

JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Csomagkövető rendszer

Készítette: Gyáni Kevin Zsolt

Neptunkód: CBOYZF

Dátum: 2023.10. 27

Tartalomjegyzék:

Bevezetés:	3
1.Feladat	3
1a Bevezetés,.....	3
1b Az adatbázis ER modell tervezése,.....	5
1c Az XDM modell alapján XML dokumentum készítése,	6
1d Az XML dokumentum alapján XMLSchema készítése - saját típusok, ref, key, keyref, speciális elemek,.....	9
2.feladat	13
2a adatolvasás,	13
2b adatmódosítás,	16
2c adatlekérdezés,.....	18
2d adatírás,.....	22

Bevezetés:

1.Feladat

1a bevezetés,

A beadandó feladatomban során egy számítástechnikai eszközök értékesítésével foglalkozó internetes áruház csomagkövető rendszerét igyekeztem lemodellezni XML struktúrában. Az áruház egyebek mellett teszteli is a számítástechnikai eszközöket, nyomon követi a gyári hibás darabokat, és több raktárral és futárszolgálattal is kapcsolatban áll. Az adatok nyilvántartása érdekében 5 egyedet hoztam létre, amelyek a következők:

<Beszallito>

<Raktarak>

<GyarihibasTermek>

<Rendeles>

<Ugyfel>

Először is érdemes pár szót ejteni a <Beszallito> egyedről, ez a kiinduló pontja a teljes adatbázisnak. Ez az egyed tárolja a különböző beszállítókat <ID> (ezek lesznek az egyedi kulcsok) szerint, illetve információkat biztosít még a csomagolás típusáról, valamint a csomag pontos áráról is. Továbbá tárolja még a várható érkezéssel kapcsolatos információkat, így a vevő pontos képet kaphat arról, hogy mikorra várható a csomagja. A <Beszallito> és a <Raktarak> között több-több kapcsolat van, ugyanis 1 beszállító cég több raktárba is szállíthat, és 1 raktárba több beszállító cég áruja is érkezhet.

A beszállítótól a <Raktarak> nevezetű egyedbe érkeznek a csomagok, mivel az áruház nemzetközi szinten is forgalmaz termékeket, így több raktára is van, amelyek más-más helyszínen helyezkednek el, éppen ezért ez a tábla tárolja a raktárba érkezett termék címét, valamint <ID>-ját (ezek lesznek az egyedi kulcsok). Ezek mellett információkat biztosít még a termék áráról, valamint raktárba érkezés pontos dátumáról, így számon lehet tartani, mennyire volt pontos a beszállítótól kapott várható érkezés.

A <Raktarak> és a <GyarihibasTermek> között 1-1 kapcsolatot létesítettem, ugyanis csak 1 egyedi <ID>-val rendelkező termék lehetséges 1 raktárban.

A webshop által forgalmazott termékek lehetnek gyári hibásak is. Éppen ezért ezeket a termékeket a raktárban ellenőrzik, és amennyiben valamilyen hiba lép fel a termék tesztelése után, azokat a beszállítón keresztül visszaküldik a forgalmazóhoz. Ebben a táblában a rendszerezés érdekében szükséges letárolni a termék <ID>-ját (ezek lesznek az egyedi kulcsok), illetve árát, márkáját és nevét a könnyebb azonosítás érdekében.

Amennyiben a termék működése helyénvalónak bizonyul, a raktárból a terméket feladják a kért <Rendeles>-re. Ebben az egyedben vannak tárolva az ezzel

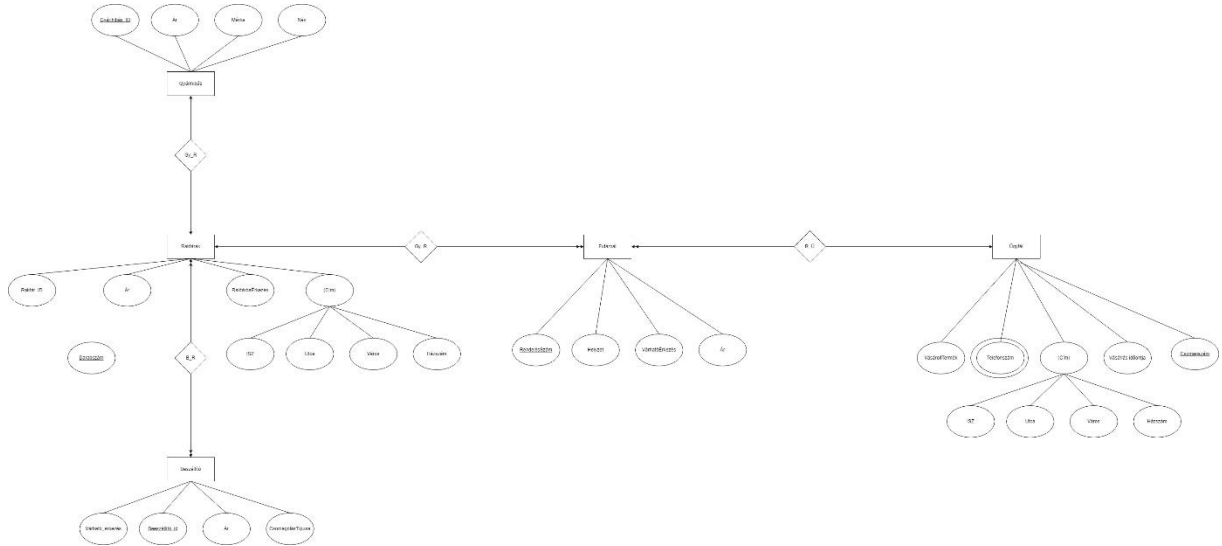
kapcsolatos információk, például a rendelés száma (ezek lesznek az egyedi kulcsok), a csomagkövetés érdekében a rendelés helyzete, illetve várható érkezése, valamint a fizetendő összeg, mivel ezek az ügyfél számára mind lényeges információk. A <Rendelés> és <Ügyfél> között több-egy kapcsolatot létesítettem, mivel 1 rendelés csak 1 ügyfélhez tartozhat, de egy ügyfélnek lehet több különböző rendelése is.

Végül pedig érdemes még beszélni az <Ügyfél> egyedről, melyben a csomagok rendelőjének beazonosításának érdekében szükséges adatok találhatóak. A vásárolt termék nevét, valamint a vásárlás időpontját. A pontosság és duplikációk elkerülésének érdekében bekérjük még az ügyfél személyi igazolványának számát (ezek lesznek az egyedi kulcsok), valamint a kapcsolattartás érdekében a telefonszámát és a kiszállításhoz szükséges lakcímet is.

1b Az adatbázis ER modell tervezése,

A már fent említett adatbázis ER-modelle a következő:

A már fent említett adatbázis ER-modelle a következő:



Ennek a konvertálása után a következő XDM modellt kapjuk:

Az adatbázis XDM (XML Document Model) modellre történő konvertálása során az adatokat XML dokumentumoké alakítjuk át. Ez a konvertálás az alábbi szabályok szerint történik:

egyed \Rightarrow elem

elemi tulajdonság \Rightarrow szöveg elem

kulcs tulajdonság \Rightarrow elemjellemező + kulcs megkötés

összetett tulajdonság \Rightarrow elemeket tartalmazó gyerekelem

többértékű tulajdonság \Rightarrow gyerekelem, ismétlődéssel

kapcsoló tulajdonság \Rightarrow elemjellemező + idegen kulcs megkötés

1:N kapcsolat \Rightarrow elemjellemző + kulcs + idegen kulcs megkötés

N:M kapcsolat \Rightarrow külön kapcsoló elem és idegen kulcsok mindkét oldalra

1c Az XDM modell alapján XML dokumentum készítése,

Az xml dokumentumban példányosítom az egyedek, strukturáltan, megfelelően jelölve a kulcsokat, ahol szükséges pedig az idegenkulcsokat. Ezt a folyamatot megismétlem a többi elemre. Létrehozom továbbá a B_R kapcsolótáblát, melynek attribútumaiként megadom a darabszámot, raktár illetve beszállító idegen kulcsát. Az xml fájl kódja a következő:

```
<?xml version="1.0" encoding="UTF-8"?>
<Csomag_követés_CB0YZF xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
xs:noNamespaceSchemaLocation="XMLSchemaCB0YZF.xsd">

  <!-- Raktárak példányosítása -->
  <raktár raktár_id="01">
    <ár>500 </ár>
    <raktárba_érkezés>2023-10-10 </raktárba_érkezés>
    <cím>
      <Isz>3881 </Isz>
      <Város>Abaújszántó </Város>
      <Utca>Béke út </Utca>
      <házszám>1 </házszám>
    </cím>
  </raktár>

  <raktár raktár_id="02">
    <ár>1500 </ár>
    <raktárba_érkezés>2023-10-11 </raktárba_érkezés>
    <cím>
      <Isz>3881 </Isz>
      <Város>Abaújszántó </Város>
      <Utca>Béke út </Utca>
      <házszám>2 </házszám>
    </cím>
  </raktár>

  <raktár raktár_id="03">
    <ár>1500 </ár>
    <raktárba_érkezés>2023-10-12 </raktárba_érkezés>
    <cím>
      <Isz>3860 </Isz>
      <Város>Encs </Város>
      <Utca>Rákóczi út </Utca>
      <házszám>3 </házszám>
    </cím>
  </raktár>

  <!-- Gyárihibás elemek példányosítása -->
  <Gyárihibás gyárihibás_id="11" raktár="01">
```

```

        <ár>100 </ár>
        <márka>Samsung </márka>
        <név>Telefon </név>
    </Gyárihiba>

    <Gyárihiba gyárihiba_id="12" raktár="02">
        <ár>200 </ár>
        <márka>Apple </márka>
        <név>Airpods </név>
    </Gyárihiba>

    <Gyárihiba gyárihiba_id="13" raktár="03">
        <ár>300 </ár>
        <márka>Sony </márka>
        <név>Playstation </név>
    </Gyárihiba>

    <!-- B_R kapcsoló tábla példányosítása -->

    <B_R darabszám="5" raktár="01" beszállító="21" ></B_R>

    <B_R darabszám="10" raktár="01" beszállító="22" ></B_R>

    <B_R darabszám="25" raktár="03" beszállító="23" ></B_R>

    <!-- Beszállító elem példányosítása -->

    <beszállító beszállító_id="21" raktár="01">
        <várható_érkezés>2023-10-18 </várható_érkezés>
        <ár>50000 </ár>
        <csomagolás_típusa>Fólia </csomagolás_típusa>
    </beszállító>

    <beszállító beszállító_id="22" raktár="02">
        <várható_érkezés>2023-10-16 </várható_érkezés>
        <ár>40000 </ár>
        <csomagolás_típusa>Boríték </csomagolás_típusa>
    </beszállító>

    <beszállító beszállító_id="23" raktár="03">
        <várható_érkezés>2023-10-19 </várható_érkezés>
        <ár>30000 </ár>
        <csomagolás_típusa>Doboz </csomagolás_típusa>
    </beszállító>

    <!-- Futár elem példányosítása -->

    <futár Rendelés_száma="31" raktár="01">
        <Helyzet>Úton </Helyzet>

```

```
<várható_érkezés>2023-10-28 </várható_érkezés>
<ár>1500 </ár>
</futár>

<futár Rendelés_száma="32" raktár="02">
  <Helyzet>Áll </Helyzet>
  <várható_érkezés>2023-10-23 </várható_érkezés>
  <ár>4000 </ár>
</futár>

<futár Rendelés_száma="33" raktár="03">
  <Helyzet>Felfüggesztve </Helyzet>
  <várható_érkezés>2023-10-29 </várható_érkezés>
  <ár>2500 </ár>
</futár>

<!-- Ügyfél elem példányosítása -->

<ügyfél Csomagszáma="41" futár="31">
  <VásárlásIdőpontja>2023-10-14 </VásárlásIdőpontja>
  <VásároltTermék>Iphone 13 pro max </VásároltTermék>
  <Telefonszám>0620-714-9284</Telefonszám>
  <cím>
    <Isz>3881 </Isz>
    <Város>Abaújszántó </Város>
    <Utca>Rákóczi út </Utca>
    <házszám>3 </házszám>
  </cím>
</ügyfél>

<ügyfél Csomagszáma="42" futár="32">
  <VásárlásIdőpontja>2023-10-12 </VásárlásIdőpontja>
  <VásároltTermék>Smasung galaxy A52 </VásároltTermék>
  <Telefonszám>0620-394-2132</Telefonszám>
  <Telefonszám>0630-153-4576</Telefonszám>
  <cím>
    <Isz>3881 </Isz>
    <Város>Abaújszántó </Város>
    <Utca>Béke út </Utca>
    <házszám>4 </házszám>
  </cím>
</ügyfél>

<ügyfél Csomagszáma="43" futár="33">
  <VásárlásIdőpontja>2023-10-18 </VásárlásIdőpontja>
  <VásároltTermék>Xbox Series X </VásároltTermék>
  <Telefonszám>0670-345-2376</Telefonszám>
  <cím>
    <Isz>3881 </Isz>
```



```

        <Város>Abaújszántó </Város>
        <Utca>Kazincy út </Utca>
        <házszám>19 </házszám>
    </cím>
</ügyfél>

```

```
</Csomag_követés_CB0YZF>
```

1d Az XML dokumentum alapján XMLSchema készítése - saját típusok, ref, key, keyref, speciális elemek,

Ezután létrehozom a xml-ben megadott típusokat illetve kulcsokat meghatározó sémát, kigyűjtöm az egyszerű típusokat, elementeket illetve ezek megszorításait beállítom. Ezt követően meghatározom a saját, komplex típusaimat, ezekre is alkalmazom a megszorításaimat, itt már felhasználva az egyszerű típusokat. Ezt követő lépésként a gyökérelemről indulva felépítem az XML struktúrát, beállítom az elsődleges kulcsokat, valamint az elsődleges kulcsokra az idegenkulcsokat, illetve az 1:1 kapcsolat megvalósításához használok a Unique kulcsszót is.

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <!-- Egyszerű típusok kigyűjtése, saját típusok meghatározása, megszorítás -->
    <xs:element name="ár" type="xs:positiveInteger"/>
    <xs:element name="raktárba_érkezés" type="xs:date"/>
    <xs:element name="márka" type="xs:string"/>
    <xs:element name="név" type="xs:string"/>
    <xs:element name="várható_érkezés" type="xs:date"/>
    <xs:element name="csomagolás_típusa" type="xs:string"/>
    <xs:element name="Helyzet" type="xs:string"/>
    <xs:element name="VásárlásIdőpontja" type="xs:date"/>
    <xs:element name="VásároltTermék" type="xs:string"/>
    <xs:element name="Telefonszám" type="TelefonszámTípus"/>

    <xs:simpleType name="TelefonszámTípus">
        <xs:restriction base="xs:string">
            <xs:pattern value="\d{4}-\d{3}-\d{4}" />
        </xs:restriction>
    </xs:simpleType>

    <!--Komplex típusokhoz saját típus meghatározása, sorrendiség, számosság etc. -->

    <xs:complexType name="raktárTípus">
        <xs:sequence>
            <xs:element ref="ár" />
            <xs:element ref="raktárba_érkezés" maxOccurs="1" />
            <xs:element name="cím">
                <xs:complexType>

```

```

        <xs:sequence>
            <xs:element name="Isz" type="xs:integer" />
            <xs:element name="Város" type="xs:string" />
            <xs:element name="Utca" type="xs:string" />
            <xs:element name="házszám" type="xs:integer" />
        </xs:sequence>
    </xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="raktár_id" type="xs:integer" use="required" />
</xs:complexType>

<xs:complexType name="GyárihibásTípus">
    <xs:sequence>
        <xs:element ref="ár" />
        <xs:element ref="márka" maxOccurs="1" />
        <xs:element ref="név" maxOccurs="1" />
    </xs:sequence>
    <xs:attribute name="gyárihibás_id" type="xs:integer" use="required" />
    <xs:attribute name="raktár" type="xs:integer" use="required" />
</xs:complexType>

<xs:complexType name="B_RTípus">
    <xs:attribute name="darabszám" type="xs:integer" use="required" />
    <xs:attribute name="raktár" type="xs:integer" use="required" />
    <xs:attribute name="beszállító" type="xs:integer" use="required" />
</xs:complexType>

<xs:complexType name="beszállítóTípus">
    <xs:sequence>
        <xs:element ref="várható_érkezés" maxOccurs="1"/>
        <xs:element ref="ár" />
        <xs:element ref="csomagolás_típusa" maxOccurs="1"/>
    </xs:sequence>
    <xs:attribute name="raktár" type="xs:integer" use="required" />
    <xs:attribute name="beszállító_id" type="xs:integer" use="required" />
</xs:complexType>

<xs:complexType name="futárTípus">
    <xs:sequence>
        <xs:element ref="Helyzet" maxOccurs="1"/>
        <xs:element ref="várható_érkezés" maxOccurs="1" />
        <xs:element ref="ár" maxOccurs="1" />
    </xs:sequence>
    <xs:attribute name="Rendelés_szám" type="xs:integer" use="required" />
    <xs:attribute name="raktár" type="xs:integer" use="required" />
</xs:complexType>

```

```

<xs:complexType name="ügyfélTípus">
  <xs:sequence>
    <xs:element ref="VásárlásIdőpontja" maxOccurs="1" />
    <xs:element ref="VásároltTermék" maxOccurs="unbounded"/>
    <xs:element ref="Telefonszám" minOccurs="1" maxOccurs="unbounded" />
    <xs:element name="cím">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Isz" type="xs:integer" />
          <xs:element name="Város" type="xs:string" />
          <xs:element name="Utca" type="xs:string" />
          <xs:element name="házszám" type="xs:integer" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="Csomagszám" type="xs:integer" use="required" />
  <xs:attribute name="futár" type="xs:integer" use="required" />
</xs:complexType>

<!-- Gyökérelemtől az elemek felhasználása -->

<xs:element name="Csomag_követés_CBOYZF">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="raktár" type="raktárTípus" minOccurs="0" maxOccurs="100"/>
      <xs:element name="Gyárihibás" type="GyárihibásTípus" minOccurs="0" maxOccurs="100"/>
      <xs:element name="B_R" type="B_RTípus" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="beszállító" type="beszállítóTípus" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="futár" type="futárTípus" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="ügyfél" type="ügyfélTípus" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <!-- Elsődleges kulcsok -->

  <xs:key name="raktár_kulcs">
    <xs:selector xpath="raktár"/>
    <xs:field xpath="@raktár_id"/>
  </xs:key>

  <xs:key name="gyárihibás_kulcs">
    <xs:selector xpath="Gyárihibás"/>
    <xs:field xpath="@gyárihibás_id"/>
  </xs:key>

```

```
</xs:key>

<xs:key name="B_R_kulcs">
  <xs:selector xpath="B_R"/>
  <xs:field xpath="@darabszám"/>
</xs:key>

<xs:key name="beszállító_kulcs">
  <xs:selector xpath="beszállító"/>
  <xs:field xpath="@beszállító_id"/>
</xs:key>

<xs:key name="futár_kulcs">
  <xs:selector xpath="futár"/>
  <xs:field xpath="@Rendelés_száma"/>
</xs:key>

<xs:key name="ügyfél_kulcs">
  <xs:selector xpath="ügyfél"/>
  <xs:field xpath="@Csomagszám"/>
</xs:key>

<!-- Idegen kulcsok -->

<xs:keyref name="gyárihibás_raktár_kulcs" refer="raktár_kulcs">
  <xs:selector xpath="Gyárihibás"/>
  <xs:field xpath="@raktár"/>
</xs:keyref>

<xs:keyref name="beszállító_raktár_kulcs" refer="raktár_kulcs">
  <xs:selector xpath="beszállító"/>
  <xs:field xpath="@raktár"/>
</xs:keyref>

<xs:keyref name="B_R_raktár_kulcs" refer="raktár_kulcs">
  <xs:selector xpath="B_R"/>
  <xs:field xpath="@raktár"/>
</xs:keyref>

<xs:keyref name="B_R_beszállító_kulcs" refer="beszállító_kulcs">
  <xs:selector xpath="B_R"/>
  <xs:field xpath="@beszállító"/>
</xs:keyref>

<xs:keyref name="futár_raktár_kulcs" refer="raktár_kulcs">
  <xs:selector xpath="futár"/>
  <xs:field xpath="@raktár"/>
</xs:keyref>
```

```

<xs:keyref name="ügyfél_futár_kulcs" refer="futár_kulcs">
  <xs:selector xpath="ügyfél"/>
  <xs:field xpath="@futár"/>
</xs:keyref>

<!-- Az 1:1 kapcsolat megvalósítás -->
<xs:unique name="Gyárihiba_Raktár_egyegy">
  <xs:selector xpath="Gyárihibás"/>
  <xs:field xpath="@raktár"/>
</xs:unique>

</xs:element>

</xs:schema>

```

2.feladat

2a adatolvasás,

A szokásos 3 könyvtárból történő import (IO,xml,w3c) után beolvasom a fájlt egy try catch (ez az I/O művelet miatt szükséges) példányosítom a DocumentBuilderFactory a normalizálását követően megnyitom az output file-t, majd meghívom a printWriter nevű függvényt ami egyszerre írja konzolra és fájlba a bemeneti xml dokumentum tartalmát. A dokumentum főbb elemeit NodeListekben tárolom el, ezeken for ciklussal megyek végig, vizsgálom a gyerekelemeket, ezeknek a tartalmát (contextét).

```

package hu.domparse.CBOYZF;

import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.StringJoiner;

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

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

public class DomReadCBOYZF {
    public static void main(String[] args) {
        try {
            File xmlFile = new File("C:\\Egyetem\\CBOYZF_XMLGyak\\XMLTas-
kCBOYZF\\XMLCBOYZF.xml");
            DocumentBuilderFactory dbFactory = DocumentBuilderFactory.
newInstance();
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
            Document doc = dBuilder.parse(xmlFile);
            doc.getDocumentElement().normalize();

```

```

File outputFile = new File("output2.xml");
PrintWriter writer = new PrintWriter(new FileWriter(outputFile,
true));

// Kiírjuk az XML főgyökér elemét a konzolra és fájlba
Element rootElement = doc.getDocumentElement();
String rootName = rootElement.getTagName();
StringJoiner rootAttributes = new StringJoiner(" ");
NamedNodeMap rootAttributeMap = rootElement.getAttributes();

for (int i = 0; i < rootAttributeMap.getLength(); i++) {
    Node attribute = rootAttributeMap.item(i);
    rootAttributes.add(attribute.getNodeName() + "=\"" + attribute.getNodeValue() + "\"");
}

System.out.print("<?xml version=\"1.0\" encoding=\"UTF-8\"?>
\n");
writer.print("<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n");

System.out.print("<" + rootName + " " + rootAttributes.toString() + "> \n");
writer.print("<" + rootName + " " + rootAttributes.toString() + "> \n");

NodeList raktarList = doc.getElementsByTagName("raktár");
NodeList gyarihibasList = doc.getElementsByTagName("Gyárihibas");
NodeList brList = doc.getElementsByTagName("B_R");
NodeList beszallitoList = doc.getElementsByTagName("beszálító");
NodeList futarList = doc.getElementsByTagName("futár");
NodeList ugyfelList = doc.getElementsByTagName("ügyfél");

// Kiírjuk az XML-t a konzolra megtartva az eredeti formázást
printNodeList(raktarList, writer);
System.out.println("");
writer.println("");
printNodeList(gyarihibasList, writer);
System.out.println("");
writer.println("");
printNodeList(brList, writer);
System.out.println("");
writer.println("");
printNodeList(beszallitoList, writer);
System.out.println("");
writer.println("");
printNodeList(futarList, writer);
System.out.println("");
writer.println("");
printNodeList(ugyfelList, writer);

// Zárjuk le az XML gyökér elemét
System.out.println("</" + rootName + ">");
writer.append("</" + rootName + ">");

writer.close();
} catch (Exception e) {

```

```

        e.printStackTrace();
    }
}

// Rekurzív függvény a NodeList tartalmának kiírására
private static void printNodeList(NodeList nodeList, PrintWriter writer) {
    for (int i = 0; i < nodeList.getLength(); i++) {
        Node node = nodeList.item(i);
        printNode(node, 0, writer);
        System.out.println(""); // Üres sor hozzáadása az elemek között
        writer.println(""); // Üres sor hozzáadása a fájlban az elemek között
    }
}

// Rekurzív függvény a Node tartalmának kiírására
private static void printNode(Node node, int indent, PrintWriter writer) {
    if (node.getNodeType() == Node.ELEMENT_NODE) {
        Element element = (Element) node;
        String nodeName = element.getTagName();
        StringJoiner attributes = new StringJoiner(" ");
        NamedNodeMap attributeMap = element.getAttributes();

        for (int i = 0; i < attributeMap.getLength(); i++) {
            Node attribute = attributeMap.item(i);
            attributes.add(attribute.getNodeName() + "=\"" + attribute.getNodeValue() + "\"");
        }

        System.out.print(getIndentString(indent));
        System.out.print("<" + nodeName + " " + attributes.toString() + ">");

        writer.print(getIndentString(indent));
        writer.print("<" + nodeName + " " + attributes.toString() + ">");

        NodeList children = element.getChildNodes();
        if (children.getLength() == 1 && children.item(0).getNodeType() == Node.TEXT_NODE) {
            System.out.print(children.item(0).getNodeValue());
            writer.print(children.item(0).getNodeValue());
        } else {
            System.out.println();
            writer.println();
            for (int i = 0; i < children.getLength(); i++) {
                printNode(children.item(i), indent + 1, writer);
            }
            System.out.print(getIndentString(indent));
            writer.print(getIndentString(indent));
        }
        System.out.println("</" + nodeName + ">");
        writer.println("</" + nodeName + ">");
    }
}

// Segédmetódus az indentáláshoz
private static String getIndentString(int indent) {
    StringBuilder sb = new StringBuilder();

```

```

        for (int i = 0; i < indent; i++) {
            sb.append("  "); // 2 spaces per indent level
        }
        return sb.toString();
    }
}

```

2b adatmódosítás,

A szokásos 3 könyvtárból történő import (IO,xml,w3c) után beolvasom a fájlt egy try catch (ez az I/O művelet miatt szükséges) példányosítom a DocumentBuilderFactory a normalizálását követően, a dokumentum főbb elemeit nodeListekben tárolom el. Az elemek módosítását úgy végzem el hogy hogy lekérem egy element tartalmát a `getElementsByTagName` dom függvénnyel, ezután a `SetContext` metódussal módosítom a tartalmát, ugyan ezt, a feladat kiírásnak megfelelően, elvégzem még 4 esetben.

```

package hu.domparsed.CBOYZF;

import javax.xml.parsers.*;
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.*;
import java.io.*;

public class DOMModifyCBOYZF {

    public static void main(String[] args) {

        try {

            // XML fájl beolvasása

            File xmlFile = new
File("C:\\Egyetem\\CBOYZF_XMLGyak\\XMLTaskCBOYZF\\XMLCBOYZF.xml");

            // builder factoryk létrehozása

            DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();

            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();

            //létrehozom a doc-ot amit később a transformhoz használok fel

            Document doc = dBuilder.parse(xmlFile);

```



```

        // lekérjük egy adott típushoz tartozó összes elemet amit egy
        listában tárolunk el

        NodeList raktarList = doc.getElementsByTagName("raktár");

        //lekérjük azt az elemet a listából amelyiket módosítani
        szeretnénk, itt index alapján történik a módosítás

        Element raktar = (Element) raktarList.item(0);

        //az elemnek megkeressük azt a tagjét amit módosítani
        szeretnénk, majd a tartlmát (content)-et beállítjuk a megfelelőre

        raktar.getElementsByTagName("Város").item(0).setTextContent("Baskó");


        //a felső mintájára elvégzek még 4 módosítást.


        NodeList beszállítóList =
        doc.getElementsByTagName("beszállító");

        Element beszállító = (Element) beszállítóList.item(0);

        beszállító.getElementsByTagName("ár").item(0).setTextContent("999999");


        NodeList gyarihibasList =
        doc.getElementsByTagName("Gyárihibás");

        Element gyarihibas = (Element) gyarihibasList.item(1);

        gyarihibas.getElementsByTagName("márka").item(0).setTextContent("Huawei");


        NodeList futarList = doc.getElementsByTagName("futár");

        Element futar = (Element) futarList.item(1);

        futar.getElementsByTagName("Helyzet").item(0).setTextContent("ebédszünet");


        NodeList ugyfelList = doc.getElementsByTagName("ügyfél");

        Element ugyfel = (Element) ugyfelList.item(1);

        ugyfel.getElementsByTagName("VásároltTermék").item(0).setTextContent("Plays
        tation 5");


        // Kiírjuk a módosított XML fájlt konzolra

        //A konzolra íratáshoz transformerFactoryt használok, mivel ez
        a leg egyszerűbb módja

        //Létrehozok a factoryból egy új példányt (instancet)

        TransformerFactory transformerFactory =
        TransformerFactory.newInstance();

```

```

        //Beállítom a transformert
        Transformer transformer = transformerFactory.newTransformer();
        //Megadom a forrás fájlt amit fent létrehoztam
        DOMSource source = new DOMSource(doc);
        //Megnyitom a streamet és konzolra kiíratom sys.out-al a fájlt
        StreamResult consoleResult = new StreamResult(System.out);
        transformer.transform(source, consoleResult);

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

```

2c adatlekérdezés,

A szokásos 3 könyvtárból történő import (IO,xml,w3c) után beolvasom a fájlt egy try chatben (ez az I/O művelet miatt szükséges) példányosítom a DocumentBuilderFactory a normalizálását követően, a dokumentum főbb elemeit nodeListekben tárolom el. A lekérdezés során egy nodelistbe lekérem az adott fő elem gyerekeit (child/descendant) majd egy for ciklussal végig iterálok a listán, majd ahol a kért adat egyezését megtalálom ,egy stringbe letárolom a kért attribútumot/tartalmat (contextet), kereszt táblás (kereszt elemes) lekérdezés esetén dupla for ciklust használok.

```

package hu.domparsed.CBOYZF;

import org.w3c.dom.*;
import javax.xml.parsers.*;
import java.io.*;

public class DOMQueryCBOYZF {
    public static void main(String[] args) {
        try {
            // XML fájl beolvasása és DOM létrehozása
            DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();

            DocumentBuilder builder = factory.newDocumentBuilder();

            Document document = builder.parse(new
File("C:\\\\Egyetem\\\\CBOYZF_XMLGyak\\\\XMLTaskCBOYZF\\\\XMLCBOYZF.xml"));

```

```

// 1. Lekérdezés: "02"-es ID-jú raktár adatai
String raktarId = "02";
NodeList raktarList = document.getElementsByTagName("raktár");
for (int i = 0; i < raktarList.getLength(); i++) {
    Element raktar = (Element) raktarList.item(i);
    String raktarIdAttribute =
raktar.getAttribute("raktár_id");
    if (raktarIdAttribute.equals(raktarId)) {
        String ar =
raktar.getElementsByTagName("ár").item(0).getTextContent();
        String raktarbaErkezes =
raktar.getElementsByTagName("raktárba_érkezés").item(0).getTextContent();
        System.out.println("Az '" + raktarId + "' ID-jú
raktárban található termék ára: " + ar);
        System.out.println("Raktárba érkezés időpontja: " +
raktarbaErkezes);
        break;
    }
}

// Lekérdezés 2: Az "12"-es ID-jú gyárihibás termék raktárának
címe

String gyariHibasId = "12";
NodeList gyariHibasList =
document.getElementsByTagName("Gyárihibás");
for (int i = 0; i < gyariHibasList.getLength(); i++) {
    Element gyariHibas = (Element) gyariHibasList.item(i);
    if
(gyariHibas.getAttribute("gyárihibás_id").equals(gyariHibasId)) {
        String raktarIdOfGyariHibas =
gyariHibas.getAttribute("raktár");
        NodeList raktarListForGyariHibas =
document.getElementsByTagName("raktár");
        for (int j = 0; j <
raktarListForGyariHibas.getLength(); j++) {
            Element raktar = (Element)
raktarListForGyariHibas.item(j);
            if
(raktar.getAttribute("raktár_id").equals(raktarIdOfGyariHibas)) {
                Element cim = (Element)
raktar.getElementsByTagName("cím").item(0);
                String isz =
cim.getElementsByTagName("Isz").item(0).getTextContent().trim();

```

```

        String varos =
cim.getElementsByTagName("Város").item(0).getTextContent().trim();

        String utca =
cim.getElementsByTagName("Utca").item(0).getTextContent().trim();

        String hazszam =
cim.getElementsByTagName("házszám").item(0).getTextContent().trim();

        System.out.println("Lekérdezés 2:");

        System.out.println("Cím: " + isz + " " + varos
+ " " + utca + " " + hazszam);

        break;

    }

    }

    break;

}

// Lekérdezés 3: A "23"-as beszállítóhoz tartozó termék darabszáma
String beszallitoId = "23";

NodeList bRList = document.getElementsByTagName("B_R");

int darabszam = 0;

for (int i = 0; i < bRList.getLength(); i++) {

    Element bR = (Element) bRList.item(i);

    String beszallitoIdInBR = bR.getAttribute("beszállító");

    if (beszallitoIdInBR.equals(beszallitoId)) {

        darabszam +=
Integer.parseInt(bR.getAttribute("darabszám"));

    }

}

System.out.println("Lekérdezés 3:");

System.out.println("A \"23\"-as beszállítóhoz tartozó termék
darabszáma: " + darabszam);

// Lekérdezés 4: Azon ügyfeleknél lévő termék és telefonszámának
kiírása, akiknek több telefonszámuk van

NodeList ugyfelList = document.getElementsByTagName("ügyfél");

for (int i = 0; i < ugyfelList.getLength(); i++) {

    Element ugyfel = (Element) ugyfelList.item(i);

```

```

        NodeList telefonszamList =
ugyfel.getElementsByTagName("Telefonszám");

        if (telefonszamList.getLength() > 1) {

            String nev =
ugyfel.getElementsByTagName("VásároltTermék").item(0).getTextContent().trim
();

            System.out.println("Lekérdezés 4:");

            System.out.println("Termék neve: " + nev);

            System.out.println("Telefonszám(ok):");

            for (int j = 0; j < telefonszamList.getLength(); j++) {

                Element telefonszam = (Element)
telefonszamList.item(j);

                System.out.println(telefonszam.getTextContent().trim());

            }

        }

    }

    // Lekérdezés 5: Azon futárok nevének kiírása, akiknél az áru
értéke 2000 fölött van

    NodeList futarList = document.getElementsByTagName("futár");

    System.out.println("Lekérdezés 5:");

    for (int i = 0; i < futarList.getLength(); i++) {

        Element futar = (Element) futarList.item(i);

        int ar =
Integer.parseInt(futar.getElementsByTagName("ár").item(0).getTextContent().
trim());

        if (ar > 2000) {

            String helyzet =
futar.getElementsByTagName("Helyzet").item(0).getTextContent().trim();

            String id = futar.getAttribute("Rendelés_szám");

            System.out.println("A rendelés id-ja: " + id + " a futár
helyzete: " + helyzet);

        }

    }
}

```

```

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

```

2d adatírás,

A szokásos 3 könyvtárból történő import (IO,xml,w3c) után beolvasom a fájlt egy try catch (ez az I/O művelet miatt szükséges) példányosítom a DocumentBuilderFactory a normalizálását követően, a dokumentum főbb elemeit nodeListekben tárolom el. A dokumentum faszerkezetének felépítését úgy végzem el hogy létrehozom a gyökér elemet, majd ehhez adom hozzá a később létrehozott főelemeket. A kiírás konzolra és fájlba az 2a,read feladathoz hasonló módon történik.

```
package hu.domparse.CBOYZF;
```

```
import java.io.File;
```

```
import java.io.FileWriter;
```

```
import java.io.PrintWriter;
```

```
import java.util.StringJoiner;
```

```
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.Document;
```

```
import org.w3c.dom.Element;
```

```
import org.w3c.dom.Node;
```

```
import org.w3c.dom.NodeList;
```

```
import org.w3c.dom.NamedNodeMap;
```

```

public class DOMWriteCBOYZF {
    public static void main(String[] args) {
        try {
            // Create a new Document
            DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
            DocumentBuilder builder = factory.newDocumentBuilder();
            Document doc = builder.newDocument();

            // Create the root element
            Element rootElement = doc.createElement("Csomag_követés_CBOYZF");
            rootElement.setAttribute("xmlns:xs", "http://www.w3.org/2001/XMLSchema-instance");
            rootElement.setAttribute("xs:noNamespaceSchemaLocation", "XMLSchemaCBOYZF.xsd");
            doc.appendChild(rootElement);

            // Add raktárak
            addRaktar(doc, rootElement, "01", "500", "2023-10-10", "3881", "Abaújszántó", "Béke út", "1");
            addRaktar(doc, rootElement, "02", "1500", "2023-10-11", "3881", "Abaújszántó", "Béke út", "2");
            addRaktar(doc, rootElement, "03", "1500", "2023-10-12", "3860", "Encs", "Rákóczi út", "3");

            // Add gyárihibás elemek
            addGyarihibas(doc, rootElement, "11", "01", "100", "Samsung", "Telefon");
            addGyarihibas(doc, rootElement, "12", "02", "200", "Apple", "Airpods");
            addGyarihibas(doc, rootElement, "13", "03", "300", "Sony", "Playstation");

            // Add B_R kapcsoló tábla
            addBR(doc, rootElement, "5", "01", "21");
            addBR(doc, rootElement, "10", "01", "22");
            addBR(doc, rootElement, "25", "03", "23");

            // Add beszállítók
            addBeszallito(doc, rootElement, "21", "01", "2023-10-18", "50000", "Fólia");
            addBeszallito(doc, rootElement, "22", "02", "2023-10-16", "40000", "Boríték");
            addBeszallito(doc, rootElement, "23", "03", "2023-10-19", "30000", "Doboz");
        }
    }
}

```

```

// Add futárok
addFutar(doc, rootElement, "31", "01", "Úton", "2023-10-28", "1500");
addFutar(doc, rootElement, "32", "02", "Áll", "2023-10-23", "4000");
addFutar(doc, rootElement, "33", "03", "Felfüggesztve", "2023-10-29", "2500");

// Add ügyfelek
addUgyfel(doc, rootElement, "41", "31", "2023-10-14", "Iphone 13 pro max", "0620-714-9284",
    "3881", "Abaújszántó", "Rákóczi út", "3");
addUgyfel(doc, rootElement, "42", "32", "2023-10-12", "Smasung galaxy A52", "0620-394-2132,
0630-153-4576",
    "3881", "Abaújszántó", "Béke út", "4");
addUgyfel(doc, rootElement, "43", "33", "2023-10-18", "Xbox Series X", "0670-345-2376",
    "3881", "Abaújszántó", "Kazinczy út", "19");

// Transform and save to file
TransformerFactory transformerFactory = TransformerFactory.newInstance();
Transformer transformer = transformerFactory.newTransformer();
transformer.setOutputProperty(OutputKeys.ENCODING, "UTF-8");
transformer.setOutputProperty(OutputKeys.INDENT, "yes");
transformer.setOutputProperty("{https://xml.apache.org/xslt}indent-amount", "2");

DOMSource source = new DOMSource(doc);
File myFile = new File("XMLCBOYZF2.xml");
StreamResult file = new StreamResult(myFile);
transformer.transform(source, file);

// Now, print the XML content to the console and a new file
printDocument(doc);
} catch (Exception e) {
    e.printStackTrace();
}
}

```



```
private static void addRaktar(Document doc, Element rootElement, String raktarId, String ar, String raktarbaErkezes,
```

```
String isz, String varos, String utca, String hazszam) {
```

```
    Element raktar = doc.createElement("raktár");
```

```
    raktar.setAttribute("raktár_id", raktarId);
```

```
    Element arElement = createElement(doc, "ár", ar);
```

```
    Element raktarbaErkezesElement = createElement(doc, "raktárba_érkezés", raktarbaErkezes);
```

```
    Element cim = doc.createElement("cím");
```

```
    Element iszElement = createElement(doc, "Isz", isz);
```

```
    Element varosElement = createElement(doc, "Város", varos);
```

```
    Element utcaElement = createElement(doc, "Utca", utca);
```

```
    Element hazszamElement = createElement(doc, "házszám", hazszam);
```

```
    cim.appendChild(iszElement);
```

```
    cim.appendChild(varosElement);
```

```
    cim.appendChild(utcaElement);
```

```
    cim.appendChild(hazszamElement);
```

```
    raktar.appendChild(arElement);
```

```
    raktar.appendChild(raktarbaErkezesElement);
```

```
    raktar.appendChild(cim);
```

```
    rootElement.appendChild(raktar);
```

```
}
```

```
private static void addGyarihibas(Document doc, Element rootElement, String gyarihibasId, String raktar, String ar,
```

```
String marka, String nev) {
```

```
    Element gyarihibas = doc.createElement("Gyárihibás");
```

```
    gyarihibas.setAttribute("gyárihibás_id", gyarihibasId);
```

```
gyaribas.setAttribute("raktár", raktar);
```

```
Element arElement = createElement(doc, "ár", ar);
```

```
Element markaElement = createElement(doc, "márka", marka);
```

```
Element nevElement = createElement(doc, "név", nev);
```

```
gyaribas.appendChild(arElement);
```

```
gyaribas.appendChild(markaElement);
```

```
gyaribas.appendChild(nevElement);
```

```
rootElement.appendChild(gyaribas);
```

```
}
```

```
private static void addBR(Document doc, Element rootElement, String darabszam, String raktar, String beszallito) {
```

```
    Element br = doc.createElement("B_R");
```

```
    br.setAttribute("darabszám", darabszam);
```

```
    br.setAttribute("raktár", raktar);
```

```
    br.setAttribute("beszállító", beszallito);
```

```
    rootElement.appendChild(br);
```

```
}
```

```
private static void addBeszallito(Document doc, Element rootElement, String beszallitold, String raktar,
```

```
String varhatoErkezes, String ar, String csomagolasTipusa) {
```

```
    Element beszallito = doc.createElement("beszállító");
```

```
    beszallito.setAttribute("beszállító_id", beszallitold);
```

```
    beszallito.setAttribute("raktár", raktar);
```

```
    Element varhatoErkezesElement = createElement(doc, "várható_érkezés", varhatoErkezes);
```

```
    Element arElement = createElement(doc, "ár", ar);
```

```
    Element csomagolasTipusaElement = createElement(doc, "csomagolás_típusa", csomagolasTipusa);
```

```
    beszallito.appendChild(varhatoErkezesElement);  
    beszallito.appendChild(arElement);  
    beszallito.appendChild(csomagolasTipusaElement);  
  
    rootElement.appendChild(beszallito);  
}
```

```
private static void addFutar(Document doc, Element rootElement, String rendelesSzam, String raktar,  
String helyzet,
```

```
String varhatoErkezes, String ar) {  
    Element futar = doc.createElement("futár");  
    futar.setAttribute("Rendelés_száma", rendelesSzam);  
    futar.setAttribute("raktár", raktar);
```

```
    Element helyzetElement = createElement(doc, "Helyzet", helyzet);  
    Element varhatoErkezesElement = createElement(doc, "várható_érkezés", varhatoErkezes);  
    Element arElement = createElement(doc, "ár", ar);
```

```
    futar.appendChild(helyzetElement);  
    futar.appendChild(varhatoErkezesElement);  
    futar.appendChild(arElement);
```

```
    rootElement.appendChild(futar);  
}
```

```
private static void addUgyfel(Document doc, Element rootElement, String csomagszam, String futar,  
String vasarlasIdopontja,
```

```
String vasaroltTermek, String telefonszam, String isz, String varos, String utca, String hazszam) {  
    Element ugyfel = doc.createElement("ügyfél");  
    ugyfel.setAttribute("Csomagszám", csomagszam);  
    ugyfel.setAttribute("futár", futar);
```

```
Element vasarlasIdopontjaElement = createElement(doc, "VásárlásIdőpontja", vasarlasIdopontja);
Element vasaroltTermekElement = createElement(doc, "VásároltTermék", vasaroltTermek);
Element telefonszamElement = createElement(doc, "Telefonszám", telefonszam);
```

```
Element cim = doc.createElement("cím");
Element iszElement = createElement(doc, "Isz", isz);
Element varosElement = createElement(doc, "Város", varos);
Element utcaElement = createElement(doc, "Utca", utca);
Element hazszamElement = createElement(doc, "házszám", hazszam);
```

```
cim.appendChild(iszElement);
cim.appendChild(varosElement);
cim.appendChild(utcaElement);
cim.appendChild(hazszamElement);
```

```
ugyfel.appendChild(vasarlasIdopontjaElement);
ugyfel.appendChild(vasaroltTermekElement);
ugyfel.appendChild(telefonszamElement);
ugyfel.appendChild(cim);
```

```
rootElement.appendChild(ugyfel);
}
```

```
private static Element createElement(Document doc, String name, String value) {
    Element element = doc.createElement(name);
    element.appendChild(doc.createTextNode(value));
    return element;
}
```

```
private static void printDocument(Document doc) {
    try {
        File outputFile = new File("output2.xml");
        PrintWriter writer = new PrintWriter(new FileWriter(outputFile, true));
```

```

// Kiírjuk az XML főgyökér elemét a konzolra és fájlba
Element rootElement = doc.getDocumentElement();
String rootName = rootElement.getTagName();
StringJoiner rootAttributes = new StringJoiner(" ");
NamedNodeMap rootAttributeMap = rootElement.getAttributes();

for (int i = 0; i < rootAttributeMap.getLength(); i++) {
    Node attribute = rootAttributeMap.item(i);
    rootAttributes.add(attribute.getNodeName() + "=\"" + attribute.getNodeValue() + "\"");
}

System.out.print("<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n");
writer.print("<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n");

System.out.print("<" + rootName + " " + rootAttributes.toString() + ">\n");
writer.print("<" + rootName + " " + rootAttributes.toString() + ">\n");

NodeList raktarList = doc.getElementsByTagName("raktár");
NodeList gyarihibasList = doc.getElementsByTagName("Gyárihiba");
NodeList brList = doc.getElementsByTagName("B_R");
NodeList beszallitoList = doc.getElementsByTagName("beszállító");
NodeList futarList = doc.getElementsByTagName("futár");
NodeList ugyfelList = doc.getElementsByTagName("ügyfél");

// Kiírjuk az XML-t a konzolra megtartva az eredeti formázást
printNodeList(raktarList, writer);
System.out.println("");
writer.println("");
printNodeList(gyarihibasList, writer);
System.out.println("");
writer.println("");
printNodeList(brList, writer);

```

```

System.out.println("");
writer.println("");
printNodeList(beszallitoList, writer);
System.out.println("");
writer.println("");
printNodeList(futarList, writer);
System.out.println("");
writer.println("");
printNodeList(ugyfellList, writer);

// Zárjuk le az XML gyökér elemét
System.out.println("</" + rootName + ">");
writer.append("</" + rootName + ">");

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

private static void printNodeList(NodeList nodeList, PrintWriter writer) {
    for (int i = 0; i < nodeList.getLength(); i++) {
        Node node = nodeList.item(i);
        printNode(node, 0, writer);
        System.out.println(""); // Üres sor hozzáadása az elemek között
        writer.println(""); // Üres sor hozzáadása a fájlban az elemek között
    }
}

private static void printNode(Node node, int indent, PrintWriter writer) {
    if (node.getNodeType() == Node.ELEMENT_NODE) {
        Element element = (Element) node;
        String nodeName = element.getTagName();

```

```

StringJoiner attributes = new StringJoiner(" ");
NamedNodeMap attributeMap = element.getAttributes();

for (int i = 0; i < attributeMap.getLength(); i++) {
    Node attribute = attributeMap.item(i);
    attributes.add(attribute.getNodeName() + "=\"" + attribute.getNodeValue() + "\"");
}

System.out.print(getIndentString(indent));
System.out.print("<" + nodeName + " " + attributes.toString() + ">");

writer.print(getIndentString(indent));
writer.print("<" + nodeName + " " + attributes.toString() + ">");

NodeList children = element.getChildNodes();
if (children.getLength() == 1 && children.item(0).getNodeType() == Node.TEXT_NODE) {
    System.out.print(children.item(0).getNodeValue());
    writer.print(children.item(0).getNodeValue());
} else {
    System.out.println();
    writer.println();
    for (int i = 0; i < children.getLength(); i++) {
        printNode(children.item(i), indent + 1, writer);
    }
    System.out.print(getIndentString(indent));
    writer.print(getIndentString(indent));
}
System.out.println("</" + nodeName + ">");
writer.println("</" + nodeName + ">");
}
}

```

// Segédmetódus az indentáláshoz

```
private static String getIndentString(int indent) {  
    StringBuilder sb = new StringBuilder();  
    for (int i = 0; i < indent; i++) {  
        sb.append(" "); // 2 spaces per indent level  
    }  
    return sb.toString();  
}  
}
```