JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Online videójáték bolt vásárlásjegyzéke

Készítette: **Mészáros Ákos**

Neptunkód: **BFNA2X**

Dátum: 2022.11.28.

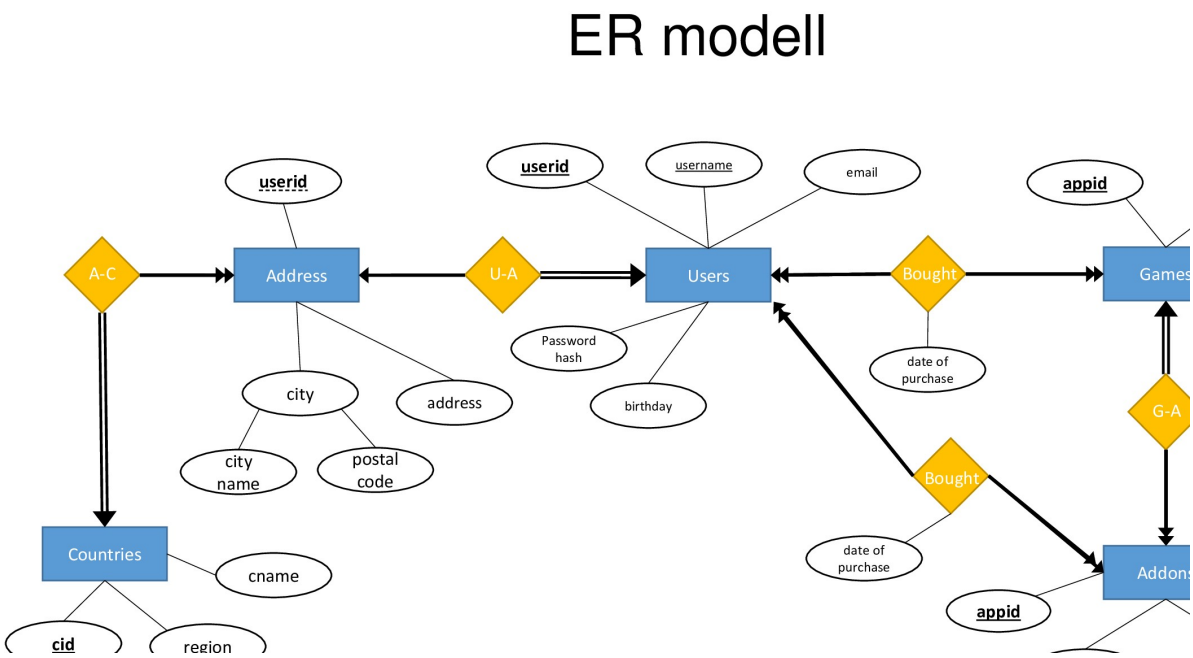
Tartalomjegyzék

A feladat leírása: Ebben a feladatban a Steam-hez hasonló online játékbolt vásárlás-nyilvántartó rendszerének a modelljét hoztam létre XML séma formájában.

1. feladat

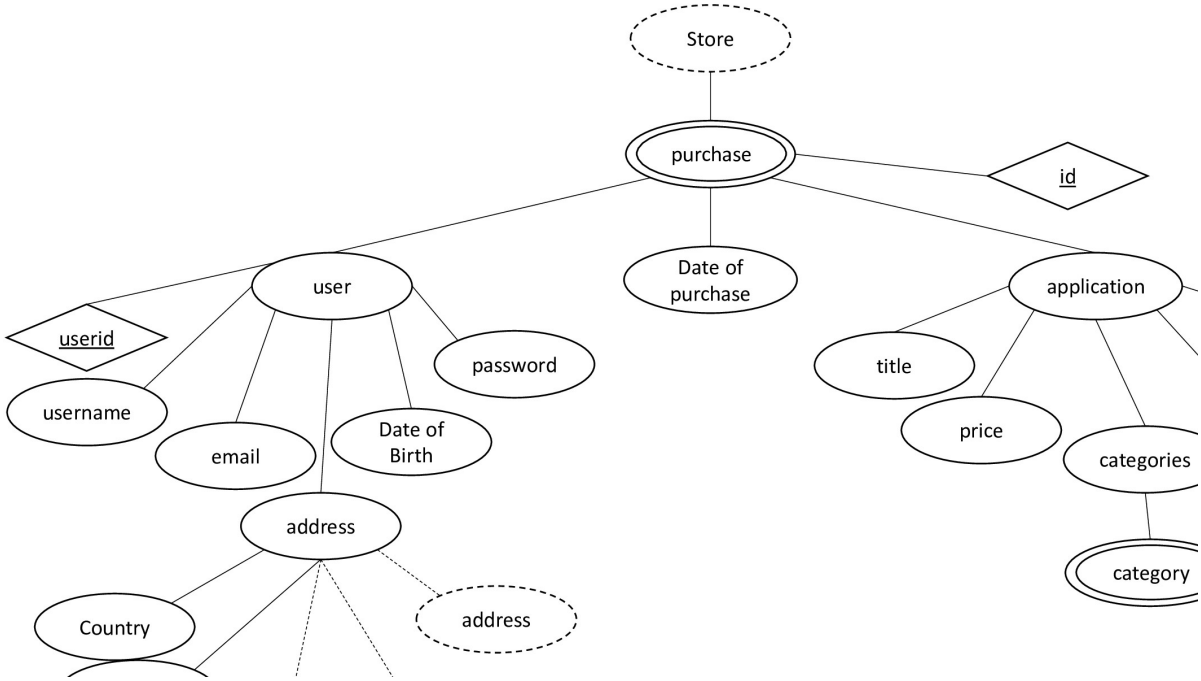
1a) Az adatbázis ER modell (Legyen legalább 5 egyed, többféle kapcsolat (1:1; 1:N; M:N), minden tulajdonság (normál, kulcs, összetett, többértékű). (Csak szerkesztő programmal rajzolt ábra megfelelő, szabványos szimbólumok használata)

Az ER modellt a PowerPoint alkalmazásban hoztam létre, egy korábbi félévben létrehozott modell átdolgozásával.



1b) Az adatbázis konvertálása XDM modellre (Csak szerkesztő programmal rajzolt ábra megfelelő, szabványos szimbólumok használata)

Az XDM modellt az ER modell alapján hoztam létre, az „Az XML modell” pdf segítségével. A Root node a ’Store’ ami több Purchase elemet tartalmaz. A purchase tartalmazza a felhasználó adatait, és az applikáció adatait, ami a Game és Addon összevonva.



1c) Az XDM modell alapján XML dokumentum készítése: (Ide kerül az XML kódja!)

Minden *többszörös előfordulási element*-ről legalább 3 példány készítése. A kódban megjegyzések használata.

Az XML dokumentumot a VS Code applikáciban hoztam létre.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-model href="XMLSchemaBFNA2X.xsd" type="application/xml"
schematypens="http://www.w3.org/2001/XMLSchema"?>
<!DOCTYPE store>
<store>
    <purchase id="p32153124563216">
        <user userid="8527438538">
            <username>St. John Paul II</username>
            <email>realpope@churchofjc.vt</email>

<password>eb939414aa6e6055fd0c2699b495e26138537f74e9c9ea47e67ab1ced536077d</pa
ssword>
            <DoB>1920-05-18</DoB>
            <address>
                <country>Vatican</country>
                <region>Europe</region>
                <city>Vatican City</city>
                <postalcode>00120</postalcode>
                <address>Saint Peter's Basilica</address>
            </address>
        </user>
        <application appid="1061910">
            <title>Metal: Hellsinger</title>
            <price>29.99€</price>
```

```
            <categories>
                <category>Action</category>
                <category>FPS</category>
                <category>Rythm</category>
                <category>Single Player</category>
            </categories>
        </application>
        <DoP>2022-10-11</DoP>
    </purchase>
    <purchase id="p25563252435235">
        <user userid="413423">
            <username>Spoopy McSpook</username>
            <email>potatoenjoyer42@gmail.com</email>

<password>c4e220591cadf3055b5b905cb9d273d614b4bcf44f1a96bf3346e8cfc59a235a</pa
ssword>
            <DoB>2000-09-25</DoB>
            <address>
                <country>Hungary</country>
                <region>Europe</region>
                <city>Sajólád</city>
                <postalcode>3572</postalcode>
            </address>
        </user>
        <application appid="223750">
            <title>DCS World</title>
            <price></price>
            <categories>
                <category>Simulation</category>
                <category>Free to Play</category>
                <category>Flight</category>
                <category>VR</category>
            </categories>
        </application>
        <DoP>2020-12-27</DoP>
    </purchase>
    <purchase id="p25563252452384">
        <user userid="413423">
            <username>Spoopy McSpook</username>
            <email>potatoenjoyer42@gmail.com</email>

<password>c4e220591cadf3055b5b905cb9d273d614b4bcf44f1a96bf3346e8cfc59a235a</pa
ssword>
            <DoB>2000-09-25</DoB>
            <address>
                <country>Hungary</country>
                <region>Europe</region>
                <city>Sajólád</city>
                <postalcode>3572</postalcode>
```

```
            </address>
        </user>
        <application appid="411950">
            <title>DCS: F/A-18C Hornet</title>
            <price>73.99€</price>
            <basegame>223750</basegame>
            <categories>
                <category>Simulation</category>
                <category>VR</category>
                <category>Military</category>
                <category>Realistic</category>
            </categories>
        </application>
        <DoP>2020-12-30</DoP>
    </purchase>
    <purchase id="p214378654856874">
        <user userid="32134">
            <username>Bob</username>
            <email>bob@idk.net</email>

<password>b0e1f85ba356628f1cc3b4b549e00fcd7841f5c6826a2200d9d99c0ccf70f19e</pa
ssword>
            <DoB>1996-01-03</DoB>
            <address>
                <country>Chile</country>
                <region>South America</region>
            </address>
        </user>
        <application appid="1434950">
            <title>HighFleet</title>
            <price>$20.99</price>
            <categories>
                <category>Single Player</category>
                <category>Strategy</category>
                <category>Indie</category>
                <category>Post-Apocalyptic</category>
            </categories>
        </application>
        <DoP>2021-11-21</DoP>
    </purchase>
</store>
```

1d) Az XML dokumentum alapján XMLSchema készítése (saját típusok, ref, key, keyref, speciális elemek) (Ide kerül az XML Schema kódja!)

Az XMLSchema-t a VS Code alapértelmezetten létrehozott schemájának átdolgozásával hoztam létre.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="store">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="purchase" maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="user">
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name="username"
type="xs:string" />
                                        <xs:element name="email">
                                            <xs:simpleType>
                                                <!--Email cím típus-->
                                                <xs:restriction
base="xs:string">
                                                    <xs:pattern value="[a-zA-
Z0-9]+@[a-zA-Z0-9.-]+.[a-zA-Z]+"></xs:pattern>
                                                </xs:restriction>
                                            </xs:simpleType>
                                        </xs:element>
                                        <xs:element name="password"
type="xs:string" />
                                        <xs:element name="DoB" type="xs:date"
/>
                                        <xs:element name="address">
                                            <xs:complexType>
                                                <xs:sequence>
                                                    <xs:element name="country"
type="xs:string" />
                                                    <xs:element name="region">
                                                        <xs:simpleType>
                                                            <!-- régió
enumerátor-->
                                                            <xs:restriction
base="xs:string">
<xs:enumeration value="Europe" />
<xs:enumeration value="North America" />
<xs:enumeration value="South America" />
<xs:enumeration value="Russia" />
<xs:enumeration value="Middle East" />
```

```
<xs:enumeration value="Asia" />

<xs:enumeration value="SEA" />

<xs:enumeration value="North Africa" />

<xs:enumeration value="South Africa" />
                                                        </xs:restriction>
                                                    </xs:simpleType>
                                                </xs:element>
                                                <xs:element name="city"
minOccurs="0" type="xs:string" />
                                                <xs:element
name="postalcode" minOccurs="0" type="xs:string" />
                                                <xs:element name="address"
minOccurs="0" type="xs:string" />
                                            </xs:sequence>
                                        </xs:complexType>
                                    </xs:element>
                                            </xs:sequence>
                                    <xs:attribute name="userid"
type="xs:string" use="required" />
                                </xs:complexType>
                            </xs:element>
                            <xs:element name="application" minOccurs="0">
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name="title"
type="xs:string" />
                                        <xs:element name="price">
                                            <xs:simpleType>
                                                <!--ár formátum típus-->
                                                <xs:restriction
base="xs:string">
                                                    <xs:pattern value="|$[0-
9]{1,4}.[0-9]{2}|[0-9]{1,3}.[0-9]{2}€" />
                                                </xs:restriction>
                                            </xs:simpleType>
                                        </xs:element>
                                        <xs:element name="basegame"
minOccurs="0" type="xs:string" />
                                        <xs:element name="categories">
                                            <xs:complexType>
                                                <xs:sequence>
                                                    <xs:element
name="category" maxOccurs="unbounded" type="xs:string" />
                                                </xs:sequence>
                                            </xs:complexType>
```

```
                                    </xs:element>
                                </xs:sequence>
                                <xs:attribute name="appid"
type="xs:string" use="required" />
                            </xs:complexType>
                        </xs:element>
                        <xs:element name="DoP" minOccurs="0"
type="xs:date" />
                    </xs:sequence>
                    <!--ID típus (a többi elemnél többször is
előfordulhat)-->
                    <xs:attribute name="id" type="xs:ID" use="required"/>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>
</xs:schema>
```

2. feladat

A feladat egy DOM program készítése az XML dokumentum - *XMLNeptunkod.xml* – adatainak adminisztrálása alapján: (ide kerül a kód - comment együtt)

Project name: DOMParseNeptunkod

Package: hu.domparse.neptunkod

Class names: (DomReadNeptunkod, DomModifyNeptunkod, DomQueryNeptunkod)

2a) adatolvasás (kód – comment együtt) – fájlnév: *DOMReadNeptunkod.java*

A fájlt VS Code alkalmazás segítségével hoztam létre.

```
package hu.domparse.bfna2x;

import java.io.File;
import java.io.FileWriter;

import javax.xml.parsers.*;

import org.w3c.dom.*;

public class DomReadBFNA2X {

    public static void main(String argv[]) throws Exception {

        File xmlFile = new File("XMLTaskBFNA2X/XMLBFNA2X.xml"); //
fájlbeolvasás
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        DocumentBuilder dBuilder = factory.newDocumentBuilder();
```

```
        Document doc = dBuilder.parse(xmlFile);

        doc.getDocumentElement().normalize();

        WriteAllChildren(doc.getChildNodes(), 0); // rekurzív olvasás
megkezdése

        FileWriter wf = new FileWriter("XMLTaskBFNA2X/XMLBFNA2X.txt"); //
output fájl létrehozása

        StringBuilder sb = new StringBuilder();

        WriteChildrenToSB(sb, doc.getChildNodes(), 0);

        wf.write(sb.toString());

        wf.close();

    }


    private static void WriteAllChildren(NodeList childNodes, int indent) {
        for (int i = 0; i < childNodes.getLength(); i++) { // minden gyerek
node-on végig futva
            Node node = childNodes.item(i);

            if (node.getNodeType() == Node.ELEMENT_NODE) { // Element típusú
node-ok kiválasztása
                Element element = (Element) node;

                NamedNodeMap attributes = element.getAttributes(); //
attribútumok keresése

                indent(indent);
                if (attributes.getLength() > 0) // attribútumokkal rendelkező
Element-ek kiírása
                System.out.println("[" + element.getNodeName() + "] " +
attributes.item(0).toString() + " BEGIN");
                else // Attribútum nélküli elemek kiírása
                System.out.println("[" + element.getNodeName() + "] BEGIN");

                WriteAllChildren(element.getChildNodes(), indent + 1); // 1-el
behúzva rekurzív meghívása

                indent(indent);
                System.out.println("[" + element.getNodeName() + "] END");
            }
```

```
            if (node.getNodeType() == Node.TEXT_NODE) { // szöveggel
rendelkező node-ok tartalmának kiírása
                if (!node.getNodeValue().trim().isEmpty()) {
                    indent(indent);
                    System.out.println(node.getNodeValue());
                }
            }

        }
    }

    private static void WriteChildrenToSB(StringBuilder sb, NodeList
childNodes, int indent) {
        for (int i = 0; i < childNodes.getLength(); i++) {
            Node node = childNodes.item(i);

            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element element = (Element) node;

                NamedNodeMap attributes = element.getAttributes();

                indent(sb, indent);
                if (attributes.getLength() > 0)
                sb.append("[" + element.getNodeName() + "] " +
attributes.item(0).toString() + " BEGIN\n");
                else // Attribútum nélküli elemek kiírása
                sb.append("[" + element.getNodeName() + "] BEGIN\n");

                WriteChildrenToSB(sb, element.getChildNodes(), indent + 1); //
1-el behúzva rekurzív meghívása

                indent(sb, indent);
                sb.append("[" + element.getNodeName() + "] END\n");
            }

            if (node.getNodeType() == Node.TEXT_NODE) { // szöveggel
rendelkező node-ok tartalmának kiírása
                if (!node.getNodeValue().trim().isEmpty()) {
                    indent(sb, indent);
                    sb.append(node.getNodeValue()+"\n");
                }
            }

        }
    }

    private static void indent(int indent) { // behúzás megvalósítása
        for (int k = 0; k < indent; k++) {
            System.out.print("    ");
```

```
        }
    }

    private static void indent(StringBuilder sb, int indent) { // behúzás
megvalósítása
        for (int k = 0; k < indent; k++) {
            sb.append("    ");
        }
    }
}
```

2b) adatmódosítás (kód – comment együtt) – fájlnév: *DOMModifyNeptunkod.java*

```
package hu.domparse.bfna2x;

import java.io.File;

import javax.xml.parsers.*;
import javax.xml.transform.*;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import javax.xml.xpath.*;

import org.w3c.dom.*;

public class DomModifyNFNA2X {

    public static void main(String argv[]) throws Exception {
        File xmlFile = new File("XMLTaskBFNA2X/XMLBFNA2X.xml");
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        DocumentBuilder dBuilder = factory.newDocumentBuilder();
        Document doc = dBuilder.parse(xmlFile);

        doc.getDocumentElement().normalize();

        NodeList purchases = doc.getElementsByTagName("purchase");

        XPath xpath = XPathFactory.newInstance().newXPath();

        //első rekord attribútumának megváltoztatása
        NamedNodeMap attr = purchases.item(0).getAttributes();
        Node nodeAttr = attr.getNamedItem("id");
        nodeAttr.setTextContent("p333333333333333");

        //'413423' userid-ű user emailének megváltoztatása
```

```
        NodeList emails = (NodeList)
xpath.compile("//user[@userid=413423]/email").evaluate(doc,
XPathConstants.NODESET);

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

emails.item(i).getChildNodes().item(0).setTextContent("potataman@gmail.com");
        }


        //utolsó purchase applikációjának nevének megváltoztatása
        NodeList appname = (NodeList)
xpath.compile("//purchase[last()]/application/title").evaluate(doc,
XPathConstants.NODESET);

        for(int i = 0; i < appname.getLength(); i++){
            appname.item(i).getChildNodes().item(0).setTextContent("High
Fleet");
        }

        //'indie' category törlése
        NodeList categories = (NodeList)
xpath.compile("//categories[category='Indie']").evaluate(doc,
XPathConstants.NODESET);

        NodeList category = categories.item(0).getChildNodes();
        Element e = (Element) categories.item(0);

        for(int i = 0; i < category.getLength(); i++){
            Node node = category.item(i);
            if(node.getTextContent().contains("Indie")){
                e.removeChild(node);
            }
        }


        //'VR' kategória 'Virtual Reality'-ra átnevezése
        NodeList vrcat = (NodeList)
xpath.compile("//category[.='VR']").evaluate(doc, XPathConstants.NODESET);
        for(int i = 0; i < vrcat.getLength(); i++){
            vrcat.item(i).setTextContent("Virtual Reality");
        }


        TransformerFactory transfactory = TransformerFactory.newInstance();
        Transformer transf = transfactory.newTransformer();

        transf.setOutputProperty(OutputKeys.ENCODING, "UTF-8"); //fájlkiíró
konfigurálása
```

```
        transf.setOutputProperty(OutputKeys.INDENT, "yes");
        transf.setOutputProperty("{http://xml.apache.org/xslt}indent-amount",
"2");

        File wf = new File("XMLTaskBFNA2X/XMLModBFNA2X.xml"); // output fájl
létrehozása

        DOMSource src = new DOMSource(doc);

        StreamResult file = new StreamResult(wf);

        transf.transform(src, file);    //output fájlba írás



    }
}
```

2c) adatlekérdezés (kód – comment együtt) – fájlnév: *DOMQueryNeptunkod.java*

A fájlt VS Code segítségével hoztam létre, az órai feladatokra alapozva.

```
package hu.domparse.bfna2x;

import java.io.File;

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

import org.w3c.dom.*;

import javax.xml.xpath.*;

public class DOMQueryBFNA2X {
    public static void main(String[] args) throws Exception {

        File xmlFile = new File("XMLTaskBFNA2X/XMLBFNA2X.xml");

        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        DocumentBuilder dBuilder = factory.newDocumentBuilder();
        Document doc = dBuilder.parse(xmlFile);

        doc.getDocumentElement().normalize();

        XPath xpath = XPathFactory.newInstance().newXPath();

        String expression1 = "/store/purchase[1]";
        String expression2 = "//user[@userid=413423]";
        String expression3 = "//application";
```

```
        String expression4 =
"store/purchase/user[contains(email,'@gmail.com')]";
        String expression5 = "//purchase[.//category='VR']";

        // root elem első purchase eleme
        System.out.println("1. expression = " + expression1);
        NodeList nodelist = (NodeList)
xpath.compile(expression1).evaluate(doc, XPathConstants.NODESET);

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

            Node node = nodelist.item(i);
            System.out.println("Aktuális elem: " + node.getNodeName());
            WriteAllChildren(nodelist, 0);
            System.out.println();
        }

        // '413423' userid-vel rendelkező user adatai
        System.out.println("2. expression = " + expression2);
        NodeList nodelist2 = (NodeList)
xpath.compile(expression2).evaluate(doc, XPathConstants.NODESET);

        for (int i = 0; i < nodelist2.getLength(); i++) {

            Node node = nodelist2.item(i);
            System.out.println("Aktuális elem: " + node.getNodeName());
            WriteAllChildren(nodelist2, 0);
            System.out.println();
        }

        // minden application elem
        System.out.println("3. expression = " + expression3);
        NodeList nodelist3 = (NodeList)
xpath.compile(expression3).evaluate(doc, XPathConstants.NODESET);

        for (int i = 0; i < nodelist3.getLength(); i++) {

            Node node = nodelist3.item(i);
            System.out.println("Aktuális elem: " + node.getNodeName());
            WriteAllChildren(nodelist3, 0);
            System.out.println();
        }

        // '@gmail.com'-ra végződő emaillel rendelkező userek
        System.out.println("4. expression = " + expression4);
        NodeList nodelist4 = (NodeList)
xpath.compile(expression4).evaluate(doc, XPathConstants.NODESET);

        for (int i = 0; i < nodelist4.getLength(); i++) {
```

```
            Node node = nodelist4.item(i);
            System.out.println("Aktuális elem: " + node.getNodeName());
            WriteAllChildren(nodelist4, 0);
            System.out.println();
        }

        // 'VR' kategóriával rendelkező userek
        System.out.println("5. expression = " + expression5);
        NodeList nodelist5 = (NodeList)
xpath.compile(expression5).evaluate(doc, XPathConstants.NODESET);

        for (int i = 0; i < nodelist5.getLength(); i++) {

            Node node = nodelist5.item(i);
            System.out.println("Aktuális elem: " + node.getNodeName());
            WriteAllChildren(nodelist5, 0);
            System.out.println();
        }
    }

    private static void WriteAllChildren(NodeList childNodes, int indent) {
        for (int i = 0; i < childNodes.getLength(); i++) { // minden gyerek
node-on végig futva
            Node node = childNodes.item(i);

            if (node.getNodeType() == Node.ELEMENT_NODE) { // Element típusú
node-ok kiválasztása
                Element element = (Element) node;

                NamedNodeMap attributes = element.getAttributes(); //
attribútumok keresése

                indent(indent);
                if (attributes.getLength() > 0) // attribútumokkal rendelkező
Element-ek kiírása
                    System.out.println("[" + element.getNodeName() + "] " +
attributes.item(0).toString() + " BEGIN");
                else // Attribútum nélküli elemek kiírása
                    System.out.println("[" + element.getNodeName() + "]
BEGIN");

                WriteAllChildren(element.getChildNodes(), indent + 1); // 1-el
behúzva rekurzív meghívása

                indent(indent);
                System.out.println("[" + element.getNodeName() + "] END");
            }
```

```
            if (node.getNodeType() == Node.TEXT_NODE) { // szöveggel
rendelkező node-ok tartalmának kiírása
                if (!node.getNodeValue().trim().isEmpty()) {
                    indent(indent);
                    System.out.println(node.getNodeValue());
                }
            }

        }
    }

    private static void indent(int indent) { // behúzás megvalósítása
        for (int k = 0; k < indent; k++) {
            System.out.print("    ");
        }
    }

}
```