JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Könyvesboltok nyilvántartása

Készítette: **Rontó Eszter**

Neptun kód: **QTFL19**

Dátum: **2023.11.28**

Tartalomjegyzék

1. Bevezetés (feladat leírása)

A beadandóm témája egy olyan adatbázis, amely egy könyvesbolt “láncnak” a boltjait tartja számon. Megnézhetjük, hogy mennyi könyvet tudnak tárolni és hogy milyen könyveik vannak jelenleg eladásra kiadva, melyik kiadó által. Emellett tudjuk, hogy melyik vendég melyik könyvet vette meg és hogy milyen kártyát használt az olcsóbb vásárlás érdekében.

> Könyvesbolt

- KB_ID: A könyvesbolt egyed elsődleges kulcsa
- Név: A Könyvesbolt neve
- Elérhetőség: A Könyvesbolt elérhetőségei (összetett tulajdonság)
- Kapacitás: A Könyvesboltban tárolható könyvek mennyisége

> Könyv

- K_ID: A Könyv egyed elsődleges kulcsa
- Cím: A Könyv címe
- Műfaj: A Könyv műfaja/i (többértékű tulajdonság)
- Író: A Könyv írójának neve

> Kiadó

- Adószám: A Kiadó egyed elsődleges kulcsa
- Név: A Kiadó neve
- Alapítás: A Kiadó alapításának dátuma
- Hely: A Kiadó címe

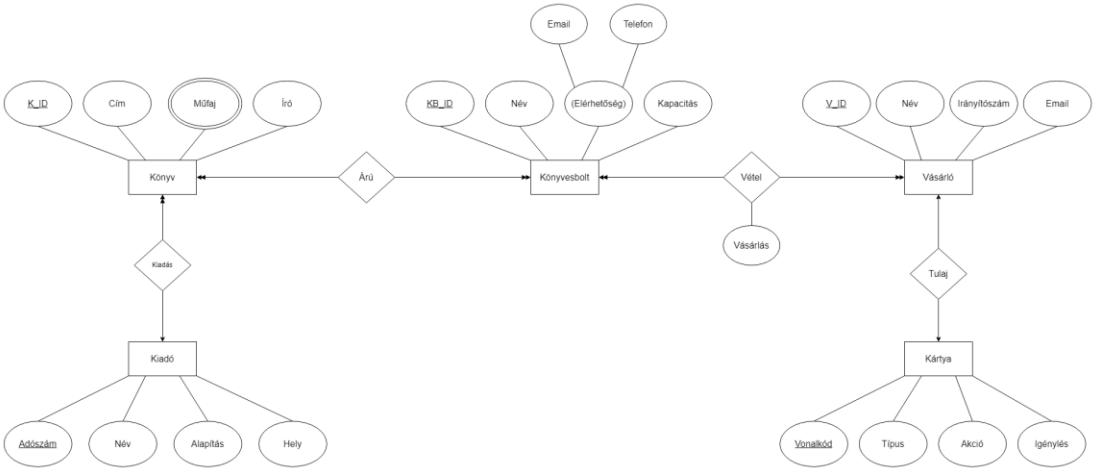
> Vásárló

- V_ID: A Vásárló egyed elsődleges kulcsa
- Név: A Vásárló neve
- Irányítószám: A Vásárló irányítószáma
- Email: A Vásárló email címe

> Kártya

- Vonalkód: A Kártya egyed elsődleges kulcsa
- Típus: A Kártya típusa (pl.: törzsvásárlói / ajándék)
- Akció: A Kártya által kapott akció értéke
- Igénylés: A Kártya igénylésének dátuma

1a) ER modell



Kapcsolatok

Kiadó és Könyv:

Egy több kapcsolat áll fent köztük, mivel egy Kiadónak lehet több Könyve, viszont egy Könyvnek csak egy Kiadója lehet.

Könyv és Könyvesbolt:

Több több kapcsolat áll fent köztük, mivel egy Könyv meg jelenhet több Könyvesboltban is és egy Könyvesboltban több Könyv is fellelhető.

Könyvesbolt és Vevő:

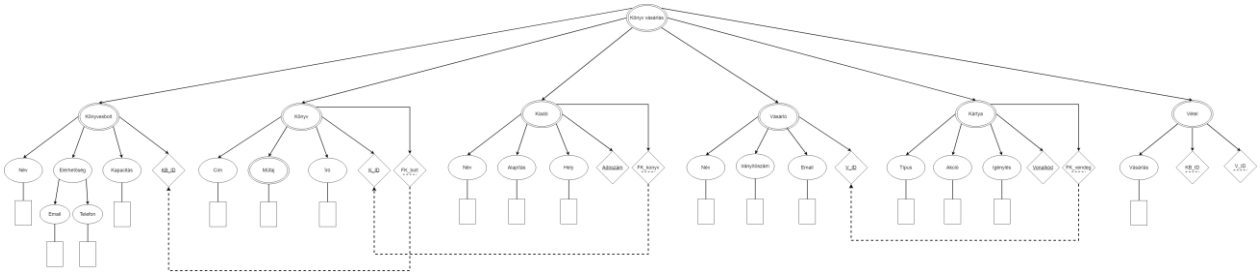
Több több kapcsolat áll fent köztük, mivel egy Könyvesboltban több Vásárló is vásárolhat és egy Vásárló több Könyvesboltban is vásárolhat.

Vásárló és Kártya:

Egy egy kapcsolat áll fent köztük, mivel egy Vásárlónak csak egy Kártyája lehet és egy Kártyának csak egy (Vevő) tulajdonosa lehet.

1b) ER modell konvertálása XDM modellre

Az XDM modellnél az elemeket ellipszissel jelöljük ezek az elemek az egyedek és azok tulajdonságai, a téglalapok jelölik, hogy a tulajdonságnak van szövege. (Ezek a szövegek majd csak az XML dokumentumban fognak megjelenni)
Az attribútumokat / kulcsokat rombusszal jelöljük és aláhúzzuk a nevüket. A többértékű tulajdonsággal rendelkező elem dupla ellipszissel tűnik ki a többi közül. A kulcsok és idegenkulcsok közötti kapcsolat szaggatott vonallal jelöljük.



1c) XML dokumentum készítése

Az XML dokumentum a root elementtel kezdődött, ami ez esetben a konyvesvasarlas volt. Ezek után elkészítettem a gyerekelemeket mindből legalább 3 példányt ezeknek az elemeknek az attribútumai közé tartoznak a kulcsok és az idegenkulcsok. Végül ezek az elemek lettek a szülő elemek és létrehoztam ezeknek a gyerek elemeit. A többértékű elemeknél több elemet is létrehoztam, illetve az összetett elemnél még ezeknek az elemeknek is lett gyerekeleme.

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

<konyvesvasarlas xmlns:xs="http://www.w3.org/2001/XMLSchema-
instance" xs:noNamespaceSchemaLocation="XMLSchemaQTFL19.xsd">



    <!-- Könyvesbolt -->
    <!-- Könyvesbolt "lánc" neve : LAP -->
    <konyvesbolt KB_ID="kb1">
        <nev>LAPos</nev>
        <elerhetoseg>
            <email>lapos@gmail.com</email>
            <telefon>06305984652</telefon>
        </elerhetoseg>
        <kapacitas>19.020</kapacitas>
    </konyvesbolt>

    <konyvesbolt KB_ID="kb2">
        <nev>aLAP</nev>
        <elerhetoseg>
            <email>alap@freemail.hu</email>
```

```
        <telefon>06201523495</telefon>
    </elerhetoseg>
    <kapacitas>45.000</kapacitas>
</konyvesbolt>

<konyvesbolt KB_ID="kb3">
    <nev>LAPoz</nev>
    <elerhetoseg>
        <email>lapoz@gmail.hu</email>
        <telefon>06708623562</telefon>
    </elerhetoseg>
    <kapacitas>20.900</kapacitas>
</konyvesbolt>



<!-- Könyv -->
<konyv K_ID="k1" FK_bolt="kb3">
    <cim>Oppa és Yeobo</cim>
    <mufaj>dráma</mufaj>
    <mufaj>romantikus</mufaj>
    <mufaj>komédia</mufaj>
    <iro>Tóth Péter Tamás</iro>
</konyv>

<konyv K_ID="k2" FK_bolt="kb3">
    <cim>Bankrablás</cim>
    <mufaj>thriller</mufaj>
    <mufaj>krimi</mufaj>
    <mufaj>akció</mufaj>
    <iro>Szabó János</iro>
</konyv>

<konyv K_ID="k3" FK_bolt="kb2">
    <cim>Shuriken</cim>
    <mufaj>akció</mufaj>
    <mufaj>tragédia</mufaj>
    <mufaj>kalandregény</mufaj>
    <iro>Bogyó Eszter</iro>
</konyv>

<konyv K_ID="k4" FK_bolt="kb1">
    <cim>Egyetem</cim>
    <mufaj>thriller</mufaj>
    <mufaj>horror</mufaj>
    <iro>Garamszegi Márton</iro>
</konyv>



<!-- Kiadó -->
```

```
<kiado Adoszam="1234567895" FK_konyv="k3">
    <nev>Pub</nev>
    <alapitas>1980.10.21</alapitas>
    <hely>California</hely>
</kiado>

<kiado Adoszam="1578623598" FK_konyv="k1">
    <nev>Money</nev>
    <alapitas>2005.09.13</alapitas>
    <hely>Bikinifenék</hely>
</kiado>

<kiado Adoszam="1452368952" FK_konyv="k4">
    <nev>OneTrickPony</nev>
    <alapitas>2010.02.30</alapitas>
    <hely>New York</hely>
</kiado>

<kiado Adoszam="1982332959" FK_konyv="k2">
    <nev>KiAD</nev>
    <alapitas>1996.05.19</alapitas>
    <hely>Budapest</hely>
</kiado>



<!-- Vásárló -->
<vasarlo V_ID="v1">
    <nev>Súkeník Erik</nev>
    <iranyitoszam>8596</iranyitoszam>
    <email>EriKing@gmail.com</email>
</vasarlo>

<vasarlo V_ID="v2">
    <nev>Szabó Dávid</nev>
    <iranyitoszam>1045</iranyitoszam>
    <email>distrect@citromail.hu</email>
</vasarlo>

<vasarlo V_ID="v3">
    <nev>Magyar János</nev>
    <iranyitoszam>1139</iranyitoszam>
    <email>makkos98@outlook.hu</email>
</vasarlo>

<vasarlo V_ID="v4">
    <nev>Bogyó Márta</nev>
    <iranyitoszam>9562</iranyitoszam>
    <email>bogyo@gmail.com</email>
</vasarlo>
```

```
<vasarlo V_ID="v5">
    <nev>Morty Smith</nev>
    <iranyitoszam>6243</iranyitoszam>
    <email>smith@gmail.com</email>
</vasarlo>



<!-- Kártya -->
<kartya vonalkod="45236874" FK_vendeg="v1">
    <tipus>Törzsvársálói</tipus>
    <akcio>15</akcio>
    <igenyles>2015.04.28</igenyles>
</kartya>

<kartya vonalkod="85632982" FK_vendeg="v2">
    <tipus>Törzsvársálói</tipus>
    <akcio>90</akcio>
    <igenyles>1980.09.10</igenyles>
</kartya>

<kartya vonalkod="65369848" FK_vendeg="v5">
    <tipus>Ajándék</tipus>
    <akcio>30</akcio>
    <igenyles>2020.12.20</igenyles>
</kartya>

<kartya vonalkod="53248725" FK_vendeg="v4">
    <tipus>Törzsvársálói</tipus>
    <akcio>30</akcio>
    <igenyles>1998.06.20</igenyles>
</kartya>

<kartya vonalkod="35986147" FK_vendeg="v3">
    <tipus>Ajándék</tipus>
    <akcio>30</akcio>
    <igenyles>2023.08.02</igenyles>
</kartya>



<!-- Vétel kapcsolat -->
<vetel KB_ID="kb1" V_ID="v3">
    <vasarlas>2018.11.04</vasarlas>
</vetel>

<vetel KB_ID="kb2" V_ID="v1">
    <vasarlas>2022.03.15</vasarlas>
</vetel>

<vetel KB_ID="kb3" V_ID="v2">
```

```
        <vasarlas>2023.08.06</vasarlas>
    </vetel>

    <vetel KB_ID="kb2" V_ID="v4">
        <vasarlas>2023.04.23</vasarlas>
    </vetel>

    <vetel KB_ID="kb1" V_ID="v5">
        <vasarlas>2020.10.08</vasarlas>
    </vetel>

    <vetel KB_ID="kb3" V_ID="v4">
        <vasarlas>2021.07.10</vasarlas>
    </vetel>


</konyvesvasarlas>
```

1d) XMLSchema készítése

Annak érdekében, hogy a későbbiekben a felépítésben lehessen rájuk hivatkozni így az egyszerű elemekkel kezdtem majd következtek a hozzájuk tartozó saját típusok (négy darab). Ezek után létrehoztam a felépítést, amiben meg adtam a minimum és maximum előfordulást is majd az attribútumokat is meg adtam, amikhez végül létrehoztam a kulcs, idegenkulcs és az 1:1 különleges kapcsolat referenciákat is.

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

    <!-- Egyszerű Elemek -->

    <!-- könyvesbolt elemek -->
    <xs:element name="nev" type="xs:string"/>
    <xs:element name="email" type="emailType"/>
    <xs:element name="telefon" type="xs:integer"/>
    <xs:element name="kapacitas" type="kapType"/>

    <!-- könyv elemek -->
    <xs:element name="cim" type="xs:string"/>
    <xs:element name="mufaj" type="xs:string"/>
    <xs:element name="iro" type="xs:string"/>

    <!-- kiadó elemek  -->
    <xs:element name="alapitas" type="dateType"/>
    <xs:element name="hely" type="xs:string"/>

    <!-- vásárló elemek -->
    <xs:element name="iranyitoszam" type="iranyType"/>
```

```
    <!-- kártya elemek -->
    <xs:element name="tipus" type="xs:string"/>
    <xs:element name="akcio" type="xs:integer"/>
    <xs:element name="igenyles" type="dateType"/>

    <!-- vétel kapcsolat eleme -->
    <xs:element name="vasarlas" type="dateType"/>



    <!-- Saját Típusok -->
    <xs:simpleType name="emailType">
        <xs:restriction base="xs:string">
            <xs:pattern value="[^@]+@[^\.]+\..+"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="iranyType">
        <xs:restriction base="xs:string">
            <xs:pattern value="[0-9][0-9][0-9][0-9]"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="kapType">
        <xs:restriction base="xs:string">
            <xs:pattern value="[0-9][0-9].[0-9][0-9][0-9]"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="dateType">
        <xs:restriction base="xs:string">
            <xs:pattern value="([12]\d{3}.(0[1-9]|1[0-2]).(0[1-
9]|[12]\d|3[01]))"/>
        </xs:restriction>
    </xs:simpleType>



    <!-- Felépítés -->
    <xs:element name="konyvesvasarlas">
        <xs:complexType>
            <xs:sequence>

                <!-- könyvesbolt -->
                <xs:element name="konyvesbolt" minOccurs="1"
maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element ref="nev"/>
                            <xs:element name="elerhetoseg">
                                <xs:complexType>
```

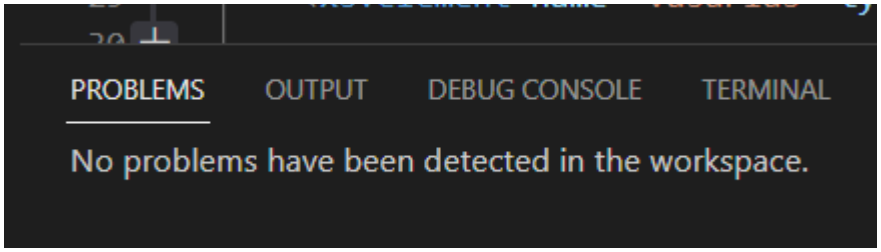
```
                                    <xs:sequence>
                                        <xs:element ref="email"/>
                                        <xs:element ref="telefon"/>
                                    </xs:sequence>
                                </xs:complexType>
                            </xs:element>
                            <xs:element ref="kapacitas"/>
                        </xs:sequence>
                        <xs:attribute name="KB_ID"
type="xs:string"/>
                    </xs:complexType>
                </xs:element>

                <!-- könyv -->
                <xs:element name="konyv" minOccurs="1"
maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element ref="cim"/>
                            <xs:element ref="mufaj" minOccurs="1"
maxOccurs="unbounded"/>
                            <xs:element ref="iro"/>
                        </xs:sequence>
                        <xs:attribute name="K_ID" type="xs:string"/>
                        <xs:attribute name="FK_bolt"
type="xs:string"/>
                    </xs:complexType>
                </xs:element>

                <!-- kiadó -->
                <xs:element name="kiado" minOccurs="1"
maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element ref="nev"/>
                            <xs:element ref="alapitas"/>
                            <xs:element ref="hely"/>
                        </xs:sequence>
                        <xs:attribute name="Adoszam" type="xs:int"/>
                        <xs:attribute name="FK_konyv"
type="xs:string"/>
                    </xs:complexType>
                </xs:element>

                <!-- vásárló -->
                <xs:element name="vasarlo" minOccurs="1"
maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element ref="nev"/>
                            <xs:element ref="iranyitoszam"/>
```

```
                            <xs:element ref="email"/>
                        </xs:sequence>
                        <xs:attribute name="V_ID" type="xs:string"/>
                    </xs:complexType>
                </xs:element>

                <!-- kártya -->
                <xs:element name="kartya" minOccurs="1"
maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element ref="tipus"/>
                            <xs:element ref="akcio"/>
                            <xs:element ref="igenyles"/>
                        </xs:sequence>
                        <xs:attribute name="vonalkod"
type="xs:int"/>
                        <xs:attribute name="FK_vendeg"
type="xs:string"/>
                    </xs:complexType>
                </xs:element>

                <!-- vétel kapcsolat -->
                <xs:element name="vetel" minOccurs="1"
maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element ref="vasarlas"/>
                        </xs:sequence>
                        <xs:attribute name="KB_ID"
type="xs:string"/>
                        <xs:attribute name="V_ID" type="xs:string"/>
                    </xs:complexType>
                </xs:element>

            </xs:sequence>
        </xs:complexType>




        <!--Kulcsok-->
        <xs:key name="konyvesbolt_kulcs">
            <xs:selector xpath="konyvesbolt"/>
            <xs:field xpath="@KB_ID"/>
        </xs:key>

        <xs:key name="konyv_kulcs">
            <xs:selector xpath="konyv"/>
            <xs:field xpath="@K_ID"/>
        </xs:key>
```

```
        <xs:key name="kiado_kulcs">
            <xs:selector xpath="kiado"/>
            <xs:field xpath="@Adoszam"/>
        </xs:key>

        <xs:key name="vasarlo_kulcs">
            <xs:selector xpath="vasarlo"/>
            <xs:field xpath="@V_ID"/>
        </xs:key>

        <xs:key name="kartya_kulcs">
            <xs:selector xpath="kartya"/>
            <xs:field xpath="@vonalkod"/>
        </xs:key>



        <!-- Idegen kulcsok -->
        <xs:keyref refer="konyvesbolt_kulcs"
name="konyvesbolt_idegen_kulcs">
            <xs:selector xpath="konyv"/>
            <xs:field xpath="@FK_bolt"/>
        </xs:keyref>

        <xs:keyref refer="konyv_kulcs" name="konyv_idegen_kulcs">
            <xs:selector xpath="kiado"/>
            <xs:field xpath="@FK_konyv"/>
        </xs:keyref>

        <xs:keyref refer="konyvesbolt_kulcs"
name="vetel_konyvesbolt_idegen_kulcs">
            <xs:selector xpath="vetel"/>
            <xs:field xpath="@KB_ID"/>
        </xs:keyref>

        <xs:keyref refer="vasarlo_kulcs"
name="vetel_vasarlo_idegen_kulcs">
            <xs:selector xpath="vetel"/>
            <xs:field xpath="@V_ID"/>
        </xs:keyref>



        <!-- 1:1 -->
        <xs:unique name="unique_kartya">
            <xs:selector xpath="kartya"/>
            <xs:field xpath="@FK_vendeg"/>
        </xs:unique>
```

```
    </xs:element>
</xs:schema>
```

Sikeres validálás :



2. DOM program készítése

2a) adatolvasás

1. Először létrehoztam egy kimeneti fájlt (DomReadQTFL19.xml), amibe az eredményt fogja kiírni a kód. Majd beolvassa a bemeneti XML fájlt (XMLQTFL19.xml) egy DOM (Document Object Model) fájlba, és normalizálja azt. A kódban különböző XML elemeket olvastattam be csoportok szerint és ezek attribútumait, valamint gyermek elemeit struktúrált formában írja ki az új fájlba és a konzolra (már ha ki van kommentelve egy adott program kód rész). A ciklusokon keresztül az összes elemet és azok tartalmát olvassa be, ellenőrzi a gyermek elemek számát végül bezárja az új fájlt és kezeli az esetleges kivételeket is, amelyek az olvasás és írás során adódhatnak.

```
package hu.domparse.QTFL19;

import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.io.PrintStream;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

public class DomReadQTFL19 {
```

```
    public static void main(String argv[]) throws SAXException,
IOException, ParserConfigurationException {

try {
/* Új fájl */
File OutPutRead = new File("DomReadQTFL19.xml");
OutputStream os = new FileOutputStream(OutPutRead);
PrintStream ps = new PrintStream(os);

/* A konzol kimenetének átirányítása az új fájlba */
System.setOut(ps);

File xmlFile = new File("XMLQTFL19.xml");

/* Dokumentum Builder létrekozása */
DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
DocumentBuilder dBuilder = factory.newDocumentBuilder();
Document doc = dBuilder.parse(xmlFile);
doc.getDocumentElement().normalize();

System.out.println("<?xml version=\"1.0\" encoding=\"UTF-8\"?>");
System.out.println("\n<" + doc.getDocumentElement().getNodeName() +
">");


/* A könyvesbolt beolvasása */
NodeList kbList = doc.getElementsByTagName("konyvesbolt");
for (int i = 0; i < kbList.getLength(); i++) {

Node nNode = kbList.item(i);
if (nNode.getNodeType() == Node.ELEMENT_NODE) {

Element elem = (Element) nNode;
String kbid = elem.getAttribute("KB_ID");

Node node1 = elem.getElementsByTagName("nev").item(0);
String kbnev = node1.getTextContent();

/* a node2 lesz az elérhetőség */

Node node3 = elem.getElementsByTagName("kapacitas").item(0);
String kap = node3.getTextContent();

/* struktúrált ki iratás */
System.out.println("\n\t<" + nNode.getNodeName() + " KB_ID=" + kbid
+ ">");
System.out.println("\t  <nev>" + kbnev + "</nev>");

if (kbList.item(i).getChildNodes().getLength() > 3) {
int db = 0;
```

```
Node node2 = elem.getElementsByTagName("elerhetoseg").item(0);
while (node2 != null) {
node2 = elem.getElementsByTagName("elerhetoseg").item(db);
if (node2 != null) {
Node n = elem.getElementsByTagName("email").item(db);
String em = n.getTextContent();
System.out.println("\t    <email>" + em + "</email>");

Node n2 = elem.getElementsByTagName("telefon").item(db);
String tel = n2.getTextContent();
System.out.println("\t    <telefon>" + tel + "</telefon>");
}
db++;
}
}
System.out.println("\t  <kapacitas>" + kap + "</kapacitas>");
}
System.out.println("\t</" + nNode.getNodeName() + ">");
}



/* A könyv elem beolvasása */
NodeList kList = doc.getElementsByTagName("konyv");
for (int i = 0; i < kList.getLength(); i++) {

Node nNode = kList.item(i);
if (nNode.getNodeType() == Node.ELEMENT_NODE) {

Element elem = (Element) nNode;
String kid = elem.getAttribute("K_ID");

Node node1 = elem.getElementsByTagName("cim").item(0);
String cim = node1.getTextContent();

/* A node2 itt a mufaj */

Node node3 = elem.getElementsByTagName("iro").item(0);
String iro = node3.getTextContent();

/* struktúrált ki iratás */
System.out.println("\n\t<" + nNode.getNodeName() + " K_ID = " + kid
+ ">");
System.out.println("\t  <cim>" + cim + "</cim>");

if (kList.item(i).getChildNodes().getLength() > 5) {
int db = 0;
Node node2 = elem.getElementsByTagName("mufaj").item(0);
while (node2 != null) {
node2 = elem.getElementsByTagName("mufaj").item(db);
if (node2 != null) {
```

```
String rh = node2.getTextContent();
System.out.println("\t  <mufaj>" + rh + "</mufaj>");
}
db++;
}
}

System.out.println("\t  <iro>" + iro + "</iro>");
}
System.out.println("\t</" + nNode.getNodeName() + ">");
}



/* Kiadó elem beolvasása */
NodeList kiList = doc.getElementsByTagName("kiado");
for (int i = 0; i < kiList.getLength(); i++) {

Node nNode = kiList.item(i);
if (nNode.getNodeType() == Node.ELEMENT_NODE) {

Element elem = (Element) nNode;
String ado = elem.getAttribute("Adoszam");

Node node1 = elem.getElementsByTagName("nev").item(0);
String kinev = node1.getTextContent();

Node node2 = elem.getElementsByTagName("alapitas").item(0);
String alap = node2.getTextContent();

Node node3 = elem.getElementsByTagName("hely").item(0);
String hely = node3.getTextContent();

/* struktúrált ki iratás */
System.out.println("\n\t<" + nNode.getNodeName() + " Adoszam = " +
ado + ">");
System.out.println("\t  <nev>" + kinev + "</nev>");
System.out.println("\t  <alapitas>" + alap + "</alapitas>");
System.out.println("\t  <hely>" + hely + "</hely>");
}
System.out.println("\t</" + nNode.getNodeName() + ">");
}


/* Vásárló egyed beolvasása */
NodeList vList = doc.getElementsByTagName("vasarlo");
for (int i = 0; i < vList.getLength(); i++) {

Node nNode = vList.item(i);
if (nNode.getNodeType() == Node.ELEMENT_NODE) {
```

```
Element elem = (Element) nNode;
String vid = elem.getAttribute("V_ID");

Node node1 = elem.getElementsByTagName("nev").item(0);
String vnev = node1.getTextContent();

Node node2 = elem.getElementsByTagName("iranyitoszam").item(0);
String irany = node2.getTextContent();

Node node3 = elem.getElementsByTagName("email").item(0);
String vem = node3.getTextContent();

/* struktúrált ki iratás */
System.out.println("\n\t<" + nNode.getNodeName() + " V_ID = " + vid
+ ">");
System.out.println("\t  <nev>" + vnev + "</nev>");
System.out.println("\t  <iranyitoszam>" + irany +
"</iranyitoszam>");
System.out.println("\t  <email>" + vem + "</email>");
}
System.out.println("\t</" + nNode.getNodeName() + ">");
}


/* Kártya egyed beolvasása */
NodeList kaList = doc.getElementsByTagName("kartya");
for (int i = 0; i < kaList.getLength(); i++) {

Node nNode = kaList.item(i);
if (nNode.getNodeType() == Node.ELEMENT_NODE) {

Element elem = (Element) nNode;
String von = elem.getAttribute("vonalkod");

Node node1 = elem.getElementsByTagName("tipus").item(0);
String tip = node1.getTextContent();

Node node2 = elem.getElementsByTagName("akcio").item(0);
String akcio = node2.getTextContent();

Node node3 = elem.getElementsByTagName("igenyles").item(0);
String ig = node3.getTextContent();

/* struktúrált ki iratás */
System.out.println("\n\t<" + nNode.getNodeName() + " vonalkod = " +
von + ">");
System.out.println("\t  <tipus>" + tip + "</tipus>");
System.out.println("\t  <akcio>" + akcio + "</akcio>");
System.out.println("\t  <igenyles>" + ig + "</igenyles>");
}
System.out.println("\t</" + nNode.getNodeName() + ">");
```

```
}


/* vétel kapcsolat */
NodeList vetList = doc.getElementsByTagName("vetel");
for (int i = 0; i < vetList.getLength(); i++) {

Node nNode = vetList.item(i);
if (nNode.getNodeType() == Node.ELEMENT_NODE) {

Element elem = (Element) nNode;
String kbid = elem.getAttribute("KB_ID");
String vid = elem.getAttribute("V_ID");

Node node1 = elem.getElementsByTagName("vasarlas").item(0);
String vas = node1.getTextContent();

/* struktúrált ki iratás */
System.out.println("\n\t<" + nNode.getNodeName() + " KB_ID = " +
kbid + " " + " V_ID = " + vid + ">");
System.out.println("\t  <vasarlas> " + vas + "</vasarlas>");

}
System.out.println("\t</" + nNode.getNodeName() + ">");
}
System.out.println("\n</" + doc.getDocumentElement().getNodeName() +
">");



/* Új fájl bezárása */
ps.close();


}catch (Exception e) {
            e.printStackTrace();
        }
}
}
```

2. A második próbálkozásomnál, egy XML fájlt olvass be, majd átalakítja a fájlt egy jól struktúrált formátumra, és mentse azt egy új XML fájlba. Először beolvassa az XMLQTFL19.xml nevű XML fájlt egy Document objektumba, és normalizálja azt. Majd egy StringWriter segítségével átalakítja az XML dokumentumot egy string formátumba. Aztán létrehoz egy Transformer objektumot, beállítja a formázást (OutputKeys.INDENT), és átalakítja az XML-t a struktúrált formátumba. Az így kapott XML stringet egy új fájlba menti (DomReadQTFL19.xml) a FileOutputStream és OutputStreamWriter segítségével. Végül kiírja az XML stringet a konzolra is a System.out.println(xmlString) segítségével.

```
package hu.domparse.QTFL19;

import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStreamWriter;
import java.io.StringWriter;
import java.io.Writer;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.transform.OutputKeys;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;

import org.w3c.dom.Document;
import org.xml.sax.SAXException;

public class DomReadQTFL19_2 {

    public static void main(String argv[]) throws SAXException,
IOException, ParserConfigurationException {

try {
File xmlFile = new File("XMLQTFL19.xml");

            /* Dokumentum Builder létrekozása */
            DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
            DocumentBuilder dBuilder = factory.newDocumentBuilder();
            Document doc = dBuilder.parse(xmlFile);
            doc.getDocumentElement().normalize();

            StringWriter sw = new StringWriter();
            Transformer t =
TransformerFactory.newInstance().newTransformer();
            t.setOutputProperty(OutputKeys.OMIT_XML_DECLARATION,
"yes");
            t.setOutputProperty(OutputKeys.INDENT, "yes");
            t.transform(new DOMSource(doc), new StreamResult(sw));
            String xmlString = sw.toString();

            /* XmlString to File */
            Writer output = null;
            File file = new File("DomReadQTFL19.xml");
            output = new OutputStreamWriter(new
FileOutputStream(file));
```

```
            output.write(xmlString);
            output.close();

            System.out.println(xmlString);

}catch (Exception e) {
            e.printStackTrace();
}


}
}
```

2b) adatmódosítás

Beolvassa a XMLQTFL19.xml nevű XML fájlt egy Document objektumba majd normalizálja azt. Igazából a kódom a különböző XML elemeket és azok attribútumait vagy tartalmát módosítja a megadott feltételeknek megfelelően, majd ezekről információt nyújt a konzolra, hogy követhető legyen, mi történt az adatokkal.

```
package hu.domparse.QTFL19;

import java.io.File;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;

public class DomModifyQTFL19 {

    public static void main(String argv[]) {

        try {
            File xmlFile = new File("XMLQTFL19.xml");

            /* Dokumentum Builder létrekozása */
            DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
DocumentBuilder dBuilder = factory.newDocumentBuilder();
            Document doc = dBuilder.parse(xmlFile);
doc.getDocumentElement().normalize();


/* 1. Könyvesbolt kapacitásának cseréja HA az attribútuma kb1 */
            NodeList kbNodeList =
doc.getElementsByTagName("konyvesbolt");
for (int i=0; i<kbNodeList.getLength(); i++) {
```

```
                Element kbNode = (Element) kbNodeList.item(i);
String kbAttr = kbNode.getAttribute("KB_ID");

if (kbAttr.equals("kb1")) {
kbNode.getElementsByTagName("kapacitas").item(0).setTextContent("20.
400");
System.out.print("1. Módosított kapacitás : " +
kbNode.getElementsByTagName("kapacitas").item(0).getTextContent());
}
            }

/* 2. Kiadó ahol 10 karakternél rövidebb a hely hossza ott a hely
mellé + (új hely szükséges)  */
NodeList kiNodeList = doc.getElementsByTagName("kiado");
for(int i=0; i<kiNodeList.getLength(); i++) {
Element kiNode = (Element) kiNodeList.item(i);
String ki_hely =
kiNode.getElementsByTagName("hely").item(0).getTextContent();

if (ki_hely.length() < 10) {
kiNode.getElementsByTagName("hely").item(0).setTextContent(ki_hely +
" (új hely szükséges)");
System.out.print("\n\n2. Új hely:" +
kiNode.getElementsByTagName("hely").item(0).getTextContent());
}
}

/* 3. Megváltoztatja az első könyv címét */
NodeList kNodeList = doc.getElementsByTagName("konyv");
Element kNode = (Element) kNodeList.item(0);
kNode.getElementsByTagName("cim").item(0).setTextContent("League of
Legends");
System.out.print("\n\n3. Új cím:" +
kNode.getElementsByTagName("cim").item(0).getTextContent());

/* 4. A vásárlóknak akinek van á a nevükben meg változtatja a
nevüket + szöveggel*/
NodeList vNodeList = doc.getElementsByTagName("vasarlo");
for(int i=0; i<vNodeList.getLength(); i++) {
Element vNode = (Element) vNodeList.item(i);
String v_nev =
vNode.getElementsByTagName("nev").item(0).getTextContent();

if (v_nev.matches(".+á.+")) {
vNode.getElementsByTagName("nev").item(0).setTextContent(v_nev + "
van egy á a nevemben");
System.out.print("\n\n4. Van á?:" +
vNode.getElementsByTagName("nev").item(0).getTextContent());
}
}
```

```
/* 5. Az első három kártya kivételével mindnek meg cseréli a tipusát
Ajándékra  */
NodeList kaNodeList = doc.getElementsByTagName("kartya");
for(int i=0; i<kaNodeList.getLength(); i++) {
Element kaNode = (Element) kaNodeList.item(i);

if (i > 2) {
kaNode.getElementsByTagName("tipus").item(0).setTextContent("Ajandek
");
System.out.print("\n\n5. Új tipus: " +
kaNode.getElementsByTagName("tipus").item(0).getTextContent());
}
}

} catch (Exception e) {
e.printStackTrace();
}



    }
}
```

2c) adatkérdezés

Beolvassa a XMLQTFL19.xml nevű XML fájlt egy Document objektumba és normalizálja azt. A kódom az egyes elemeket lekérdezi és a feltételeknek megfelelően információt jelenít meg a konzolon, lehetővé téve az XML fájlban szereplő adatok szűrését és listázását a programban definiált kritériumok alapján.

```
package hu.domparse.QTFL19;

import java.io.File;
import java.util.ArrayList;
import java.util.List;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;


public class DomQueryQTFL19 {

public static void main(String[] args) {
```

```
        List<String> kb_lista = new ArrayList<>(); /* könyvesbolt
lista */
        List<String> k_lista = new ArrayList<>(); /* könyv lista */
        List<String> v_lista = new ArrayList<>(); /* vasarlo lista
*/
        List<String> ki_lista = new ArrayList<>(); /* kiado lista */
        List<String> kar_lista = new ArrayList<>(); /* kiado lista
*/

        try{
            File xmlFile = new File("XMLQTFL19.xml");

            /* Dokumentum Builder létrekozása */
            DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
DocumentBuilder dBuilder = factory.newDocumentBuilder();
            Document doc = dBuilder.parse(xmlFile);
            doc.getDocumentElement().normalize();

            /* 1. Lekérdezi a könyvesboltok nevét és kapacitását
azoknak ahol az email címében található gmail rész */
            NodeList kbNodeList =
doc.getElementsByTagName("konyvesbolt");
            System.out.print("\n1. Könyvesboltok gmail-el
lekérdezés");

            for (int i=0; i<kbNodeList.getLength(); i++) {
                Element kbElement = (Element) kbNodeList.item(i);
                String kb = kbElement.getTextContent();
                kb_lista.add(kb);

                String kb_email =
kbElement.getElementsByTagName("email").item(0).getTextContent();

                if (kb_email.matches(".+@gmail.+")){
                    System.out.print("\n" +
kbElement.getElementsByTagName("nev").item(0).getTextContent() + " :
" +
kbElement.getElementsByTagName("kapacitas").item(0).getTextContent()
+ "db");
                }
            }

            /* 2. Lekérdezi az összes könyv címét és íróját */
            NodeList kNodeList = doc.getElementsByTagName("konyv");
            System.out.print("\n\n\n2. Összes könyv lekérdezése
(cím, író)");

            for (int i=0; i<kNodeList.getLength(); i++) {
                Element kElement = (Element) kNodeList.item(i);
                String k_sum = kElement.getTextContent();
```

```
                k_lista.add(k_sum);

                System.out.print("\nKönyv címe : " +
kElement.getElementsByTagName("cim").item(0).getTextContent());
                System.out.print("\t írója : " +
kElement.getElementsByTagName("iro").item(0).getTextContent());
            }

            /* 3. Lekérdezi egy konkrét vásárló adatait (Súkeník
Erik) */
            NodeList vNodeList =
doc.getElementsByTagName("vasarlo");
            System.out.print("\n\n\n3. Súkeník Erik adatainak
lekérdezése");

            for (int i=0; i<vNodeList.getLength(); i++) {
                Element vElement = (Element) vNodeList.item(i);
                String v_sum = vElement.getTextContent();
                v_lista.add(v_sum);

                String v_name =
vElement.getElementsByTagName("nev").item(0).getTextContent();

                if (v_name.equals("Súkeník Erik")) {
                    System.out.print("\nID : " +
vElement.getAttribute("V_ID") + "\nIrányítószáma : " +
vElement.getElementsByTagName("iranyitoszam").item(0).getTextContent
() + "\nEmail címe : " +
vElement.getElementsByTagName("email").item(0).getTextContent());
                }
            }

            /* 4. Lekérdezi az összes kiadó nevét, alapítási dátumát
és helyét ahol az alapítás 2000 előtt történt */
            NodeList kiNodeList = doc.getElementsByTagName("kiado");
            System.out.print("\n\n\n4. Kiadó alapítás lekérdezés");

            for (int i=0; i<kiNodeList.getLength(); i++) {
                Element kiElement = (Element) kiNodeList.item(i);
                String kiado = kiElement.getTextContent();
                ki_lista.add(kiado);

                String ki_alap =
kiElement.getElementsByTagName("alapitas").item(0).getTextContent();

                if (ki_alap.startsWith("19")) {
                    System.out.print("\nNeve : " +
kiElement.getElementsByTagName("nev").item(0).getTextContent() +
"\nAlapítása : " +
kiElement.getElementsByTagName("alapitas").item(0).getTextContent()
+ "\nHelye : " +
```

```
kiElement.getElementsByTagName("hely").item(0).getTextContent() +
"\n");
                }
            }

            /* 5. Lekérdezi az összes kártyát amelyiknek a kulcsában
(vonalkódjában) található kettes */
            NodeList karNodeList =
doc.getElementsByTagName("kartya");
            System.out.print("\n\n\n5. Kártya vonalkód lekérdezés");

            for (int i=0; i<karNodeList.getLength(); i++) {
                Element karElement = (Element) karNodeList.item(i);
                String kartya = karElement.getTextContent();
                kar_lista.add(kartya);

                String kar_id = karElement.getAttribute("vonalkod");

                if (kar_id.matches(".+2.+")){
                    System.out.print("\nTipus : " +
karElement.getElementsByTagName("tipus").item(0).getTextContent() +
"\nAkció : " +
karElement.getElementsByTagName("akcio").item(0).getTextContent() +
"\n");
                }
            }

        }catch (Exception e) {
            e.printStackTrace();
        }


    }
}
```

2d) adatírás

A kódom először létrehoz egy új XML dokumentumot a DocumentBuilderFactory és DocumentBuilder osztályok segítségével. Ezt követően létrehoz egy gyökér elemet (konyvesvasarlas), majd különböző elemeket hoz létre az XML fájlban tárolt adatoknak megfelelően pl(konyvesbolt, konyv stb.). Az elemek létrehozása során a megfelelő attribútumokat és alárendelt elemeket állítja be. Az egyes elemek előtt kommenteket is hozzáad. Végül, az elkészült XML dokumentumot a TransformerFactory és Transformer osztályok segítségével transzformálja és írja ki az eredményt a konzolra és egy XML fájlba.

```
package hu.domparse.QTFL19;
```

```
import java.io.File;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.transform.OutputKeys;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;

import org.w3c.dom.Comment;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;

public class DomWriteQTFL19 {

    public static void main(String argv[]) throws Exception {
        DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
DocumentBuilder dBuilder = factory.newDocumentBuilder();

Document doc = dBuilder.newDocument();

        Element root = doc.createElementNS("XMLQTFL19",
"konyvesvasarlas");
        doc.appendChild(root);

        /* könyvesboltok */
        String[] email = {"lapos@gmail.com"};
        String[] tel = {"06305984652"};
        root.appendChild(createKbolt(doc, "kb1", "LAPos", email,
tel, "19.020"));
        String[] email2 = {"alap@freemail.hu"};
        String[] tel2 = {"06201523495"};
        root.appendChild(createKbolt(doc, "kb2", "aLAP", email2,
tel2, "45.000"));
        String[] email3 = {"lapoz@gmail.hu"};
        String[] tel3 = {"06708623562"};
        root.appendChild(createKbolt(doc, "kb3", "LAPoz", email3,
tel3, "20.900"));

        Element elem = (Element)
doc.getElementsByTagName("konyvesbolt").item(0);
        Comment com = doc.createComment("Konyvesboltok");
        elem.getParentNode().insertBefore(com, elem);


        /* könyvek */
        String[] mufaj = {"dráma", "romantikus", "komédia"};
```

```
        root.appendChild(createKonyv(doc, "k1", "Oppa és Yeobo",
mufaj, "Tóth Péter Tamás"));
        String[] mufaj2 = {"dráma", "romantikus", "komédia"};
        root.appendChild(createKonyv(doc, "k2", "Bankrablás",
mufaj2, "Szabó János"));
        String[] mufaj3 = {"dráma", "romantikus", "komédia"};
        root.appendChild(createKonyv(doc, "k3", "Shuriken", mufaj3,
"Bogyó Eszter"));
        String[] mufaj4 = {"dráma", "romantikus", "komédia"};
        root.appendChild(createKonyv(doc, "k4", "Egyetem", mufaj4,
"Garamszegi Márton"));

        elem = (Element) doc.getElementsByTagName("konyv").item(0);
        com = doc.createComment("Konyvek");
        elem.getParentNode().insertBefore(com, elem);


        /* kiadók */
        root.appendChild(createKiado(doc, "1234567895", "Pub",
"1980.10.21", "California"));
        root.appendChild(createKiado(doc, "1578623598", "Money",
"2005.09.13", "Bikinifenék"));
        root.appendChild(createKiado(doc, "1452368952",
"OneTrickPony", "2010.02.30", "New York"));
        root.appendChild(createKiado(doc, "1982332959", "KiAD",
"1996.05.19", "Budapest"));

        elem = (Element) doc.getElementsByTagName("kiado").item(0);
        com = doc.createComment("Kiadok");
        elem.getParentNode().insertBefore(com, elem);


        /* vásárlók */
        root.appendChild(createVas(doc, "v1", "Súkeník Erik",
"8596", "EriKing@gmail.com"));
        root.appendChild(createVas(doc, "v2", "Szabó Dávid", "1045",
"distrect@citromail.hu"));
        root.appendChild(createVas(doc, "v3", "Magyar János",
"1139", "makkos98@outlook.hu"));
        root.appendChild(createVas(doc, "v4", "Bogyó Márta", "9562",
"bogyo@gmail.com"));
        root.appendChild(createVas(doc, "v5", "Morty Smith", "6243",
"smith@gmail.com"));

        elem = (Element)
doc.getElementsByTagName("vasarlo").item(0);
        com = doc.createComment("Vasarlok");
        elem.getParentNode().insertBefore(com, elem);


        /* kártyák */
```

```
        root.appendChild(createKar(doc, "45236874", "Törzsvársálói",
"15", "2015.04.28"));
        root.appendChild(createKar(doc, "85632982", "Törzsvársálói",
"90", "1980.09.10"));
        root.appendChild(createKar(doc, "65369848", "Ajándék", "30",
"2020.12.20"));
        root.appendChild(createKar(doc, "53248725", "Törzsvársálói",
"30", "1998.06.20"));
        root.appendChild(createKar(doc, "35986147", "Ajándék", "30",
"2023.08.02"));

        elem = (Element) doc.getElementsByTagName("kartya").item(0);
        com = doc.createComment("Kartyak");
        elem.getParentNode().insertBefore(com, elem);


        /* vétel kapcsolatok */
        root.appendChild(createVet(doc, "kb1", "v3", "2018.11.04"));
        root.appendChild(createVet(doc, "kb2", "v1", "2022.03.15"));
        root.appendChild(createVet(doc, "kb3", "v2", "2023.08.06"));
        root.appendChild(createVet(doc, "kb2", "v4", "2023.04.23"));
        root.appendChild(createVet(doc, "kb1", "v5", "2020.10.08"));
        root.appendChild(createVet(doc, "kb3", "v4", "2021.07.10"));

        elem = (Element) doc.getElementsByTagName("vetel").item(0);
        com = doc.createComment("Vetel kapcsolatok");
        elem.getParentNode().insertBefore(com, elem);



        /* Transform */
        TransformerFactory transformerFactory =
TransformerFactory.newInstance();
Transformer transf = transformerFactory.newTransformer();

transf.setOutputProperty(OutputKeys.ENCODING, "UTF-8");
transf.setOutputProperty(OutputKeys.INDENT, "yes");
transf.setOutputProperty("{https://xml.apache.org/xslt}indent-
amount", "2");

// File létrehozása
DOMSource source = new DOMSource(doc);
File OutPutFile = new File("XMLQTFL191.xml");

// Konzolra való kiíratás
StreamResult console = new StreamResult(System.out);
StreamResult file = new StreamResult(OutPutFile);

transf.transform(source, console);
transf.transform(source, file);
    }
```

```
    private static Node createElement(Document doc, String name,
String value) {
Element node = doc.createElement(name);
node.appendChild(doc.createTextNode(value));

return node;
}

private static Node[] appendArray(Document doc, String name,
String[] value) {
Element nodes[] = new Element[value.length];

for (int i = 0; i < value.length; i++) {

nodes[i] = doc.createElement(name);
nodes[i].appendChild(doc.createTextNode(value[i]));

}
return nodes;
}

    private static Node createKbolt(Document doc, String kbid,
String nev, String[] email, String[] telefon, String kapacitas) {
        Element kb = doc.createElement("konyvesbolt");

        kb.setAttribute("KB_ID", kbid);
        kb.appendChild(createElement(doc, "nev", nev));
        Node[] node_email = appendArray(doc, "email", email);
        Node[] node_tel = appendArray(doc, "telefon", telefon);
        kb.appendChild(createElement(doc, "kapacitas", kapacitas));

        for (int i=0; i<email.length; i++) {
            Element eler = doc.createElement("elerhetoseg");
            eler.appendChild(node_email[i]);
            eler.appendChild(node_tel[i]);
            kb.appendChild(eler);
        }
        return kb;
    }


    private static Node createKonyv(Document doc, String kid, String
cim, String[] mufaj, String iro) {
        Element k = doc.createElement("konyv");

        k.setAttribute("K_ID", kid);
        k.appendChild(createElement(doc, "cim", cim));
        Node[] node_muf = appendArray(doc, "mufaj", mufaj);
        k.appendChild(createElement(doc, "iro", iro));
```

```
        for(int i=0; i<mufaj.length; i++) {
            k.appendChild(node_muf[i]);
        }
        return k;
    }


    private static Node createKiado(Document doc, String ado, String
nev, String alap, String hely) {
        Element ki = doc.createElement("kiado");

        ki.setAttribute("Adoszam", ado);
        ki.appendChild(createElement(doc, "nev", nev));
        ki.appendChild(createElement(doc, "alapitas", alap));
        ki.appendChild(createElement(doc, "hely", hely));
        return ki;
    }


    private static Node createVas(Document doc, String vid, String
nev, String irany, String email) {
        Element vas = doc.createElement(("vasarlo"));

        vas.setAttribute("V_ID", vid);
        vas.appendChild(createElement(doc, "nev", nev));
        vas.appendChild(createElement(doc, "iranyitoszam", irany));
        vas.appendChild(createElement(doc, "email", email));
        return vas;
    }


    private static Node createKar(Document doc, String von, String
tip, String akc, String igeny) {
        Element kar = doc.createElement("kartya");

        kar.setAttribute("vonalkod", von);
        kar.appendChild(createElement(doc, "tipus", tip));
        kar.appendChild(createElement(doc, "akcio", akc));
        kar.appendChild(createElement(doc, "igenyles", igeny));
        return kar;
    }


    private static Node createVet(Document doc, String kbid, String
vid, String vasar) {
        Element vet = doc.createElement("vetel");

        vet.setAttribute("KB_ID", kbid);
        vet.setAttribute("V_ID", vid);
        vet.appendChild(createElement(doc, "vasarlas", vasar));
```

```
        return vet;
    }


}
```