Лабораторная работа

Разработать XML-формат для представления расписания учебных занятий вашей группы. В расписании должна храниться информация о занятиях на каждый день недели. Для каждого учебного занятия указаны название предмета, аудитория, преподаватель, время начала и окончания, тип занятия (лекция или практика).

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

<timesheet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="timesheet\_schema.xsd">

  <day name="Понедельник">

    <tutorial subject="Математика" room="11" teacher="Иванов И.И." start="9:00" end="10:30" type="лекция"/>

    <tutorial subject="Русский язык" room="12" teacher="Петрова М.С." start="11:00" end="12:30" type="практика"/>

    <tutorial subject="Химия" room="14" teacher="Васильева О.П." start="13:00" end="14:30" type="лекция"/>

  </day>

  <day name="Вторник">

    <tutorial subject="Физика" room="13" teacher="Сидоров А.В." start="10:00" end="11:30" type="лекция"/>

    <tutorial subject="Химия" room="14" teacher="Васильева О.П." start="12:00" end="13:30" type="практика"/>

    <tutorial subject="Биология" room="15" teacher="Соколова Л.М." start="14:00" end="15:30" type="лекция"/>

  </day>

  <day name="Среда">

    <tutorial subject="Математика" room="11" teacher="Иванов И.И." start="9:00" end="10:30" type="практика"/>

    <tutorial subject="Русский язык" room="12" teacher="Петрова М.С." start="11:00" end="12:30" type="лекция"/>

    <tutorial subject="Геометрия" room="16" teacher="Соколов П.А." start="14:00" end="15:30" type="лекция"/>

  </day>

  <day name="Четверг">

    <tutorial subject="Физика" room="13" teacher="Сидоров А.В." start="10:00" end="11:30" type="практика"/>

    <tutorial subject="Химия" room="14" teacher="Васильева О.П." start="12:00" end="13:30" type="лекция"/>

    <tutorial subject="Геометрия" room="16" teacher="Соколов П.А." start="14:00" end="15:30" type="лекция"/>

  </day>

  <day name="Пятница">

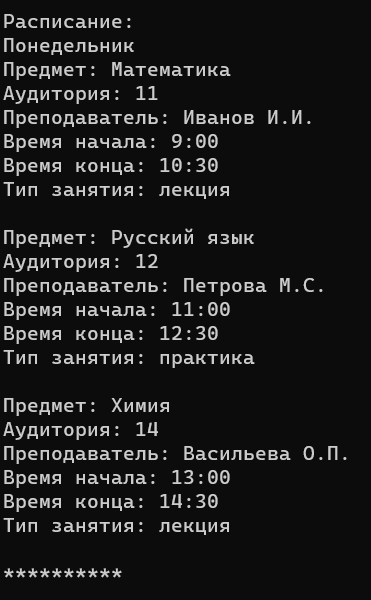
    <tutorial subject="Биология" room="15" teacher="Соколова Л.М." start="10:00" end="11:30" type="практика"/>

    <tutorial subject="Геометрия" room="16" teacher="Соколов П.А." start="12:00" end="13:30" type="практика"/>

  </day>

</timesheet>

**В разработанном формате представить расписание текущей недели.**

****

public void ScheduleForTheCurrentWeek()

{

Console.WriteLine("Расписание: ");

XmlNodeList days = doc.SelectNodes("/timesheet/day");

foreach (XmlNode day in days)

{

Console.WriteLine(day.Attributes["name"].Value);

XmlNodeList tutorials = day.SelectNodes("tutorial");

foreach (XmlNode tutorial in tutorials)

{

Console.WriteLine($"Предмет: {tutorial.Attributes["subject"].Value}");

Console.WriteLine($"Аудитория: {tutorial.Attributes["room"].Value}");

Console.WriteLine($"Преподаватель: {tutorial.Attributes["teacher"].Value}");

Console.WriteLine($"Время начала: {tutorial.Attributes["start"].Value}");

Console.WriteLine($"Время конца: {tutorial.Attributes["end"].Value}");

Console.WriteLine($"Тип занятия: {tutorial.Attributes["type"].Value}");

Console.WriteLine();

}

Console.WriteLine(new string('\*',10));

Console.WriteLine();

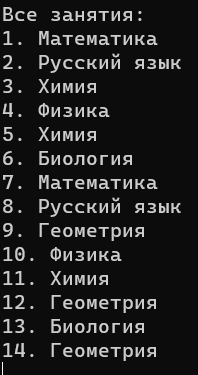
}

Console.ReadLine();

}

**Разработать xPath-запросы**

* Получить все занятия на данной неделе



public void AllLessons()

{

XmlNodeList tutorials = doc.SelectNodes("//tutorial");

Console.WriteLine("Все занятия: ");

int i = 1;

foreach (XmlNode tutorial in tutorials)

{

string subject = tutorial.Attributes["subject"].Value;

Console.WriteLine(i + ". " + subject);

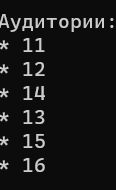
i++;

}

Console.ReadLine();

}

* Получить все аудитории, в которых проходят занятия.



public void AllRooms()

{

XmlNodeList rooms = doc.SelectNodes("//tutorial/@room");

HashSet<string> uniqueRooms = new HashSet<string>();

foreach (XmlNode room in rooms)

{

uniqueRooms.Add(room.Value);

}

Console.WriteLine("Аудитории:");

foreach (string room in uniqueRooms)

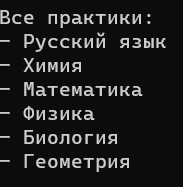
{

Console.WriteLine("\* "+ room);

}

}

* Получить все практические занятия на неделе.



public void AllPractice()

{

XmlNodeList practices = doc.SelectNodes("//tutorial[@type='практика']");

Console.WriteLine("Все практики:");

foreach (XmlNode tutorial in practices)

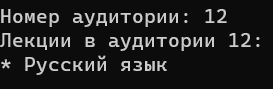
{

Console.WriteLine("- "+tutorial.Attributes["subject"].Value);

}

}

* Получить все лекции, проводимые в указанной аудитории.



public void AllLectureInRoom()

{

Console.Write("Номер аудитории: ");

string room = Console.ReadLine();

XmlNodeList lectures = doc.SelectNodes("//tutorial[@room='" + room + "' and @type='лекция']");

Console.WriteLine("Лекции в аудитории " + room+ ":");

if (lectures != null && lectures.Count > 0)

{

foreach (XmlNode tutorial in lectures)

{

Console.WriteLine("\* "+tutorial.Attributes["subject"].Value);

}

}

else

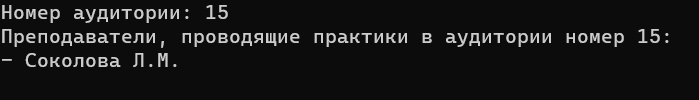
{

Console.WriteLine("Лекций в данной аудитории нет");

}

}

* Получить список всех преподавателей, проводящих практики в указанной аудитории.



public void AllTeachersPracticeInRoom()

{

Console.Write("Номер аудитории: ");

string room = Console.ReadLine();

XmlNodeList tutorials = doc.SelectNodes($"//tutorial[@room='{room}' and @type='практика']");

Console.WriteLine($"Преподаватели, проводящие практики в аудитории номер {room}:");

if (tutorials != null && tutorials.Count > 0)

{

foreach (XmlNode tutorial in tutorials)

{

string teacher = tutorial.Attributes["teacher"].Value;

Console.WriteLine("- "+ teacher);

}

}

else

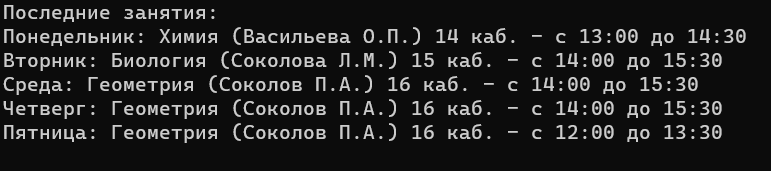
{

Console.WriteLine($"В данной аудитории нет практики");

}

}

* Получить последнее занятие для каждого дня недели.



public void LastLesson()

{

XmlNodeList lastLessons = doc.SelectNodes("//day/tutorial[last()]");

Console.WriteLine("Последние занятия:");

if (lastLessons != null && lastLessons.Count > 0)

{

foreach (XmlNode tutorial in lastLessons)

{

string day = tutorial.ParentNode.Attributes["name"].Value;

string subject = tutorial.Attributes["subject"].Value;

string teacher = tutorial.Attributes["teacher"].Value;

string room = tutorial.Attributes["room"].Value;

string start = tutorial.Attributes["start"].Value;

string end = tutorial.Attributes["end"].Value;

Console.WriteLine($"{day}: {subject} ({teacher}) {room} каб. - с {start} до {end}");

}

}

else

{

Console.WriteLine("В расписании нет занятий");

}

}

* Получить общее количество занятий за всю неделю.



public void TotalCountLessons()

{

int totalLessons = GetTotalLessons(doc);

Console.WriteLine($"Общее количество занятий: {totalLessons}");

static int GetTotalLessons(XmlDocument doc)

{

// Получаем узлы всех занятий

XmlNodeList allLessons = doc.SelectNodes("//tutorial");

if (allLessons != null)

{

return allLessons.Count;

}

return 0;

}

}

**Описать DTD схему для разработанного формата. Произвести валидацию xml-документа**

<!ELEMENT timesheet (day+)>

<!ELEMENT day (tutorial+)>

<!ATTLIST day name CDATA #REQUIRED>

<!ELEMENT tutorial EMPTY>

<!ATTLIST tutorial

    subject CDATA #REQUIRED

    room CDATA #REQUIRED

    teacher CDATA #REQUIRED

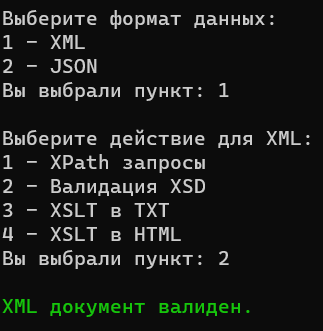
    start CDATA #REQUIRED

    end CDATA #REQUIRED

    type CDATA #REQUIRED

>

**Описать XML Schema для разработанного формата. Произвести валидацию xml-документа.**



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

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="timesheet">

    <xs:complexType>

      <xs:sequence>

        <xs:element name="day" maxOccurs="unbounded">

          <xs:complexType>

            <xs:sequence>

              <xs:element name="tutorial" maxOccurs="unbounded">

                <xs:complexType>

                  <xs:attribute name="subject" type="xs:string" use="required"/>

                  <xs:attribute name="room" type="xs:string" use="required"/>

                  <xs:attribute name="teacher" type="xs:string" use="required"/>

                  <xs:attribute name="start" type="xs:string" use="required"/>

                  <xs:attribute name="end" type="xs:string" use="required"/>

                  <xs:attribute name="type" type="xs:string" use="required"/>

                </xs:complexType>

              </xs:element>

            </xs:sequence>

            <xs:attribute name="name" type="xs:string" use="required"/>

          </xs:complexType>

        </xs:element>

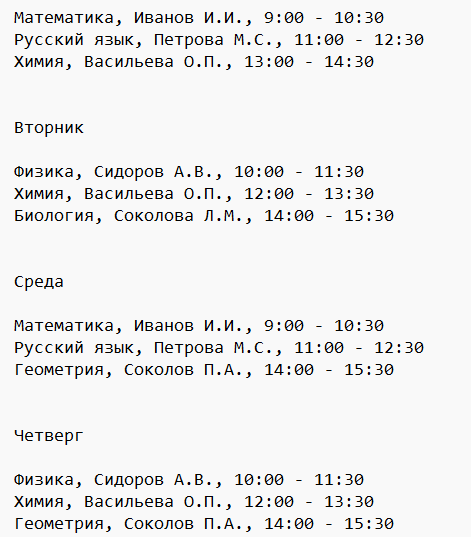
      </xs:sequence>

    </xs:complexType>

  </xs:element>

</xs:schema>

**Описать XSLT-преобразование xml-документа в текстовый вид (\*.txt).**



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

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="timesheet">

<xsl:for-each select="day">

<xsl:value-of select="name"/>

<xsl:text>&#xA;&#xA;</xsl:text>

<xsl:for-each select="tutorial">

<xsl:value-of select="subject"/>

<xsl:text>, </xsl:text>

<xsl:value-of select="teacher"/>

<xsl:text>, </xsl:text>

<xsl:value-of select="start"/>

<xsl:text> - </xsl:text>

<xsl:value-of select="end"/>

<xsl:text>&#xA;</xsl:text>

</xsl:for-each>

<xsl:text>&#xA;&#xA;</xsl:text>

</xsl:for-each>

</xsl:template>

</xsl:stylesheet>

**Описать XSLT-преобразование xml-документа в html-страницу (расписание должно быть представлено в виде таблицы)**



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

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">

<html>

<head>

<style>

h2 { text-align: center }

table {

border-collapse: collapse;

width: 100%;

margin: 15px 0;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

font-family: 'Arial', sans-serif;

}

th, td {

border: 1px solid #ddd;

padding: 12px;

text-align: left;

}

th {

background-color: #f2f2f2;

font-weight: bold;

}

tr:nth-child(even) {

background-color: #f9f9f9;

}

tr:nth-child(odd) {

background-color: #e6e6e6;

}

tr:hover {

background-color: #d9edf7;

transition: background-color 0.3s;

}

</style>

</head>

<body>

<h2 >Расписание</h2>

<table>

<tr>

<th>День</th>

<th>Предмет</th>

<th>Преподаватель</th>

<th>Аудитория</th>

<th>Начало</th>

<th>Конец</th>

<th>Тип</th>

</tr>

<xsl:apply-templates select="//tutorial"/>

</table>

</body>

</html>

</xsl:template>

<xsl:template match="tutorial">

<tr>

<td>

<xsl:value-of select="../name"/>

</td>

<td>

<xsl:value-of select="subject"/>

</td>

<td>

<xsl:value-of select="teacher"/>

</td>

<td>

<xsl:value-of select="room"/>

</td>

<td>

<xsl:value-of select="start"/>

</td>

<td>

<xsl:value-of select="end"/>

</td>

<td>

<xsl:value-of select="type"/>

</td>

</tr>

</xsl:template>

</xsl:stylesheet>

**Решить аналогичные задачи для формата JSON**

public class DesignerJSON

{

JObject timesheetJson;

public DesignerJSON()

{

timesheetJson = new JObject();

string jsonText = System.IO.File.ReadAllText("timesheet.json");

timesheetJson = JObject.Parse(jsonText);

}

public void ScheduleForTheCurrentWeek()

{

Console.WriteLine("Расписание:");

foreach (var day in timesheetJson["timesheet"]["day"])

{

Console.ForegroundColor = ConsoleColor.Green;

Console.WriteLine(day["name"]);

Console.ResetColor();

foreach (var tutorial in day["tutorial"])

{

Console.WriteLine($"Предмет: {tutorial["subject"]}");

Console.WriteLine($"Аудитория: {tutorial["room"]}");

Console.WriteLine($"Преподаватель: {tutorial["teacher"]}");

Console.WriteLine($"Время начала: {tutorial["start"]}");

Console.WriteLine($"Время конца: {tutorial["end"]}");

Console.WriteLine($"Тип занятия: {tutorial["type"]}\n");

}

Console.WriteLine("----\n");

}

Console.ReadLine();

}

public void AllLessons()

{

Console.WriteLine("Все занятия:");

int i = 1;

foreach (var tutorial in timesheetJson.SelectTokens("$.timesheet.day[\*].tutorial[\*]"))

{

string subject = tutorial["subject"].ToString();

Console.WriteLine($"{i}. {subject}");

i++;

}

Console.ReadLine();

}

public void AllRooms()

{

HashSet<string> uniqueRooms = new HashSet<string>();

foreach (var room in timesheetJson.SelectTokens("$.timesheet.day[\*].tutorial[\*].room"))

{

uniqueRooms.Add(room.ToString());

}

Console.WriteLine("Аудитории:");

foreach (string room in uniqueRooms)

{

Console.WriteLine($"\* {room}");

}

}

public void AllPractice()

{

Console.WriteLine("Все практики:");

foreach (var tutorial in timesheetJson.SelectTokens("$.timesheet.day[\*].tutorial[?(@.type=='практика')]"))

{

Console.WriteLine($"- {tutorial["subject"]}");

}

}

public void AllLectureInRoom()

{

Console.Write("Номер аудитории: ");

string room = Console.ReadLine();

Console.WriteLine($"Лекции в аудитории {room}:");

var lectures = timesheetJson.SelectTokens($"$.timesheet.day[\*].tutorial[?(@.room == '{room}' && @.type == 'лекция')]");

if (lectures.Any())

{

foreach (var lecture in lectures)

{

Console.WriteLine($"\* {lecture["subject"]}");

}

}

else

{

Console.WriteLine("Лекций в данной аудитории нет");

}

}

public void AllTeachersPracticeInRoom()

{

Console.Write("Номер аудитории: ");

string room = Console.ReadLine();

Console.WriteLine($"Преподаватели, проводящие практики в аудитории номер {room}:");

var tutorials = timesheetJson.SelectTokens($"$.timesheet.day[\*].tutorial[?(@.room == '{room}' && @.type == 'практика')]");

if (tutorials.Any())

{

foreach (var tutorial in tutorials)

{

Console.WriteLine($"- {tutorial["teacher"]}");

}

}

else

{

Console.WriteLine($"В данной аудитории нет практики");

}

}

public void LastLesson()

{

Console.WriteLine("Последние занятия:");

foreach (JToken day in timesheetJson["timesheet"]["day"])

{

JArray tutorials = (JArray)day["tutorial"];

var lastLesson = tutorials[tutorials.Count - 1];

Console.WriteLine(day["name"] + ": " + lastLesson["subject"]);

}

}

public void TotalCountLessons()

{

int totalLessons = GetTotalLessons(timesheetJson);

Console.WriteLine($"Общее количество занятий: {totalLessons}");

static int GetTotalLessons(JObject timesheet)

{

var allLessons = timesheet.SelectTokens("$.timesheet.day[\*].tutorial[\*].subject");

return allLessons.Count();

}

}