Практическая работа № 4

Тема 1. Основные форматы данных YAML, INI. REST API

Цель работы: Изучить нормативные документы, стандарты, синтаксис данных YAML. Изучить два способы конвертации форматов Microsoft в YAML. Изучить стандарт YML-Yandex.

Написать код сценарий с форматом данных YAML.

Изучить формат INI и целье гоприменения.

План занятия.

1.Изучить теоретическую часть.

2.Изучить нормативные документы – стандарт формата обмена данных YAML, стандарт YML-Yandex.

3.Выполнить примеры создания кода сценария формата данных YAML.

4.Выполнить практические задания. Формат данных YAML.

5.Подготовить отчет практической работы.

6.Выполнить задание семинара.

7.Ответить на контрольные вопросы.

1. Теоретическая часть.

YAML (рекурсивный акроним англ. «YAML Ain't Markup Language» — «YAML — не язык разметки») — дружественный формат сериализации данных, концептуально близкий к языкам разметки, но ориентированный на удобство ввода-вывода типичных структур данных многих языков программирования [1].

В трактовке названия отражена история развития: на ранних этапах YAML расшифровывался как yet another markup language («ещё один язык разметки») и даже позиционировался как конкурент XML, но позже был переименован с целью акцентировать внимание на данных, а не на разметке документов [2].

Аббревиатура «Рекурсивный YAML» означает «YAML не является языком разметки». Это полезно в *файлах конфигурации* и тех *приложениях*, где данные передаются или хранятся.

Он предназначен для многих языков, таких как XML (расширяемый язык разметки). YAML кодирует скаляры. Людям легче читать по сравнению с JSON и XML [3].

Синтаксис

YAML-файл должен открываться 3 дефисами и закрывать 3 точками.

```
---
author:
  name: Andrey Ivabov
  job: Admin
  skill: Normal
...
```

Синтаксис YAML минималистичен, особенно по сравнению с синтаксисом XML. В спецификации указано, что большое влияние на YAML оказал стандарт RFC 822.

Ниже приведены образцы различных компонентов разметки. Следует заметить, что наличие варианта записи в однострочном формате делает JSON допустимым подмножеством YAML.

Последовательности (списки)

```
--- # Список фильмов: последовательность в блочном формате
- Casablanca
- Spellbound
- Notorious
--- # Список покупок: последовательность в однострочном формате
[milk, bread, eggs, juice]
```

Сопоставления имени и значения (словари)

```
--- # Блочный формат
name: "John Smith"
age: 33
--- # Однострочный формат
{name: "John Smith", age: 33}
```

Строковые данные
Переводы строк сохраняются

```
--- |
  There was a young fellow of Warwick
  Who had reason for feeling euphoric
      For he could, by election
      Have triune erection
  Ionic, Corinthian, and Doric
```

Переводы строк исчезают

```
--- >
  Wrapped text
  will be folded
  into a single
  paragraph

  Blank lines denote
  paragraph breaks
```

Последовательности из сопоставлений

```
- {name: John Smith, age: 33}
- name: Mary Smith
  age: 27
```

Сопоставления из последовательностей

```
men: [John Smith, Bill Jones]
women:
  - Mary Smith
  - Susan Williams
```

Основные элементы YAML:

- потоки YAML используют печатаемые Unicode-символы, как UTF-8, так и UTF-16
- отступы из пробелов (символы табуляции не допускаются) используются для обозначения структуры
- комментарии начинаются с символа «решётки» (#), могут начинаться в любом месте строки и продолжаются до конца строки
- списки обозначаются начальным дефисом (-) с одним членом списка на строку, либо члены списка заключаются в квадратные скобки ([ ]) и разделяются запятой и пробелом (,)
- ассоциативные массивы представлены двоеточием с пробелом (:) в виде ключ: значение, по одной паре ключ-значение на строку, либо в виде пар, заключённых в фигурные скобки и разделенных запятой и пробелом (,)
- ключ в ассоциативном массиве может иметь в качестве префикса вопросительный знак (?), что позволяет указать сложный ключ, например представленный в виде списка
- строки записываются без кавычек, однако могут быть заключены в одиночные или двойные кавычки
- внутри двойных кавычек могут быть использованы экранированные символы в C-стиле, начинающиеся с обратной косой (\)

- YAML позволяет задавать подстановки с помощью якорей & и псевдонимов (*).
- явное задание типа оформляется путём '!![указание типа]'. Пример, !!str 100 после разбора выдаст строковое значение «100» вместо целого числа 100.
- значения типа Дата/Время задаются в формате YYYY-MM-DD или YYYY-MM-DD HH:MM:SS. Если необходимо задать дату как строку, нужно заключать её в кавычки («2012-12-21»)

Использование

Среди программных систем, использующих YAML как формат для файлов конфигурации, — Ruby on Rails, Docker Compose, Kubernetes (притом поддерживается взаимно-однозначное соответствие с форматом JSON), Dancer, Symfony, GAE framework, Google App Engine, Dart, Home Assistant.

Браузер Chrome

Пример отображения языков (латиница, кириллица)
- латиница
<meta charset="utf-8" />
- кириллица (проверить для использования формата YAML)
<meta charset="ANSI" />

Литература

1.Yaml Википедия
https://ru.wikipedia.org/wiki/YAML
2. If YAML ain't markup language, what is it? (англ.). *StackOverflow* (6 августа 2011). Дата обращения: 15 марта 2021. Архивировано 23 апреля 2021 года.
3.Пример загрузки файла YAML для тестирования
https://www.learningcontainer.com/sample-yaml-file-download-for-testing/
4.YAML за 5 минут: синтаксис и основные возможности
https://tproger.ru/translations/yaml-za-5-minut-sintaksis-i-osnovnye-vozmozhnosti
5. Сравнение YAML с XML и JSON.
https://wiki.merionet.ru/articles/rukovodstvo-po-yaml-vse-chto-vam-nuzhno-znat-za-5-minut
6.YML (Yandex Market Language) — собственный стандарт Яндекса
https://yandex.ru/support2/marketplace/ru/assortment/auto/yml?ysclid=m0elu9e6sx906486502
7.Данные о товарах на Маркете
https://yandex.ru/support2/marketplace/ru/assortment/fields/#market-category
8.Способы создания прайс-лист (фид xml/yml) для Яндекс.Маркета
https://dzen.ru/a/XWkSY58nIQCtkvpA?ysclid=m0e90uck55993227099

9.Oren Ben-Kiki, Clark Evans, Brian Ingerson. YAML. Ain't Markup Language (YAML™). Version 1.1. Working Draft 2005-01-18-CVS. 2005. - 86 pages.
URL: https://yaml.org/spec/1.1/current.pdf
10. Работа с предпочтениями групповой политики: управление INI-файлами
URL: https://habr.com/ru/articles/178027/
11. Википедия INI
URL: https://ru.wikipedia.org/wiki/.ini

2.Нормативные документы. Формат YAML

Стандарт.
RFC 822 (англ.)
https://tools.ietf.org/html/rfc822
https://datatracker.ietf.org/doc/html/rfc822

Синтаксис.
RFC 822, спецификация (англ.)
https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%81

Програмнное обеспечение:
1) Notepad++
2) Блокнот (Windows)
3) Блокнот – онлайн: https://notepadonline.ru/app

YAML Viewer
https://jsonformatter.org/yaml-viewer
Online YAML To CSV converter
https://www.becsv.com/yaml-csv.php

3. Код сценария формата данных YAML

3.1 Примеры создания кода сценария формата данных YAML

Для качественного освоения практических работ изучите синтаксис и примеры, представленные в учебном пособии (авторы: Oren Ben-Kiki, Clark Evans, Brian Ingerson [9].

Давайте создадим YAML-файл, который будет содержать краткое резюме.

Внимание
YAML-файл должен открываться 3 дефисами и закрывать 3 точками.

Добавим их и три строки: имя автора, позицию, уровень скилов и статус [4]:

```
---
author:
  name: Ivan Katkov
  job: Tech writer
  skill: Normal
...
```

Помним, что YAML абсолютно лоялен к записи булевых значений, поэтому значения: «yes», «YES», «true», «True» — будут интерпретированы как логическое true.

Пример YAML:	Пример JSON:

`---`
`#YAML`
` Imaro:`
` author: Charles R. Saunders`
` language: English`
` publication-year: 1981`
` pages: 224`
`...`
	`{`
	`  "Imaro": {`
	`  "author": "Charles R. Saunders",`
	`  "language": "English",`
	`  "publication-year": "1981",`
	`  "pages": 224`
	`  }`
	`}`

Поддержка комментариев

YAML позволяет добавлять комментарии после символа #, как в Python.

```
# This a comment on the first.
# And this a commented second line.
```

Числа

Числа – это одни из скаляров, которые могут использоваться в качестве значения YAML-файлов. Числа бывают десятичными, с плавающей запятой, экспоненциальными, восьмеричными и шестнадцатеричными. Если добавлять их без одинарных или двойных кавычек, то они обрабатываются как строки.

```
decimal     : 10
float       : 2.5
exponential : 5.0e+12
infinity    : .inf
octal       : 0o12
hexadecimal : 0xF
```

Примечание: Будьте осторожны при представлении некоторых стандартных типов чисел (время, восьмеричные и шестнадцатеричные коды и т.д.). Они могут быть некорректно интерпретированы. Вот несколько примеров.

```
time  : 6:00
hexval: 0x12
octval: 0o25
```

В этом примере время преобразуется в минуты, поэтому 6:00 интерпретируется как 360. 0x12 обрабатывается как шестнадцатеричный код, поэтому оно переводится в десятеричное значение 18, а 0o25 интерпретируется в десятеричное значение 21. Так что если вы не собираетесь использовать их в таком виде, то добавляйте эти значения в кавычках.

Примечание: Начиная со спецификации YAML 1.2, 0o представляет восьмеричные значения. В более ранних версиях для этих целей использовался 0.

Логические значения

Логические значения похожи на другие языки программирования/представления с двумя состояниями: true или false. Логическими значениями в YAML-файлах считаются не только стандартные true/false, но также yes/no и on/off (исключение: если они добавлены в кавычках).

Примечание: Следите за тем, как вы используете значения yes/no и on/off. Если добавлять их не в одинарных кавычках, то они будут интерпретироваться как логические значения true/false.

Отступы

В синтаксисе YAML-файлов используется система отступов, как в Python. Необходимо использовать пробелы, а не табуляцию, чтобы избежать путаницы.

Это избавляет от лишних символов, которые есть в JSON и XML (кавычки, скобки, фигурные скобки).

В итоге читаемость файла значительно повышается.

Отсутствие исполняемых файлов

YAML не содержит исполняемых файлов. Поэтому можно безопасно обмениваться YAML-файлами с третьей стороной.

Чтобы использовать исполняемые файлы, YAML нужно интегрировать с другими языками, например Perl или Java.

Пары ключ-значение

Большинство данных в YAML-файле хранятся в виде пары ключ-значение, где ключ — это имя пары, а значение — связанные данные.

Скаляры и маппинг

Скаляр представляет собой одно значение, которому соответствует имя.

YAML поддерживает стандартные типы: int и float, boolean, string и null.

Они могут быть представлены в разных видах: шестнадцатеричном, восьмеричном или экспоненциальном. Также существуют специальные типы для математических сущностей, такие как: бесконечность, -бесконечность и NAN.

```
integer: 25
hex: 0x12d4 #равно 4820
octal: 023332 #равно 9946
float: 25.0
exponent: 12.3015e+05 #равно 1230150.0
boolean: Yes
string: "25"
infinity: .inf # преобразуется в бесконечность
neginf: -.Inf #преобразуется в минус бесконечность
not: .NAN #Not a Number
null: ~
```

Строки

Строка — это коллекция символов, которая может содержать слово или предложение. Можно использовать либо |, для отдельных строк, либо >, для параграфов. Кавычки в YAML не нужны.

В YAML-файле вы можете прописывать строки без кавычек. Либо добавлять их в одинарные или двойные кавычки. Можно даже пользоваться всеми тремя способами.

```
signature: "John Williams
Sales Executive
XYZ Company LA."
```

Но при использовании любого из специальных символов ниже:

```
:, {, }, [, ], ,, &, *, #, ?, |, -, <, >, =, !, %, @, `
```

Не нужно ничего экранировать. Воспользуйтесь одинарными кавычками – с ними строка будет интерпретироваться как нужно.

Если в строке содержатся специальные символы, которые нужно экранировать с помощью , то добавлять такую строку нужно в двойных кавычках. Пример:

```
str: Hello World
data: |
   Это
   Отдельные
   Строки
data: >
   Это
   один параграф
   текста
```

Одиночная строка

Если вы хотите разбить одну строку на несколько строчек, но синтаксический анализатор должен будет интерпретировать их как одиночную строку, то сделайте следующее:

```
message: >
  this is a normal string
  which spans more than multiple lines
  but needs to be treated
  as a single line.
```

Начните код с символа > , а затем впишите содержимое строки в виде блока с отступами. В конце проанализированного содержимого добавляется . Если вам не нужен символ новой строки, то воспользуйтесь символом >-.

Многострочное значение

При добавлении строки, которая разделяется на несколько строк и анализируется так же, как написана, воспользуйтесь схемой ниже.

```
message: |
  this is a multi-line string
  which spans across multiple lines
  and needs to be treated
  as a multi-line string.
```

| добавляет в конце проанализированного содержимого символ перехода на новую строку. Если вы не хотите видеть в конце проанализированного содержимого символ , то лучше воспользуйтесь |-.

Нулевые значения

Вы можете представить нулевое значение с помощью ~ или null. Такие значения добавляются без кавычек.

```
foo: ~
bar: null
```

Массивы

В YAML значения можно представить в форме массивов/списков. Пример массива данных:

```
teams:
  - name: Australia
    rank: 3
  - name: New Zealand
    rank: 4
  - name: England
    rank: 1
  - name: India
    rank: 2
```

В примере выше представлен массив объектов с названием и рангом команды. Обязательно проверьте, что отбивка строк и отступы заданы единообразно. В противном случае, файл будет некорректным.

Каждый элемент массива обозначается через дефис -. Все элементы должны находиться на одном уровне с одинаковыми отступами.

Если вместо объектов в массиве используются примитивные элементы, то их можно представить следующим образом:

```
teams: [ Australia, New Zealand, England, India ]
```

Объекты

Объекты в YAML представляются так:

```
student:
  name: John
  age: 22
  city: NY
  college: NYU
  gpa: 3.5
```

Все атрибуты внутри объекта должны находиться на одном уровне и с одинаковым отступом. Именно это условие указывает на принадлежность к одному объекту и гарантирует корректность YAML-документа.

Последовательности

Последовательности — это структуры данных похожие на списки или массивы, которые хранят несколько значений под одним ключом. Они определяются с помощью отступов или [ ].

```
shopping:
- milk
- eggs
- juice
```

Однострочные последовательности выглядят лаконичнее, но хуже читаются.

```
shopping: [milk, eggs, juice]
```

Словари

Словари — это коллекции пар ключ-значение, которые хранятся под одним ключом. Они позволяют разделить данные на логические категории.

```
Employees:
- dan:
    name: Dan D. Veloper
    job: Developer
    team: DevOps
- dora:
   name: Dora D. Veloper
   job: Project Manager
   team: Web Subscriptions
```

Словари могут содержать более сложные структуры, что позволяет хранить сложные реляционные данные.

Возможности YAML?

Anchors (якоря)
Templates (шаблоны)
Взаимодействие с Docker, Ansible и т. д.
Расширенные последовательности и маппинг.
Расширенные типы данных (timestamp, null и т. д.)

Сравнение YAML с XML и JSON [5].
Код сценария формата данных YAML, XML, JSON

YAML

```yaml
apis:
  - name: login
    port: 8080
  - name: profile
    port: 8090
```

XML

```xml
<apis>
    <api>
        <name>login</name>
        <port>8080</port>
    </api>
    <api>
        <name>profile</name>
        <port>8090</port>
    </api>
</apis>
```

JSON

```json
{
  "apis": [
    {
      "name": "login",
      "port": 8080
    },
    {
      "name": "profile",
      "port": 8090
    }
  ]
}
```

Символы индикатора

Символы-индикаторы имеют особую семантику, используемую для описания содержимого документа YAML. В следующей таблице это показано подробно.

Таблица – символы индикатора

Sr.No.	Характер и функциональность
1	_ Он обозначает запись последовательности блоков
2	? Он обозначает ключ сопоставления
3	: Он обозначает значение сопоставления
4	, Он обозначает запись для сбора потока
5	[Это запускает последовательность потоков
6	] Это завершает последовательность операций
7	{ Это запускает отображение потока
8	} На этом отображение потока завершается
9	# Он обозначает комментарии
10	& Он обозначает свойство привязки узла
11	* Он обозначает псевдоним узла
12	! Он обозначает тег узла
13	| Он обозначает скаляр буквального блока
14	> Он обозначает скаляр сложенного блока

15	` Скаляр потока, заключенный в одинарные кавычки, заключен в кавычки
16	" Двойные кавычки окружают скаляр потока, заключенный в двойные кавычки
17	% Это обозначает используемую директиву

3.2 Стандарт YML (Yandex)

YML (Yandex Market Language) — собственный стандарт Яндекса, основанный на XML. В YML-файлах можно целиком описать каталог магазина в формате, удобном для автоматической генерации [6].

Файлы для обновления каталога и файлы для управления размещением

С помощью YML-файлов можно решать две задачи:

- добавлять товары в каталог и обновлять информацию о них;
- управлять размещением товаров в магазинах.

YML-файлы, решающие разные задачи, отличаются набором обязательных элементов.

Заголовок

Нужен в любом YML-файле.

Пишется так:

```
<?xml version="1.0" encoding="UTF-8"?>
```

Заголовок занимает первую строку и начинается с нулевого символа. Подойдут кодировки UTF-8 и Windows-1251.

Корневой элемент <yml_catalog>

Нужен в любом YML-файле.

В любом XML-документе есть корневой элемент. Формат YML в качестве корневого использует элемент <yml_catalog> с атрибутом date.

В атрибуте укажите дату и время момента, на который актуальны данные в файле. Загружаемая версия каталога должна быть не старше 10 дней.

Дату и время нужно указать согласно стандарту RFC 3339, вот так:

```
<yml_catalog date="2022-05-22T14:37+03:00">
```

Обязательно указывайте часовой пояс. Он отсчитывается от UTC — например, красноярское время записывается так:

```
<yml_catalog date="2022-05-22T15:08+07:00">
```

Дату и время в будущем указывать нельзя.

Элемент <shop>
Нужен в любом YML-файле.

В элемент <yml_catalog> в единственном экземпляре входит элемент <shop> без атрибутов. Он описывает магазин, для которого вы готовите файл.

В него вложены:

Элемент	**Тип данных**	**Смысл**
<name>	Текст: <name>BestSeller</name>	Название вашего магазина
<company>	Текст: <company>Tne Best inc.</company>	Название вашей компании
<url>	Текст: <url>http://best.seller.ru</url>	Адрес сайта магазина, записанный согласно стандарту RFC 3986
<platform>	Текст: <platform>uCoz</platform>	Название системы управления контентом
<categories>	Содержит вложенные элементы <category>	Список категорий товаров, продаваемых в магазине
<offers>	Содержит вложенные элементы <offer>	Список предложений — товаров, продающихся в магазине, с ценами

Элемент <categories>

Нужен только в YML-файле, управляющем товарами

Вложен в элемент <shop>, не имеет атрибутов. Помещается **перед** элементом <offers>. Содержит сколько угодно элементов <category>, каждый из которых описывает одну из категорий доступных в магазине товаров.

При создании категорий следуйте рекомендациям:

- указывайте конкретные категории — например, набор ножей лучше отнести к категории **Столовые приборы**, а не просто **Посуда**;

- выбирайте категории, которые описывают товар, а не абстрактный признак — например, лучше указать **Духи**, а не **Подарки**.

Каждой категории нужно присвоить уникальный идентификатор — целое положительное число длиной до 18 цифр — и записать его в атрибут id. Запись числа не должна начинаться с нуля — например, 055 не подойдет.

Чтобы вложить одну категорию в другую, используйте атрибут parentId.

Получится так:

```
<categories>
    <category id="1">Книги</category>
    <category id="2" parentId="1">Детективы</category>
    <category id="3" parentId="1">Боевики</category>
    <category id="4">Видео</category>
    <category id="5" parentId="4">Комедии</category>
    <category id="6">Принтеры</category>
    <category id="7">Оргтехника</category>
</categories>
```

Элемент <offers>

Нужен в любом YML-файле. Содержимое вложенных элементов <offer> зависит от задачи, которую решает YML-файл.

Помещается **после** элемента <categories>. Не имеет атрибутов. Содержит сколько угодно элементов <offer>, каждый из которых описывает один товар в магазине. У <offer> есть обязательный атрибут id, который содержит ваш SKU товара. Что такое SKU

Внутри <offer> нужно указать характеристики товара (название, описание, производителя и так далее) или параметры размещения (цену, скидки, остатки и так далее).

Параметры доставки и самовывоза

Если магазин работает по модели DBS, в YML-файле можно задать, описывающие параметры доставки и самовывоза. Их нужно разместить:

- перед <offers>, чтобы задать параметры для всего магазина;

- внутри <offer>, если нужно переопределить параметры для конкретного товара.

Чтобы Маркет учитывал параметры, заданные в YML-файле, нужно включить опцию Использовать данные из прайс-листа в кабинете.

Элементы <delivery> и <pickup>

Нужны только в YML-файле, управляющем размещением.

<delivery>

Указывает, доступна ли курьерская доставка. Значение по умолчанию — true.

Добавьте в файл <delivery>false</delivery>, если магазин не доставляет товары.

<pickup>

Указывает, доступен ли самовывоз. Значение по умолчанию — true.

Добавьте в файл <pickup>false</pickup>, если **самовывоз недоступен**.

Элемент <delivery-options>

Нужен только в YML-файле, управляющем размещением.

Не имеет атрибутов. Содержит до пяти элементов <option>, каждый из которых описывает один из способов курьерской доставки (например, обычная, ускоренная и так далее).

Может быть вложен в <shop> и в <offer>.

Внутри <shop> элемент <delivery-options> определяет способы доставки, которые магазин предлагает по умолчанию.

Внутри <offer> элемент <delivery-options> указывает, что для конкретного товара действуют не общие условия доставки, а специальные.

Параметры способа доставки <option> задаются атрибутами:

- cost — не используется. Задайте любое число, чтобы файл прошел техническую проверку.

- days — срок в рабочих днях, целое число или интервал, записанный через дефис. Для доставки в день заказа укажите 0, для доставки на следующий день — 1. Максимальное значение — 60. На Маркете нельзя продавать товары с неизвестным сроком доставки. Если поле days оставить пустым, то товар будет скрыт с витрины.

- order-before — до которого часа можно оформить доставку этим способом, чтобы срок начал отсчитываться с сегодняшнего дня. Целое число от 0 до 24. Это необязательный атрибут, значение по умолчанию — 13.

Результат:

```
<delivery-options>
    <option cost="123" days="4" order-before="18"/>
</delivery-options>
```

Здесь указано, что покупатель может оформить доставку в течение 4 дней. Если он оформляет заказ после 18:00, срок начнет отсчитываться со следующего дня. Способы доставки должны отличаться друг от друга и ценой, и сроками.

Элемент <pickup-options>

Нужен только в YML-файле, управляющем размещением.

Не имеет атрибутов. Содержит до пяти элементов <option>, каждый из которых описывает один из способов самовывоза.

Может быть вложен в <shop> и в <offer>.

Внутри <shop> элемент <pickup-options> определяет способы самовывоза, которые магазин предлагает по умолчанию.

Внутри <offer> элемент <pickup-options> указывает, что для конкретного товара действуют не общие условия самовывоза, а специальные.

Параметры способа самовывоза задаются атрибутами:

- cost — не используется. Задайте любое число, чтобы файл прошел техническую проверку.

- days — срок в рабочих днях, целое число или интервал, записанный через дефис. Для самовывоза в день заказа укажите 0, для самовывоза на следующий день — 1. Максимальное значение — 60. На Маркете нельзя продавать товары с неизвестным сроком самовывоза. Если поле days оставить пустым, то товар будет скрыт с витрины.

- order-before — до которого часа можно оформить самовывоз этим способом, чтобы срок начал отсчитываться с сегодняшнего дня. Целое число от 0 до 24. Это необязательный атрибут, значение по умолчанию — 13.

Результат:

```
<pickup-options>
    <option cost="123" days="2" order-before="18"/>
</pickup-options>
```

Здесь указано, что товар приедет на пункт выдачи в течение двух дней. Если покупатель оформляет заказ с самовывозом после 18:00, срок начнет отсчитываться со следующего дня.

4. Темы практических заданий

Цель: Написать код сценария управления размещением товара и услуг в формате YML (стандарт YML-Yandex).

Задание.

1.Написать код сценария управления размещением товара и услуг (рисунок 1).

2.Условие, что товар приедет на пункт выдачи от 1 до 5 дней. Если покупатель оформляет заказ с самовывозом после 16:00, если доставка с 10:00, срок начнет отсчитываться со следующего дня.

3.Сохранить файл с расширением ***.xml.

5.Отчет практической работы

Подготовить отчет практической работы в соответствии со структурой отчета. Отчет представить в формате А4, файл Word.

Структура отчета

1. Титульный лист.
2. Тема. Цель работы. Задание.
3. Решение (программное обеспечение код, рисунки, таблицы и др.).
5. Выводы.
6. Список литературы.
7. Приложение (при необходимости).

Отчет, представить на проверку:
Папка: Иванов АА (ПИ21-1)/ПР-4/
Файл (word): **ИвановАА-пр4.docx**
Файлы: ***.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<yml_catalog date="2022-08-15T14:37:38+03:00">
    <shop>
        <name>BestSeller</name>
        <company>Tne Best inc.</company>
        <url>http://best.seller.ru</url>
        <platform>uCoz</platform>
        <version>1.0</version>
        <agency>Технологичные решения</agency>
        <email>example-email@gmail.com</email>
        <delivery-options>
            <option cost="200" days="1"/>
        </delivery-options>
        <pickup-options>
            <option cost="200" days="1"/>
        </pickup-options>
        <offers>
            <offer id="901299">

<url>http://best.seller.ru/product_page.asp?pid=12345</url>
                <price>8990</price>
                <oldprice>9990</oldprice>

<enable_auto_discounts>true</enable_auto_discounts>
                <currencyId>RUR</currencyId>
                <vat>VAT_20</vat>
                <delivery>true</delivery>
                <pickup>true</pickup>
                <delivery-options>
                    <option cost="300" days="1" order-
before="18"/>
                </delivery-options>
                <pickup-options>
                    <option cost="300" days="1-3"/>
                </pickup-options>
                <count>100</count>
                </offer>
            <offer id="123467">

<url>http://best.seller.ru/product_page.asp?pid=12345</url>
                <price>1099</price>
                <oldprice>1399</oldprice>

<enable_auto_discounts>false</enable_auto_discounts>
                <currencyId>USD</currencyId>
                <vat>VAT_20</vat>
                <count>23</count>
            </offer>
        </offers>
    </shop>
</yml_catalog>
```

Рисунок 1 – Код сценария управления размещением товара и услуг

6.Семинар № 2

Тема: Основные форматы данных (формат YAML, YML-Yandex)

Цель: Организация занятий с возможностью изучения и применения новых, актуальных методов, технических и технологических решений, решения проблем, перспектив для повышения эффективности технологии автоматизации открытых данных.

Структура семинара (План занятий)

1.Организация (проверка пристуствующих, готовность обучающихся к занятию, формирование групп – 2-3 чел. на одну тему).

2.Ознакомление студентами с темами семинара.

3.Выбор темы семинара (формирование списка групп семинара и тем).

4.Выполнить задание семинара:

- поиск информации по тематике;
- составить рабочий проект презентации (согласовать с преподавателем);
- подготовить презентацию и доклад по тематике семинара;
- выступление (2-3 мин/группа). Ответы на вопросы.

5.Заключение.Подведение итогово семинара.

Темы семинара

1.Как создать прайс-лист (фид xml/yml) для Яндекс.Маркета [8].

1.1.Виды прайс-листов;

1.2.Создание прайс-лист при помощи плагина/модуля CMS;

1.3.Создание прайс-лист при помощи шаблона Excel;

1.4.Создаем YML-файл вручную;

https://drive.google.com/drive/u/0/folders/1ODb0l3zDaPfcnvwcLdciIEZDODkcDlpj

1.5.Валидация фида.

2.Преимущества использования формата YAML.

3.Недостатки, проблемы, уязвимые места объектов формата YAML.

4.Форматы данных YAML, примеры в интернет-магазине.

5.Нормативные документы РФ для применения формата YAML.

6.Изучить формат INI и цель его применения [10, 11].

Источники информации

1.Статьи (Периодические публикации, Журналы РФ, зарубежные) 2-3 статьи.

2.Практические материалы сценариев (примеры) с официальных и частных сайтов.

7.Контрольные вопросы

1.Для какой цели был создан формат YAML, YML-Yandex.

2.Какие используют способы создания формата YAML.

3.Назовите синтаксис YAML.

4.В каких отраслях применяют формат YAML.

5.Какой нормативный документ определяет использование формата YAML в РФ и зарубежом.

6.Сравните код сценария формата данных YAML, XML, JSON.

7.Назовите преимущества и недостакти способов сценария формата данных YAML, XML, JSON.