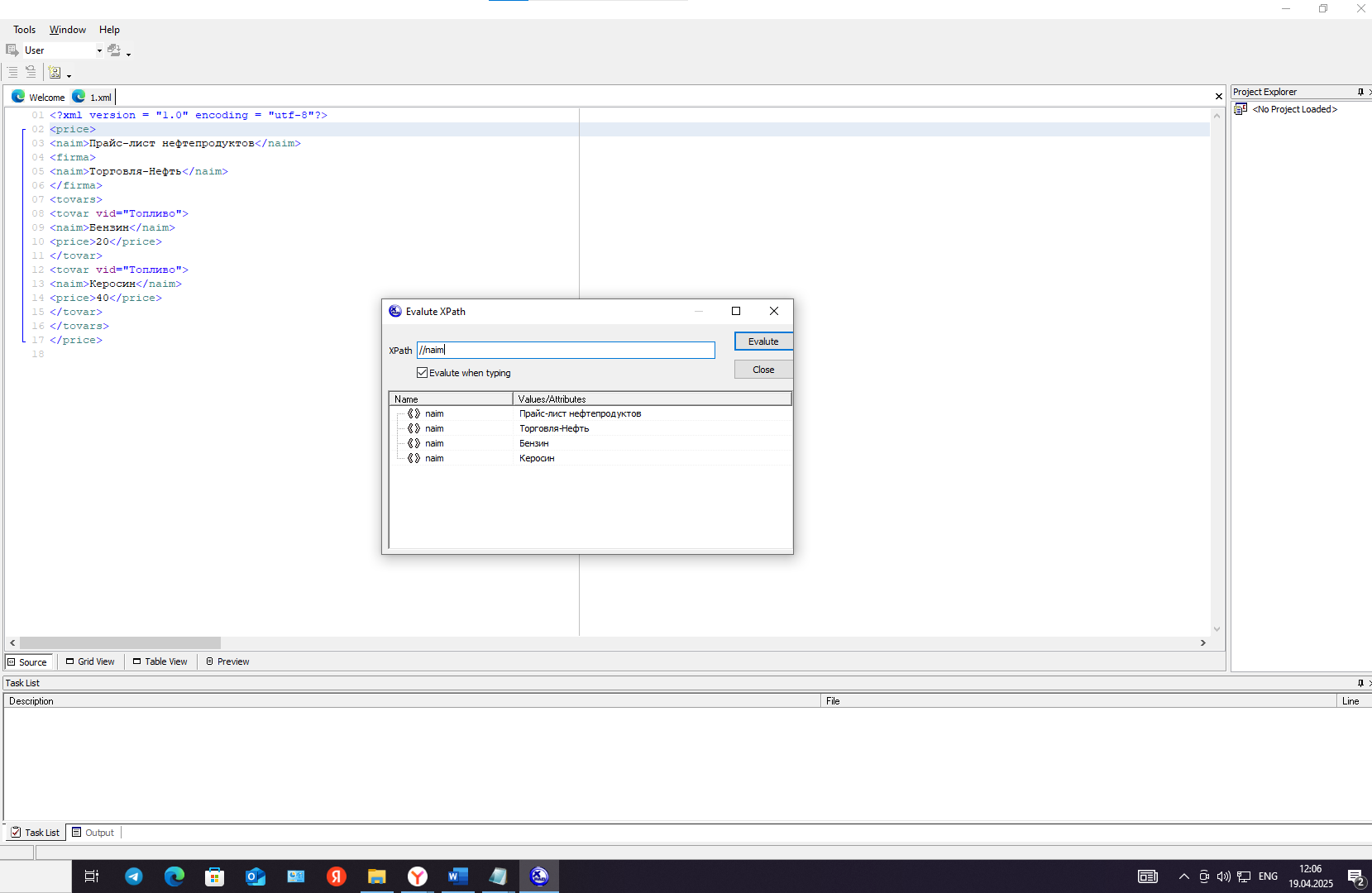


Рис. 54 Поиск

Полученный результат совпадает с ожидаемым, из чего можно сделать вывод о правильности решения задачи.

## Задание №2

Для Документа 1 из задания 1 требуется найти информационное содержание всех элементов naim.

Начало выражения соответствует выражению из задания 1.Выражение принимает следующий вид: //naim/text() (рис. 55)

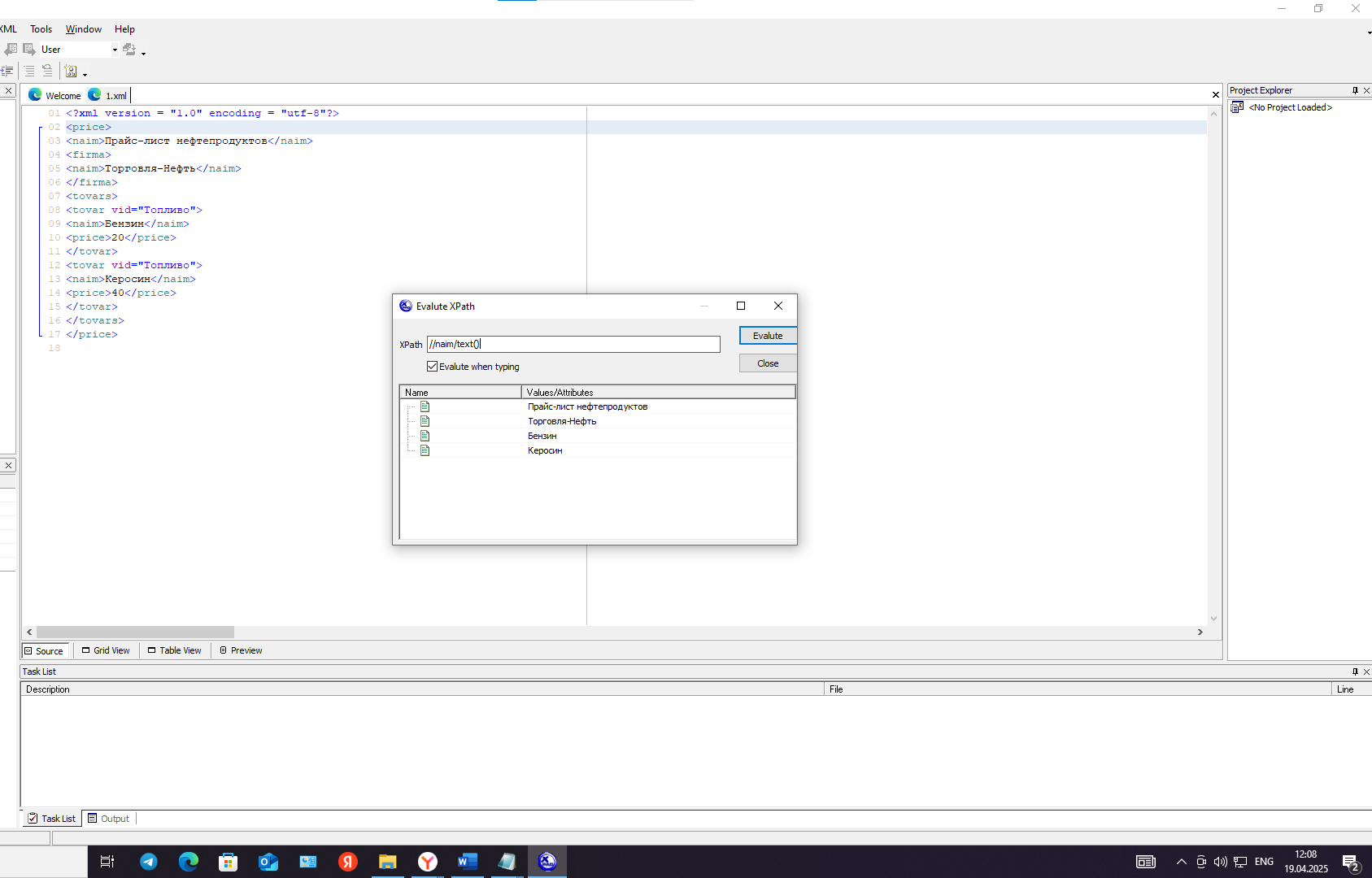


Рис. 55 Поиск

## Задание №3

Найти все элементы с именем tovar и положением с номером 2.

Составление выражения – для нахождения элементов tovar используется следующее выражение: //tovar Для нахождения элемента с заданным положением используется номер в квадратных скобках. Следовательно, полное выражение - //tovar[2]

Все товары (рис. 56):

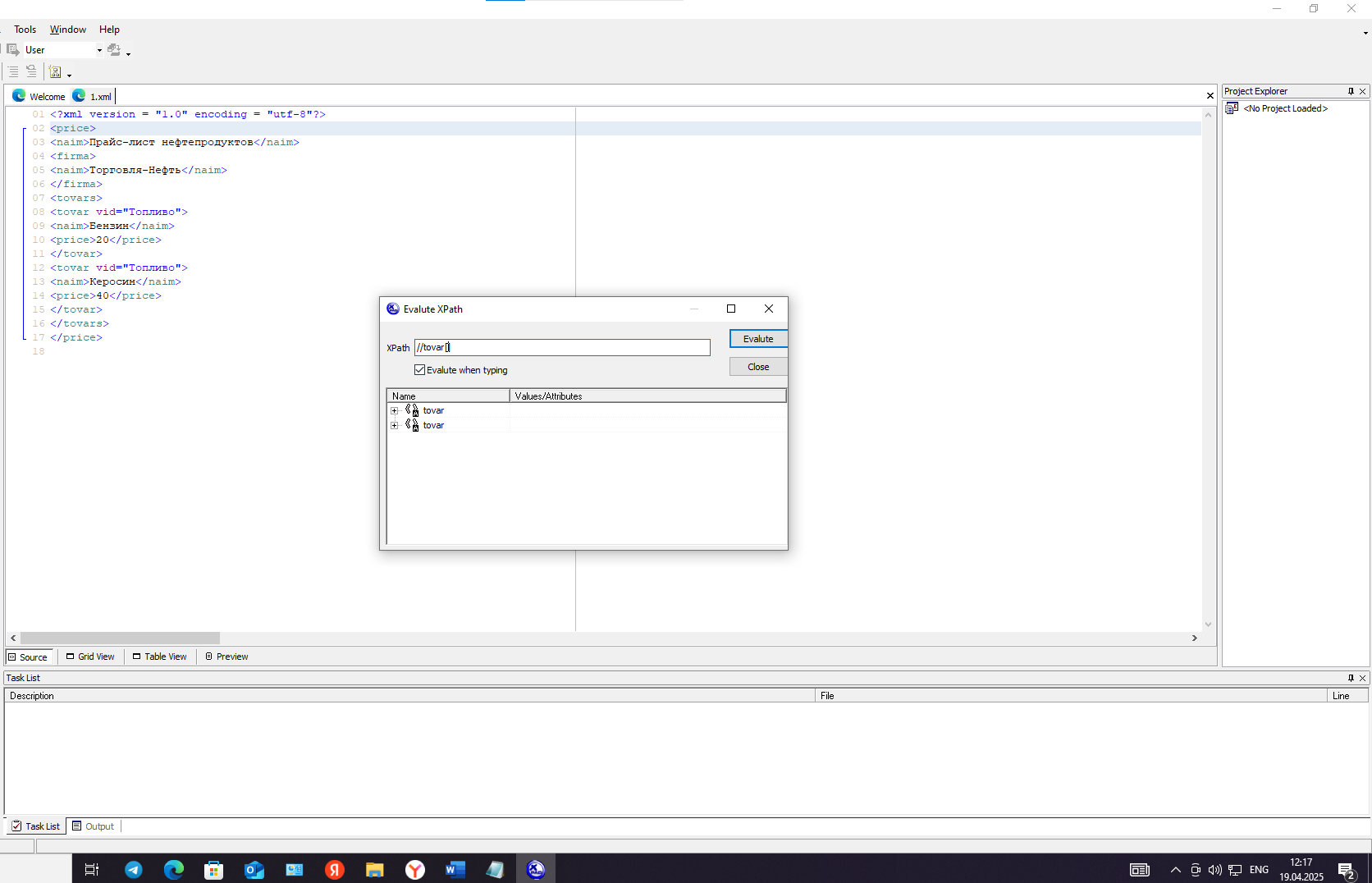


Рис. 56 Поиск

Именно второй товар (рис. 57):

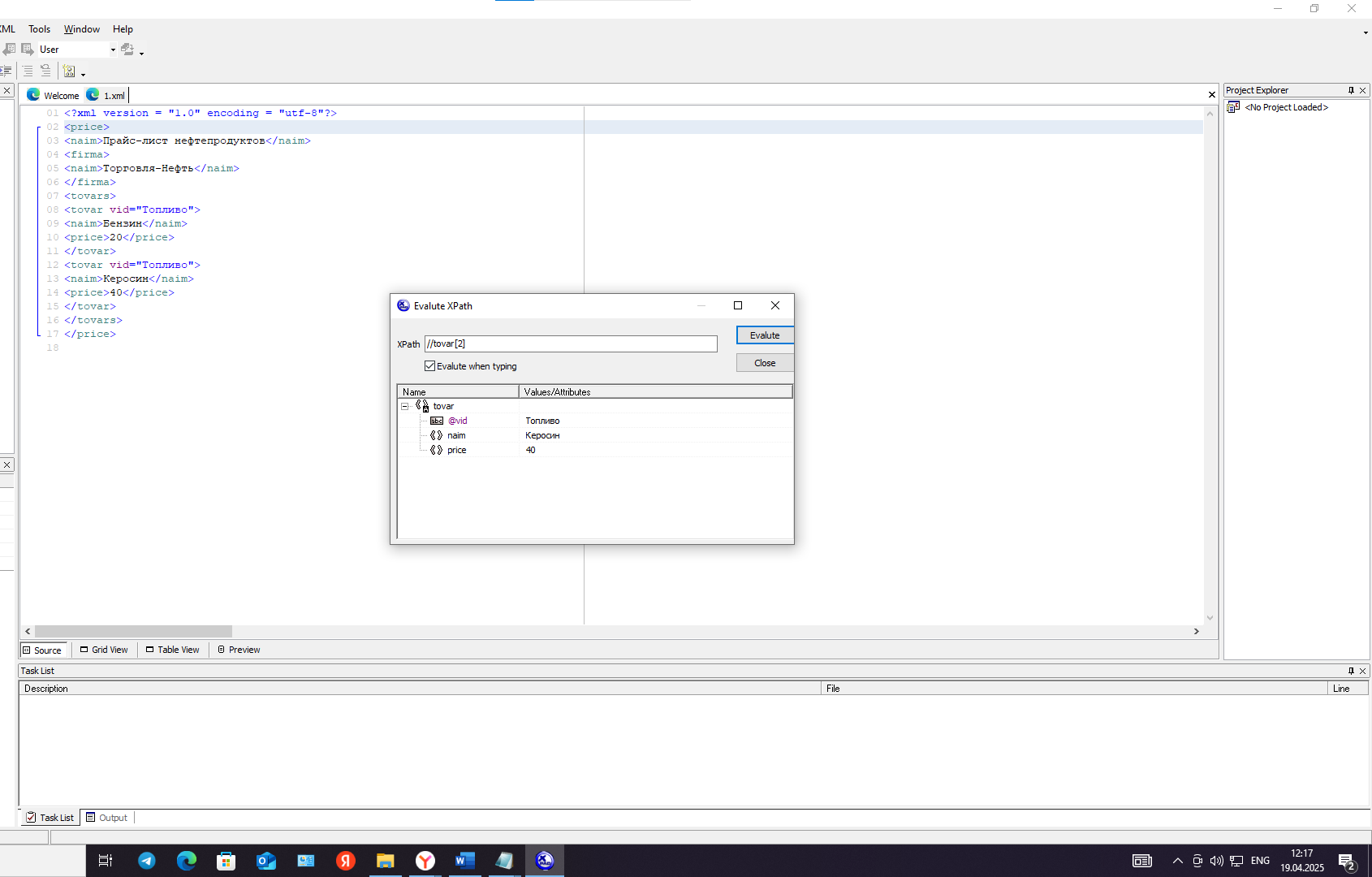


Рис. 57 Поиск

## Задание №4

Найти атрибуты vid для всех элементов. Для этого составим выражение //@vid.

Результат поиска приведен на рисунке 58.

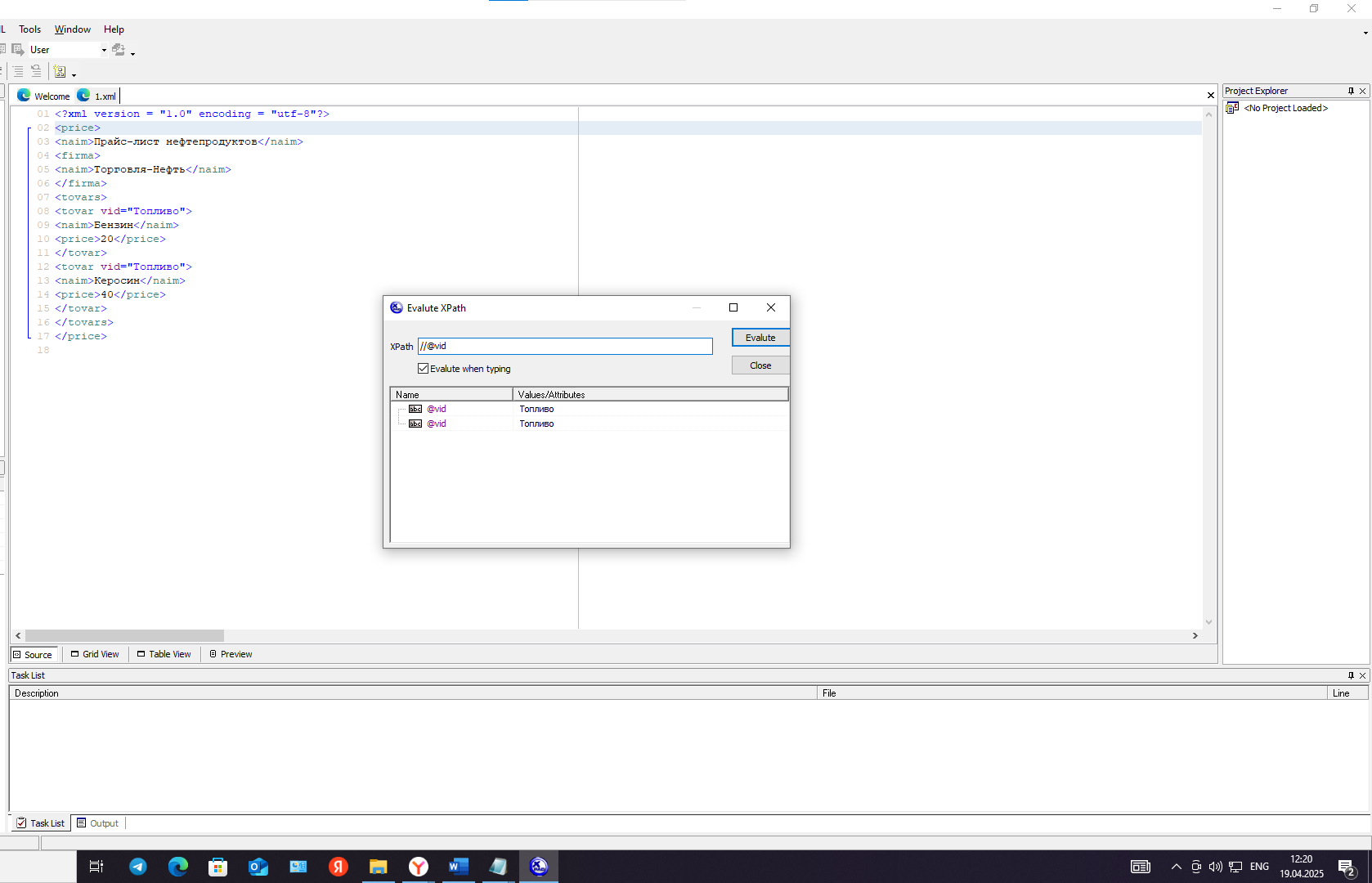


Рис. 58 Поиск

# Практическая работа №7

## Задание №1

Преобразование XML-документа в HTML-документ с использованием XSL-файла.

Был создан документ следующего содержания (рис. 59):

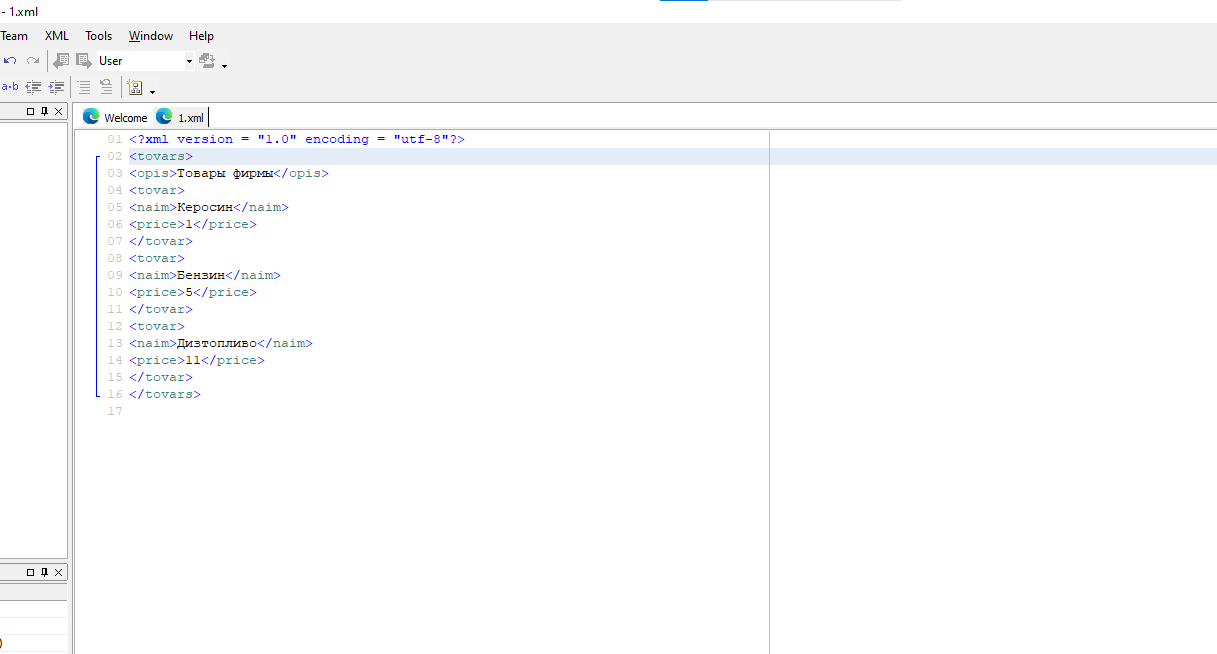


Рис. 59 Документ XML

Для создания html документа следующего содержания (рис.60) необходимо продумать все html элементы и создать XSL файл (рис. 61).

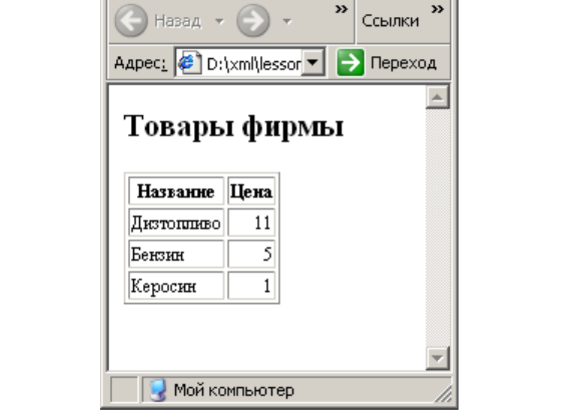


Рис. 60 HTML пример

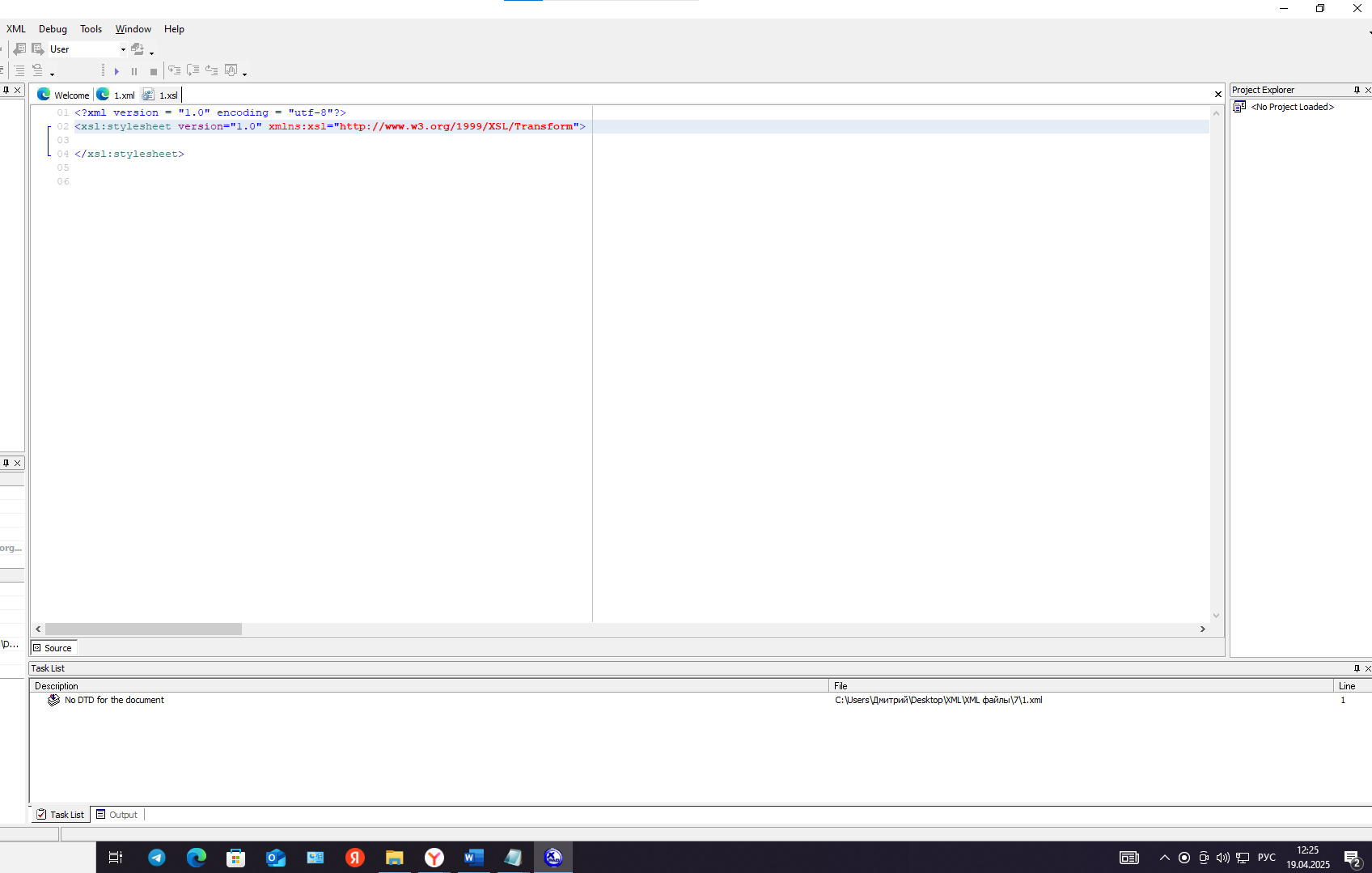


Рис. 61 Файл XSL

Затем подключим файл XSL в XML (рис. 62)

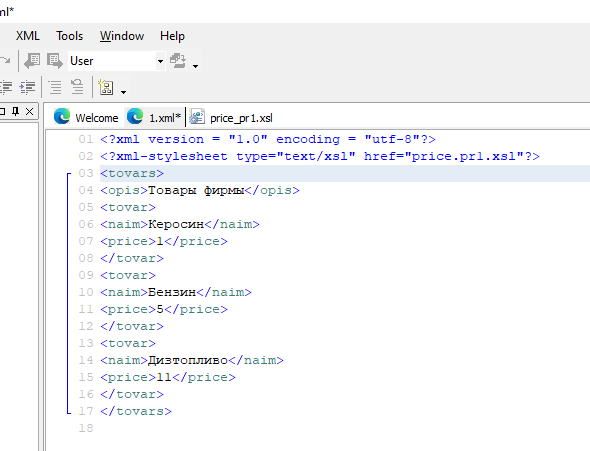


Рис. 62 Подключение

Затем потребуется добавить правило соответствия для корневого элемента XPath - <xsl:template match=”/”>. Элемент должен иметь начальный и конечный теги. Вид документа показан на рис. 63



Рис. Файл XSL

На следующем шаге требуется добавить правило соответствия для элемента opis, который является дочерним элементом для элемента tovars. Также потребуется указать теги заголовка первого уровня <H1> и обозначить применение правил соответствия <xsl:apply-templates>. Вид полученного XSL-файла показан на рис. 64

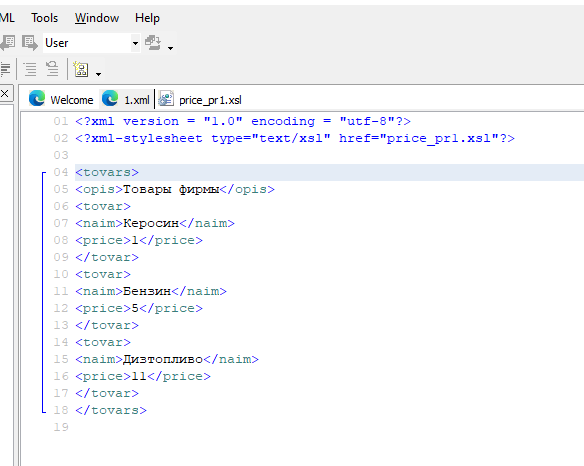


Рис. 64 Структура XML

Проверим работоспособность в файле Preview (рис.65).

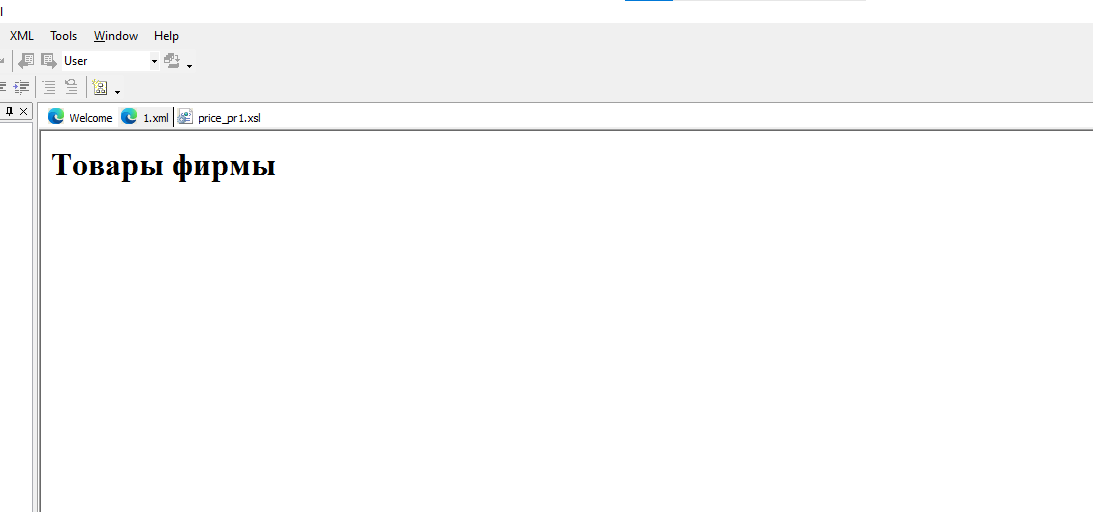


Рис. 65 Просмотр

Следующим этапом будет формирование таблицы. Для этого потребуется указать правило, которое будет отбирать элементы tovar и выводить информационное содержимое подчиненных элементов naim и price. Это правило будет выглядеть следующим образом (рис. 66):

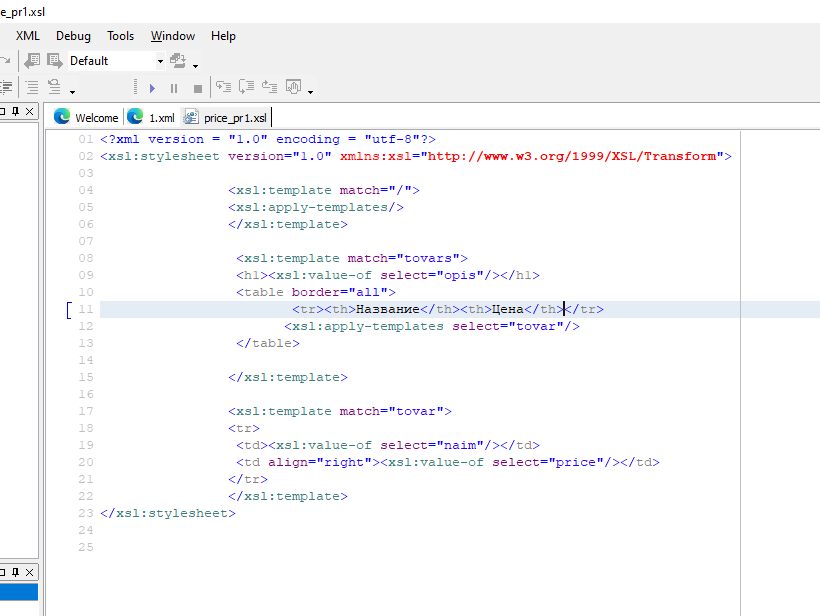


Рис. 66 Добавляем правило

Делаем проверку (рис. 67):

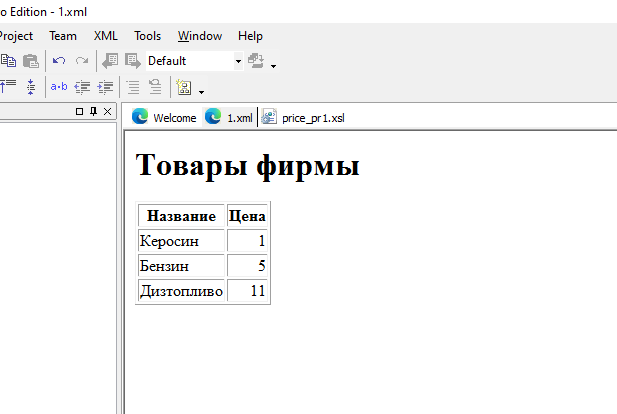


Рис. 67 Просмотр

Осталось сделать сортировку товаров в таблице, поэтому применим правило (рис. 68):

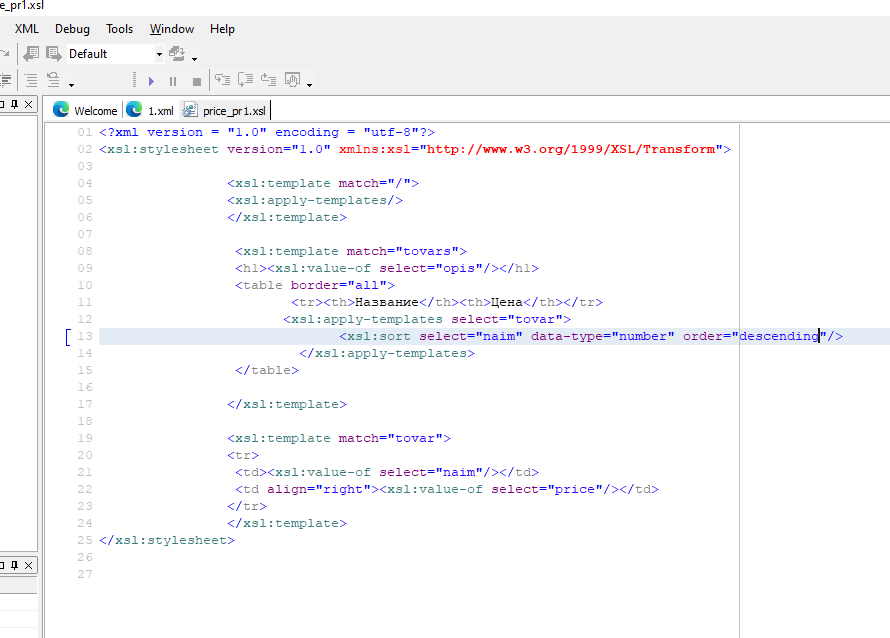


Рис. 68 Сортировка товаров

Делаем проверку в браузере (рис. 69):

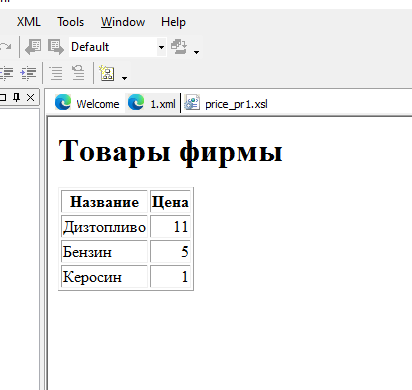


Рис. 69 Просмотр

# Практическая работа №8

## Задание №1

Информация по товарам фирмы экспортирована из программы 1С в виде файла Excel.

Требуется сформировать XML документ со структурой, показанной на рис. 70

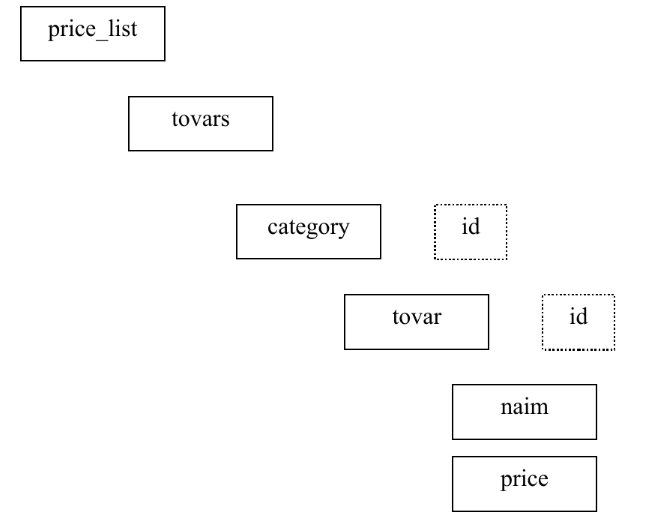


Рис. 70 Схема

Вид файла exel (рис.71)

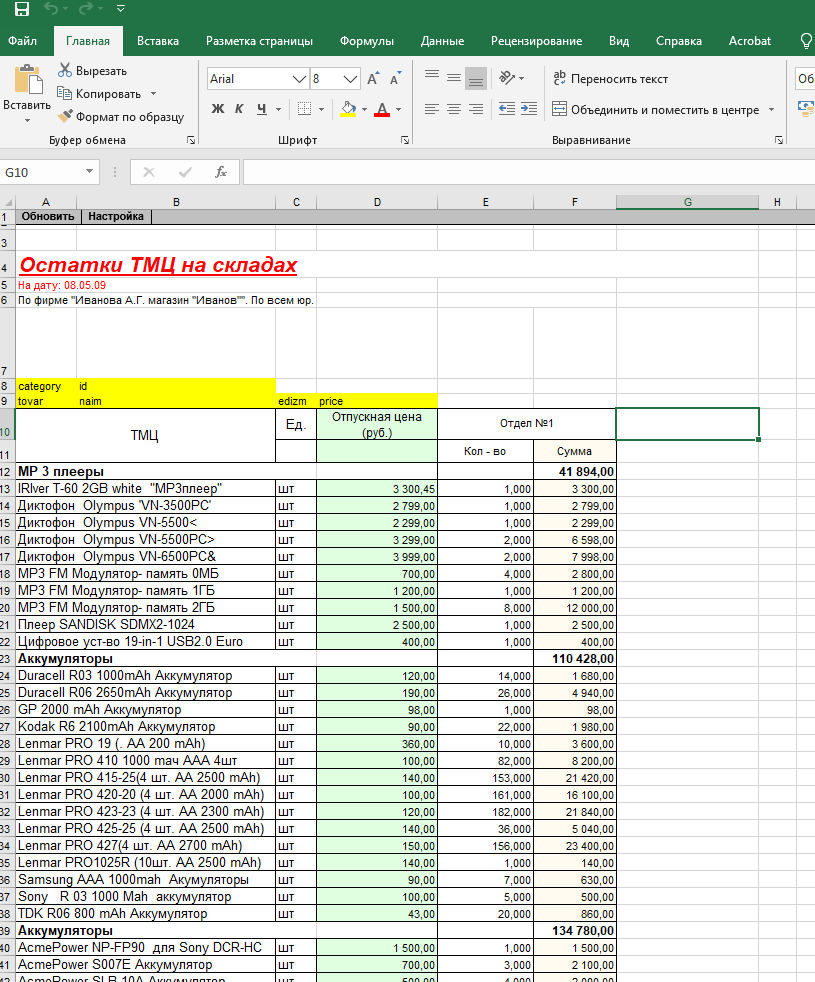


Рис. 71 Таблица товаров

Из всей информации в указанной таблице, для формирования XML документа потребуется наименование товаров (колонка A), единица измерения (колонка C), отпускная цена (колонка D). Наименование категории товара расположено также в колонке A, поэтому

потребуется определить, какая именно информация расположена в колонке A – наименование товара или категория. Для этого можно проверить колонку D - если она заполнена, то в строке находится информация о товаре, если нет – то о категории. При этом можно сделать предположение, что цена не может быть пустой.

Формирование текста документа можно начать с заполнения имен элементов. Это можно сделать в любых незаполненных ячейках. Например, в ячейках A8, A9, B8, B9, C9, D9. Для наглядности эти ячейки «залиты» желтым цветом. Вид таблицы показан на рис. 72

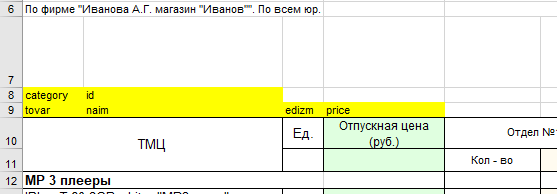


Рис. 72 Ключевые ячейки

Далее создадим формулу для заполнения ячеек соответствующим тегом. Формула имеет следящий вид: =ЕСЛИ(ЕПУСТО(D12); "</" & $A$8 & "><" & $A$8 & ">";"<" & $A$9 & "></" & $A$9 & ">") (используем абсолютную адресацию).

Проверяем формулу (рис. 73):

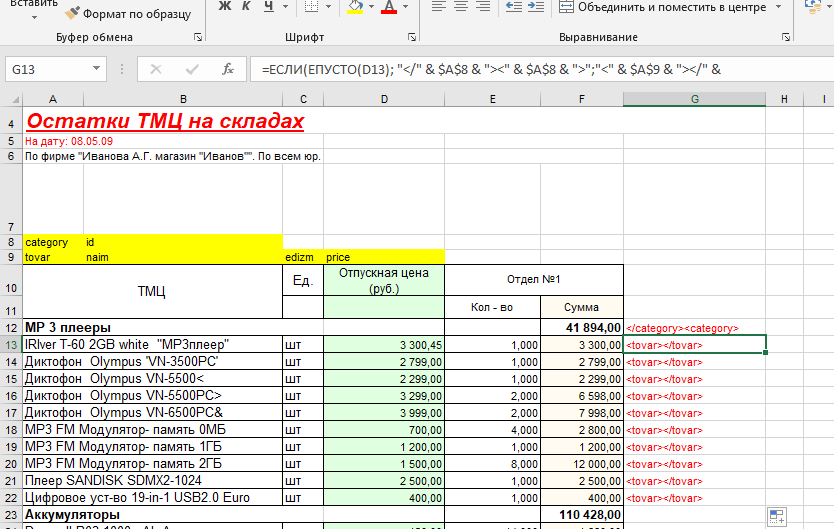


Рис. 73 Формула

Следующим шагом потребуется создать подчиненные элементы для элемента tovar. Можно воспользоваться теми же приемами – наименования элементов брать из ячеек B9, C9, D9 c абсолютной адресацией, а информационное содержание – из соответствующей строки таблицы (адресация относительная). Формула примет следующий вид: =ЕСЛИ(ЕПУСТО(D12);"</"&$A$8&"><"&$A$8&">";"<"&$A$9&">" & "<" & $B$9& ">" & A12 & "</"&$B$9&">" & "<" & $C$9 & ">" & C12 & "</"& $C$9 &">" & "<" &$D$9 & ">" & D12 & "</"&$D$9&">"&"</"&$A$9&">")

Результат работы формулы (рис.74):

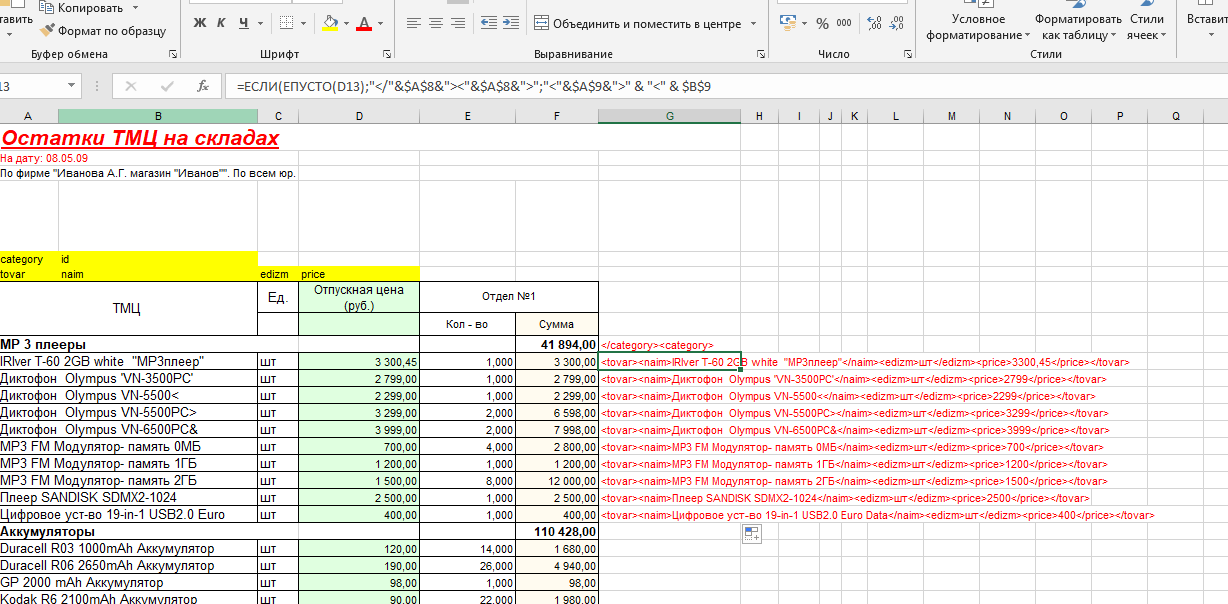


Рис. 74 Формула

Следующим шагом будет обработка атрибутов. В структуре XML-документа показаны 2 атрибута, имена у них одинаковы – id. Имя атрибута сохранено в ячейке B8. Значение атрибута id у элемента category будет равно имени категории и расположено в колонке A, значением атрибута id у элемента tovar будет уникальное число. Для формирования уникального числа можно воспользоваться функцией СТРОКА(), которая возвращает номер текущей строки. Формула примет следующий вид:

=ЕСЛИ(ЕПУСТО(D12);"</"&$A$8&"><"&$A$8 & " " & $B$8 & "=""" & A12 &""">";"<"&$A$9 & " " & $B$8 & "=""" & СТРОКА() &""">" & "<" & $B$9 & ">" & A12 & "</"&$B$9&">" & "<" & $C$9 & ">" & C12 & "</"& $C$9 &">" & "<" & $D$9 & ">" & D12 & "</"&$D$9&">"&"</"&$A$9&">").

Результат работы формулы (рис. 75):

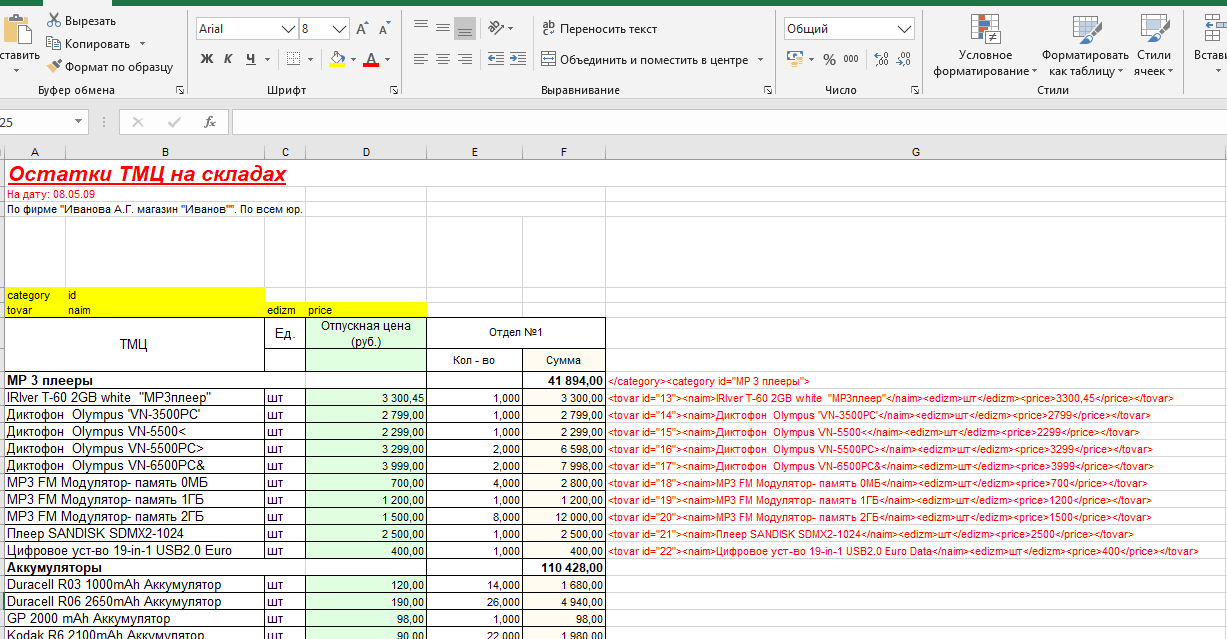


Рис. 75 Формула

На следующем шаге потребуется заменить наименовании товара и наименовании категории символы, которые не могут применяться в информационной части XML-документа на соответствующие сущности. Эти символы и сущности будут сохранены в ячейках G7…K8. Для

замены будет использована функция ПОДСТАВИТЬ и абсолютная адресация ячеек. Формула примет следующий вид:

=ЕСЛИ(ЕПУСТО(D12);"</"&$A$8&"><"&$A$8 & " " & $B$8 & "=""" & ПОДСТАВИТЬ(ПОДСТАВИТЬ(ПОДСТАВИТЬ(ПОДСТАВИТЬ(ПОДСТАВИТЬ(A12;$G$7;

$G$8);$H$7;$H$8);$I$7;$I$8);$J$7;$J$8);$K$7;$K$8) &""">";"<"&$A$9 & " " & $B$8 & "=""" & СТРОКА() &""">" & "<" & $B$9 & ">" & ПОДСТАВИТЬ(ПОДСТАВИТЬ(ПОДСТАВИТЬ(ПОДСТАВИТЬ(ПОДСТАВИТЬ(A12;$G$7; $G$8);$H$7;$H$8);$I$7;$I$8);$J$7;$J$8);$K$7;$K$8) & "</"&$B$9&">" & "<" & $C$9 & ">" & C12 & "</"& $C$9 &">" & "<" & $D$9 & ">" & D12 & "</"&$D$9&">"&"</"&$A$9&">")

Добавляем её для всех товаров (рис. 76):

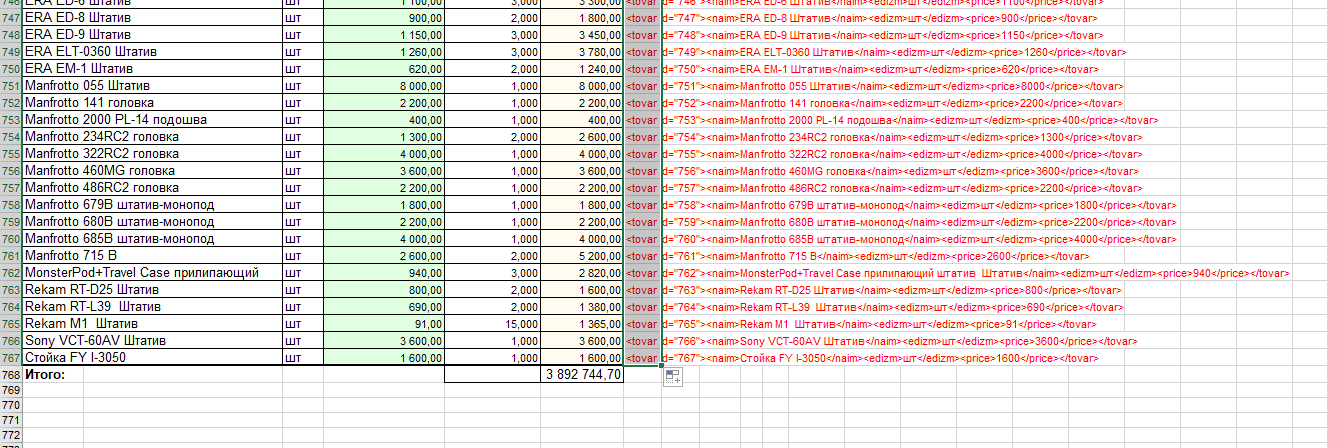


Рис. 76 Заполнение всех ячеек

На последнем шаге потребуется создать шаблон XML-документа в редакторе XMLPad с указанием начальных и конечных тегов для элементов price\_list, tovars, конечного тэга для элемента category и заголовка документа. Шаблон будет иметь следующий вид (рис. 77):

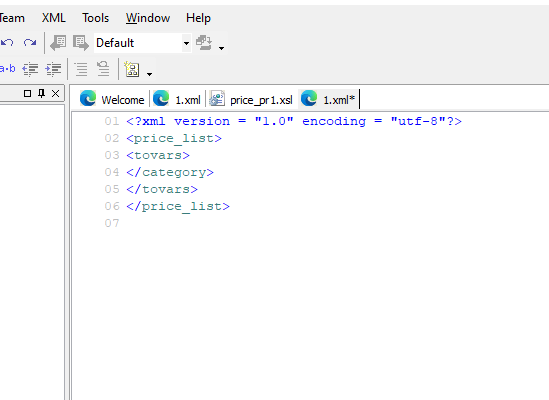


Рис. 77 XML Базовый

Далее копируем из Excel сформированную разметку и получаем следующий результат (рис. 78):

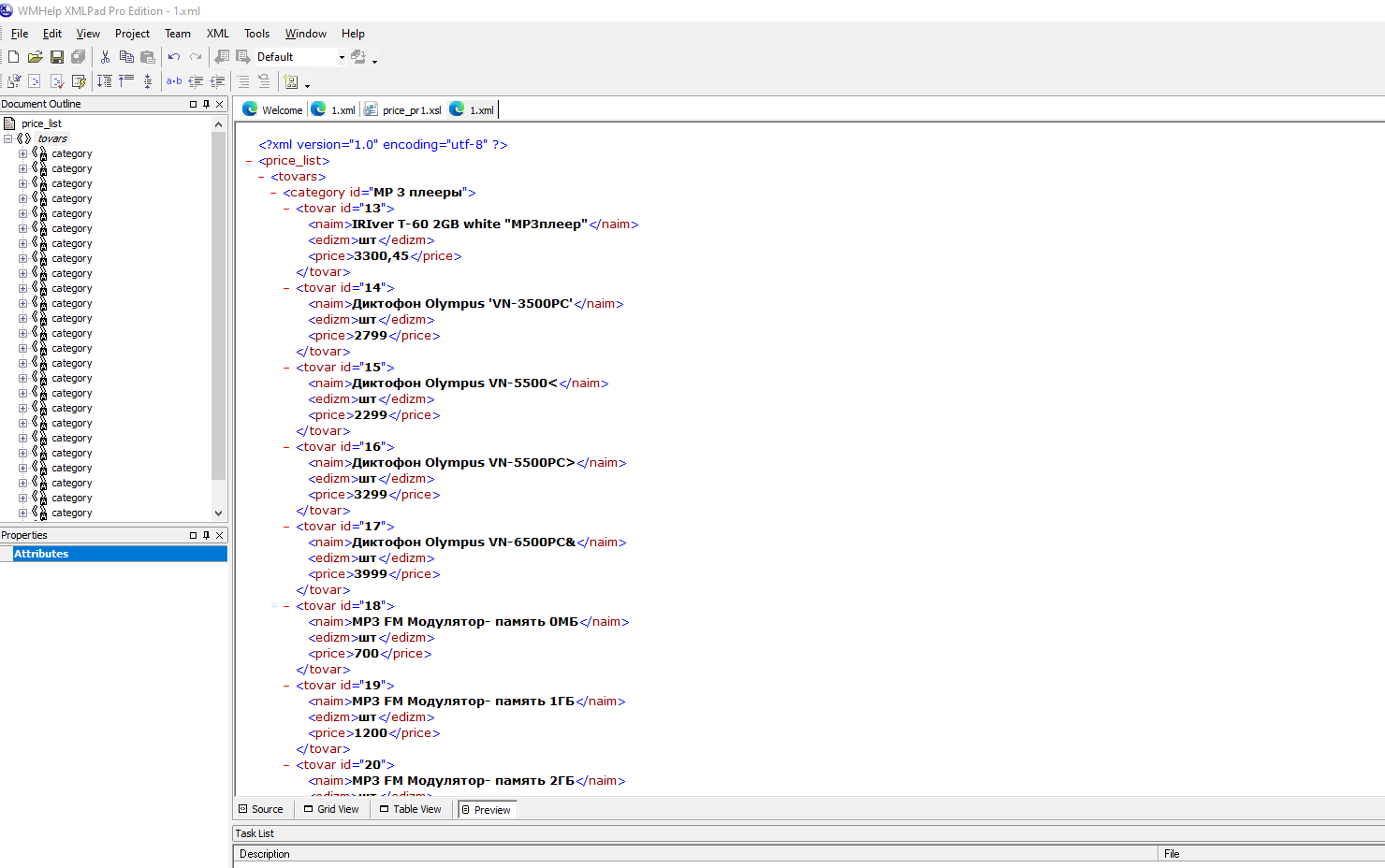


Рис. 78 Готовая разметка XML