JEGYZŐKÖNYV

Adatkezelés XML környezetben

Autósiskolák nyilvántartása

Készítette: Nagy Máté

Neptunkód: U3ROFS

Dátum: 2023.11.09.

Tartalom

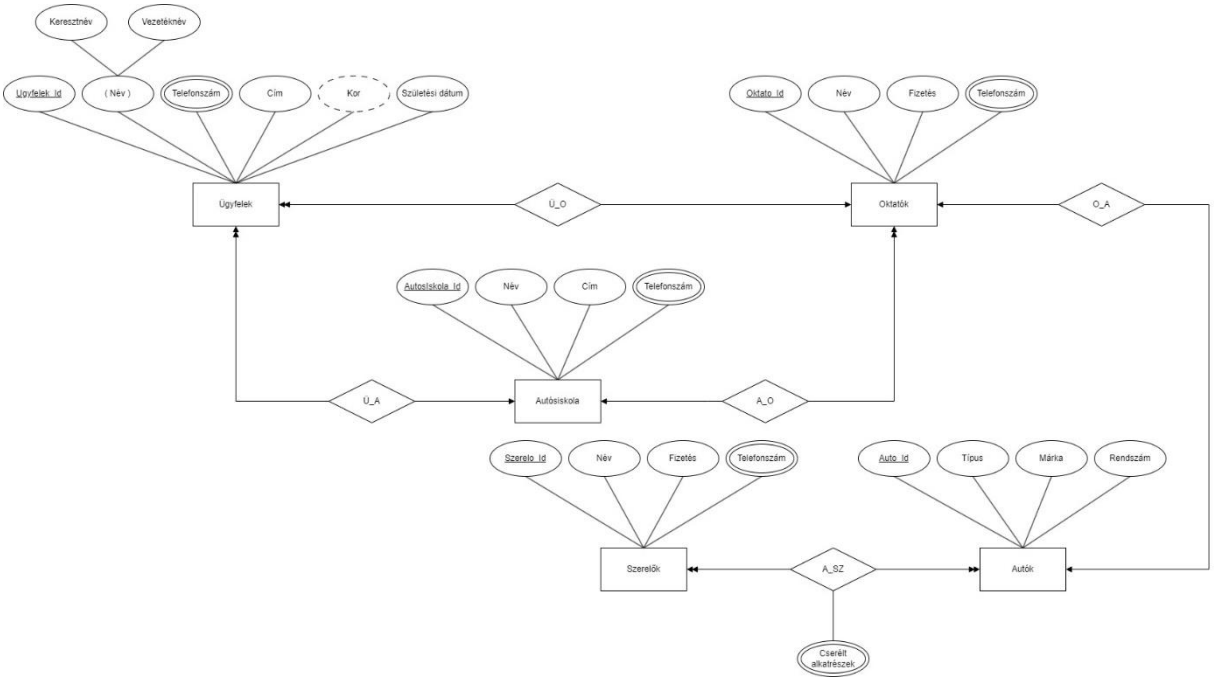
1.Feladat

1a) A feladat témája

A beadandó témája egy olyan adatbázis, amely az autósiskolákat és a hozzá kapcsolódó egyedeket tartja nyilván.

- **Autósiskola**
  - Autosiskola_Id: ez az elsődleges kulcs.
  - Név: ez az autósiskola neve.
  - Cím: ez az autósiskola címe.
  - Telefonszám: többértékű tulajdonság az autósiskola telefonszámai.
- **Ügyfelek**
  - Ugyfelek_Id: ez az elsődleges kulcs.
  - Név: több értékű tulajdonság(keresztnév, vezetéknév).
  - Cím: az ügyfelek címe.
  - Kor: az ügyfél életkora.
  - Születési dátum: születési dátum.
- **Oktatók**
  - Oktato_Id: ez az elsődleges kulcs
  - Név: oktató teljes neve.
  - Fizetés: oktató fizetése.
  - Telefonszám: többértékű tulajdonság az oktató telefonszámai.
- **Autók**
  - Auto_Id: ez az elsődleges kulcs.
  - Típus: az autó típusa.
  - Márka: az autó márkája
  - Rendszám: az autó rendszáma.
- **Szerelők**
  - Szerelo_Id: ez az elsődleges kulcs.
  - Név: a szerelő teljes neve.
  - Fizetés: szerelő fizetése.
  - Telefonszám: többértékű tulajdonság az szerelő telefonszámai.

A feladat ER modellje:



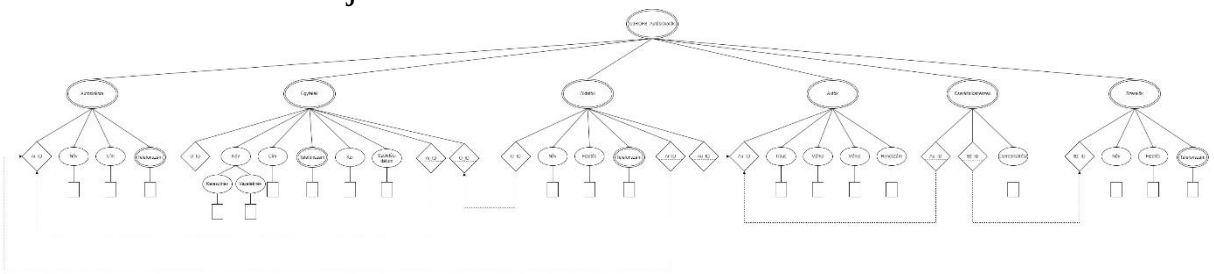
Az egyedek közötti kapcsolat

- Ü_A kapcsolat: egy autósiskolához több ügyfél tartozik 1:N.
- A_O kapcsolat: egy autósiskolához több oktató tartozik 1:N.
- Ü_O kapcsolat: egy oktatóhoz több ügyfél tartozik 1:N.
- O_A kapcsolat: egy oktatóhoz egy autó tartozik 1:1.
- A_SZ: több autó több szerelőhöz tartozik.

1b) Az ER-modell konvertálása XDM modellre

XDM modellnél háromféle jelölést alkalmazhatunk. Ezek az ellipszis, a rombusz, illetve a téglalap. Az ellipszis jelöli az elemeket minden egyedből elem lesz, ezen felül a tulajdonságokból is. A rombusz jelöli az attribútumokat, amelyek a kulcs tulajdonságokból keletkeznek. A téglalap jelöli a szöveget, amely majd az XML dokumentumban fog megjelenni. Azoknak az elemeknek, amelyek többször is előfordulhatnak, a jelölése dupla ellipszissel történik. Az idegenkulcsok és a kulcsok közötti kapcsolatot szaggatott vonalas nyíllal jelöljük.

A feladat XDM modellje:



1c) XML dokumentum készítése

Az XDM modell alapján az XML dokumentumot úgy készítettem el, hogy először is a root elementtel kezdtem, ami az U3ROFS_Autosiskolak volt.

A gyermek elemeiből 3-3 példányt hoztam létre, ezeknek az elemeknek az attribútumai közé tartoznak a kulcsok, illetve idegenkulcsok is, mindezek után ezeknek az elemeknek létrehoztam a többi gyermek elementet is.

XML dokumentum forráskódja

```
?xml version="1.0" encoding="utf-8"?>
<U3ROFS_Autosiskolak xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="XMLSchemaU3ROFS.xsd">

    <!-- Autosiskolák -->
    <Autosiskola ai_id="1">
        <nev>Go Car</nev>
        <cim>1111 Budapest, Kossuth Lajos utca 1.</cim>
        <telefon>06-70-123-4567</telefon>
        <telefon>06-30-123-4567</telefon>
    </Autosiskola>
    <Autosiskola ai_id="2">
        <nev>Guruljunk</nev>
        <cim>1111 Budapest, Petőfi utca 2.</cim>
        <telefon>06-70-123-4567</telefon>
    </Autosiskola>
    <Autosiskola ai_id="3">
        <nev>UNI</nev>
        <cim>3515, Miskolc, Egyetem út 1</cim>
        <telefon>06-70-123-4567</telefon>
        <telefon>06-30-123-4567</telefon>
    </Autosiskola>

    <!--Ügyfelek-->
    <Ugyfel u_id="1" ai_id="1" o_id="1">
        <nev>
            <vezeteknev>Nagy</vezeteknev>
```

```
            <keresztnev>Máté</keresztnev>
        </nev>
        <telefon>06-70-123-4567</telefon>
        <telefon>06-30-123-4567</telefon>
        <kor>18</kor>
        <szuletesi_datum>2003-01-01</szuletesi_datum>
    </Ugyfel>
    <Ugyfel u_id="2" ai_id="2" o_id="2">
        <nev>
            <vezeteknev>Gyáni</vezeteknev>
            <keresztnev>Kevin</keresztnev>
        </nev>
        <telefon>06-70-123-4567</telefon>
        <kor>18</kor>
        <szuletesi_datum>2003-01-01</szuletesi_datum>
    </Ugyfel>
    <Ugyfel u_id="3" ai_id="3" o_id="3">
        <nev>
            <vezeteknev>Kovács</vezeteknev>
            <keresztnev>Ádám</keresztnev>
        </nev>
        <telefon>06-70-123-4567</telefon>
        <telefon>06-30-123-4567</telefon>
        <kor>18</kor>
        <szuletesi_datum>2003-01-01</szuletesi_datum>
    </Ugyfel>

    <!-- Oktatók -->
    <Oktato o_id="1" ai_id="1">
        <nev>Kovács János</nev>
        <fizetes>300000</fizetes>
        <telefon>06-70-123-4567</telefon>
        <telefon>06-30-123-4567</telefon>
    </Oktato>
    <Oktato o_id="2" ai_id="2">
        <nev>Kiss János</nev>
        <fizetes>300000</fizetes>
        <telefon>06-70-123-4567</telefon>
    </Oktato>
    <Oktato o_id="3" ai_id="3">
        <nev>Nagy János</nev>
        <fizetes>300000</fizetes>
        <telefon>06-70-123-4567</telefon>
        <telefon>06-30-123-4567</telefon>
    </Oktato>

    <!-- Autók -->
    <Auto au_id="1" o_id="1">
        <rendszam>ABC-111</rendszam>
```

```
        <tipus>Astra</tipus>
        <marka>opel</marka>
    </Auto>
    <Auto au_id="2" o_id="2">
        <rendszam>ABC-222</rendszam>
        <tipus>Focus</tipus>
        <marka>Ford</marka>
    </Auto>
    <Auto au_id="3" o_id="3">
        <rendszam>ABC-333</rendszam>
        <tipus>Corolla</tipus>
        <marka>Toyota</marka>
    </Auto>

    <!-- Szerelők -->
    <Szerelo sz_id="1" au_id="1">
        <nev>Kovács Abdul</nev>
        <fizetes>400000</fizetes>
        <telefon>06-70-123-1111</telefon>
        <telefon>06-30-123-1112</telefon>
    </Szerelo>
    <Szerelo sz_id="2" au_id="2">
        <nev>Kiss Adorján</nev>
        <fizetes>420000</fizetes>
        <telefon>06-70-123-2222</telefon>
    </Szerelo>
    <Szerelo sz_id="3" au_id="3">
        <nev>Nagy Ferenc</nev>
        <fizetes>450000</fizetes>
        <telefon>06-70-123-3333</telefon>
        <telefon>06-30-123-3334</telefon>
    </Szerelo>

    <!-- Cserélt alkatrészek -->
    <cserealkatreszek au_id="1" sz_id="1">
        <cserealkatresz>fékbetét</cserealkatresz>
        <cserealkatresz>féktárcsa</cserealkatresz>
    </cserealkatreszek>
    <cserealkatreszek au_id="2" sz_id="2">
        <cserealkatresz>motor</cserealkatresz>
        <cserealkatresz>lengőkar</cserealkatresz>
    </cserealkatreszek>
    <cserealkatreszek au_id="3" sz_id="3">
        <cserealkatresz>szélvédő</cserealkatresz>
    </cserealkatreszek>

</U3ROFS_Autosiskolak>
```

1d) XMLSchema készítése

Az XML Schemám meghatározza az adatokat, mint például az iskola nevét, címét, és a telefon számokat, amelyeket egy TelefonszámTípus néven definiált saját típus szerint kell formázni. A SzuletesiDatumTipus típus korlátozza a születési dátumokat 1900 és 2005 közé. Továbbá komplex típusokat is definiál, mint az AutosiskolaTipus, UgyfelTipus, OktatoTipus, AutoTipus, SzereloTipus, és CserealkatreszTipus, melyek különféle attribútumokat és elemeket tartalmaznak. Az adatbázis integritásának megőrzése érdekében elsődleges (PK) és idegen kulcsok (FK) meghatározására kerül sor, valamint egyediség biztosítása (pl. minden autónak egyedülálló oktatója lehet) az Auto elem esetében. Az XML séma így biztosítja, hogy az adatok szerkezete és kapcsolatai érvényesek és következetesek legyenek.

Az XMLSchema forráskódja:

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

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

    <!-- Egyszerű típusok, saját típusok meghatározása -->
    <xs:element name="nev" type="xs:string" />
    <xs:element name="cim" type="xs:string" />
    <xs:element name="telefon" type="TelefonszámTípus" />
    <xs:element name="kor" type="xs:integer" />
    <xs:element name="szuletesi_datum" type="SzuletesiDatumTipus" />
    <xs:element name="fizetes" type="xs:integer" />
    <xs:element name="vezeteknev" type="xs:string" />
    <xs:element name="keresztnev" type="xs:string" />
    <xs:element name="rendszam" type="xs:string" />
    <xs:element name="tipus" type="xs:string" />
    <xs:element name="marka" type="xs:string" />
    <xs:element name="cserealkatresz" type="xs:string" />


    <xs:simpleType name="TelefonszámTípus">
        <xs:restriction base="xs:string">
            <xs:pattern value="\d{2}-\d{2}-\d{3}-\d{4}" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="SzuletesiDatumTipus">
        <xs:restriction base="xs:date">
            <xs:minInclusive value="1900-01-01" />
            <xs:maxInclusive value="2005-12-31" />
        </xs:restriction>
```

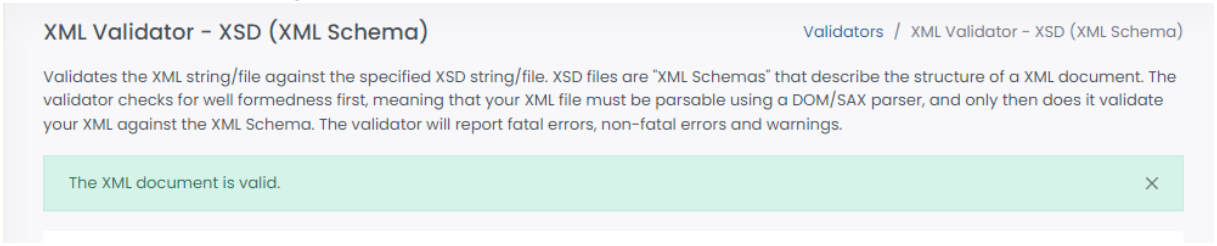
```
    </xs:simpleType>

    <!-- Komplex típusok meghatározása -->
    <xs:complexType name="AutosiskolaTipus">
        <xs:sequence>
            <xs:element name="nev" type="xs:string" />
            <xs:element name="cim" type="xs:string" />
            <xs:element ref="telefon" maxOccurs="unbounded" />
        </xs:sequence>
        <xs:attribute name="ai_id" type="xs:integer" />
    </xs:complexType>

    <xs:complexType name="UgyfelTipus">
        <xs:sequence>
            <xs:element name="nev">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element ref="vezeteknev" />
                        <xs:element ref="keresztnev" />
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element ref="telefon" maxOccurs="unbounded" />
            <xs:element ref="kor" minOccurs="1" />
            <xs:element ref="szuletesi_datum" />
        </xs:sequence>
        <xs:attribute name="u_id" type="xs:integer" />
        <xs:attribute name="ai_id" type="xs:integer" />
        <xs:attribute name="o_id" type="xs:integer" />
    </xs:complexType>

    <xs:complexType name="OktatoTipus">
        <xs:sequence>
            <xs:element ref="nev" />
            <xs:element ref="fizetes" />
            <xs:element ref="telefon" maxOccurs="unbounded" />
        </xs:sequence>
        <xs:attribute name="o_id" type="xs:integer" />
        <xs:attribute name="ai_id" type="xs:integer" />
    </xs:complexType>

    <xs:complexType name="AutoTipus">
        <xs:sequence>
            <xs:element ref="rendszam" />
            <xs:element ref="tipus" />
            <xs:element ref="marka" />
        </xs:sequence>
        <xs:attribute name="au_id" type="xs:integer" />
        <xs:attribute name="o_id" type="xs:integer" />
```

```
    </xs:complexType>

    <xs:complexType name="SzereloTipus">
        <xs:sequence>
            <xs:element ref="nev" />
            <xs:element ref="fizetes" />
            <xs:element ref="telefon" maxOccurs="unbounded" />
        </xs:sequence>
        <xs:attribute name="sz_id" type="xs:integer" />
        <xs:attribute name="au_id" type="xs:integer" />
    </xs:complexType>

    <xs:complexType name="CserealkatreszTipus">
        <xs:sequence>
            <xs:element ref="cserealkatresz" maxOccurs="unbounded" />
        </xs:sequence>
        <xs:attribute name="au_id" type="xs:integer" />
        <xs:attribute name="sz_id" type="xs:integer" />
    </xs:complexType>

    <!-- Gyökér elem meghatározása -->
    <xs:element name="U3ROFS_Autosiskolak">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Autosiskola" type="AutosiskolaTipus"
maxOccurs="unbounded" />
                <xs:element name="Ugyfel" type="UgyfelTipus"
maxOccurs="unbounded" />
                <xs:element name="Oktato" type="OktatoTipus"
maxOccurs="unbounded" />
                <xs:element name="Auto" type="AutoTipus" maxOccurs="unbounded"
/>
                <xs:element name="Szerelo" type="SzereloTipus"
maxOccurs="unbounded" />
                <xs:element name="cserealkatreszek" type="CserealkatreszTipus"
maxOccurs="unbounded" />
            </xs:sequence>
        </xs:complexType>

        <!--Elsődleges
        kulcsok-->
        <xs:key name="AutosiskolaPK">
            <xs:selector xpath="Autosiskola" />
            <xs:field xpath="@ai_id" />
        </xs:key>

        <xs:key name="UgyfelPK">
            <xs:selector xpath="Ugyfel" />
            <xs:field xpath="@u_id" />
```

```
        </xs:key>

        <xs:key name="OktatoPK">
            <xs:selector xpath="Oktato" />
            <xs:field xpath="@o_id" />
        </xs:key>

        <xs:key name="AutoPK">
            <xs:selector xpath="Auto" />
            <xs:field xpath="@au_id" />
        </xs:key>

        <xs:key name="SzereloPK">
            <xs:selector xpath="Szerelo" />
            <xs:field xpath="@sz_id" />
        </xs:key>

        <!-- Idegen kulcsok -->
        <xs:keyref name="UgyfelAIFK" refer="AutosiskolaPK">
            <xs:selector xpath="Ugyfel" />
            <xs:field xpath="@ai_id" />
        </xs:keyref>

        <xs:keyref name="UgyfelOFK" refer="OktatoPK">
            <xs:selector xpath="Ugyfel" />
            <xs:field xpath="@o_id" />
        </xs:keyref>

        <xs:keyref name="OktatoFK" refer="AutosiskolaPK">
            <xs:selector xpath="Oktato" />
            <xs:field xpath="@ai_id" />
        </xs:keyref>

        <xs:keyref name="AutoFK" refer="OktatoPK">
            <xs:selector xpath="Auto" />
            <xs:field xpath="@o_id" />
        </xs:keyref>

        <xs:keyref name="SzereloFK" refer="AutoPK">
            <xs:selector xpath="Szerelo" />
            <xs:field xpath="@au_id" />
        </xs:keyref>

        <xs:keyref name="CserealkatreszAUFK" refer="AutoPK">
            <xs:selector xpath="Cserealkatresz" />
            <xs:field xpath="@au_id" />
        </xs:keyref>

        <xs:keyref name="CserealkatreszSFK" refer="SzereloPK">
```

```
            <xs:selector xpath="Cserealkatresz" />
            <xs:field xpath="@sz_id" />
        </xs:keyref>

        <!-- 1:1 -->
        <xs:unique name="Auto">
            <xs:selector xpath="Auto" />
            <xs:field xpath="@o_id" />
        </xs:unique>
    </xs:element>
</xs:schema>
```

Validáció sikeressége:



2. Feladat

2a) DOM file beolvasás

ReadXMLDocument metódus: Ez a metódus felelős az XML fájl beolvasásáért. A filePath paraméterként megadott elérési útvonalon található fájlt nyitja meg.

DocumentBuilderFactory és DocumentBuilder: Ezek az osztályok a DOM parser alapvető elemei. A DocumentBuilderFactory létrehoz egy olyan környezetet, amelyben DocumentBuilder objektumokat hozhatunk létre. A DocumentBuilder feladata az XML fájl beolvasása és egy Document objektum létrehozása, amely a fájl DOM reprezentációját tartalmazza.

Normalizálás: A doc.getDocumentElement().normalize() hívás eltávolítja a felesleges whitespace karaktereket az XML dokumentumból, ami segít a későbbi feldolgozásban.

printDocument metódus: Ez a metódus felelős az XML dokumentum tartalmának kiírásáért. A dokumentum gyökérelemét és annak attribútumait írja ki először, majd a többi elemet rekurzívan.

NodeList és Node objektumok: A különböző XML elemek (például Autosiskola, Ugyfel, stb.) NodeList objektumokban vannak tárolva. Ezeket a listákat a getElementsByTagName metódus segítségével nyerjük ki. A Node objektumok reprezentálják az XML dokumentum egyes elemeit.

printNodeList és printNode metódusok: Ezek a metódusok végzik a NodeList objektumokon belüli Node objektumok kiírását. A printNode metódus kezeli az elemek attribútumait és gyermek elemeit, valamint gondoskodik a megfelelő behúzásokról (indentálásról).

DomReadU3ROFS forráskódja:

```
public class DomReadU3ROFS {
    public static void ReadXMLDocument(String filePath) {
        try {
```

```
            // Fájl beolvasása
            File inputFile = new File(filePath);
            // Ez létrehoz egy singleton objektumot, amely lehetővé teszi a
dokumentumok
            DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();
            // Ez a dokumentumépítő példányok létrehozására szolgál
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
            Document doc = dBuilder.parse(inputFile);
            doc.getDocumentElement().normalize();
            printDocument(doc);
        } catch (ParserConfigurationException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } catch (SAXException e) {
            e.printStackTrace();
        }
    }

    private static void printDocument(Document doc) {
        try {
            File outputFile = new File("XML_U3ROFS1.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 autosiskolaList =
doc.getElementsByTagName("Autosiskola");
            NodeList ugyfelList = doc.getElementsByTagName("Ugyfel");
```

```
            NodeList oktatoList = doc.getElementsByTagName("Oktato");
            NodeList autoList = doc.getElementsByTagName("Auto");
            NodeList szereloList = doc.getElementsByTagName("Szerelo");
            NodeList cserealkatreszekList =
doc.getElementsByTagName("cserealkatreszek");

            // Kiírjuk az XML-t a konzolra megtartva az eredeti formázást
            printNodeList(autosiskolaList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(ugyfelList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(oktatoList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(autoList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(szereloList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(cserealkatreszekList, 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, 1, writer);
            System.out.println("");
            writer.println("");
        }
    }

    private static void printNode(Node node, int indent, PrintWriter writer) {
        // Ha a node egy szöveg node, akkor kiírjuk a tartalmát
        if (node.getNodeType() == Node.ELEMENT_NODE) {
            Element element = (Element) node;
            String nodeName = element.getTagName();
            StringJoiner attributes = new StringJoiner(" ");
```

```
            NamedNodeMap attributeMap = element.getAttributes();
            // Kiírjuk az elem nevét és attribútumait
            for (int i = 0; i < attributeMap.getLength(); i++) {
                Node attribute = attributeMap.item(i);
                attributes.add(attribute.getNodeName() + "=\"" +
attribute.getNodeValue() + "\"");
            }
            // Kiírjuk az elem tartalmát
            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 + ">");
        }

    }

    private static String getIndentString(int indent) {
        StringBuilder sb = new StringBuilder();
        for (int i = 0; i < indent; i++) {
            // A szóközök száma, amivel indentálunk
            sb.append("  ");
        }
        return sb.toString();
      }
}
```

2b) DOM adatmódosítás

Az adat módosításhoz a következő segéd metódusokat használtam:

- getElementsByTagName: ezzel kértem le a különböző elemeket egy listába amelyeket módosítottam.
- setTextContext: ezzel módosítottam az elementhez tartozó szöveget.
- setAttribute: ezzel módosítottam az attribútumokat.

Ennek létrehoztam egy osztályt DomQueryU3ROFS néven, valamint egy metódust ModifyElement néven. Ez végig iterál az elements-eken és az összes egyedre lefuttatom a ModifyPrescribedElements-et ami módosítja a tulajdonságokat.

DomModifyU3ROFS osztály ModifyPrescribedElements függvény forráskódja:

```
public class DomModifyU3ROFS {

    public static void ModifyElement(String filePath) {
        // Fájl beolvasása
        try {
            File inputFile = new File(filePath);
            // Ez létrehoz egy singleton objektumot, amely lehetővé teszi a
dokumentumok
            // építését
            DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();
            // Ez a dokumentumépítő példányok létrehozására szolgál
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
            // Ez a dokumentum építésére szolgál
            Document doc = dBuilder.parse(inputFile);
            // A dokumentum normalizálása
            doc.getDocumentElement().normalize();
            ModifyPrescribedElements(doc);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private static void ModifyPrescribedElements(Document doc) {
        // Root Element lekérése
        NodeList nList = doc.getElementsByTagName("U3ROFS_Autosiskolak");
        Element element = (Element) nList.item(0);
        // Megváltoztatom az autosiskolákból az első elem nevét
        NodeList autosiskolaList =
element.getElementsByTagName("Autosiskola");
        Element autosiskola = (Element) autosiskolaList.item(0);
        autosiskola.getElementsByTagName("nev").item(0).setTextContent("Fast
Car");
        // Megváltoztatom az atributumot is
```

```
        autosiskola.setAttribute("ai_id", "4");

        // Például az első Ugyfel nevét változtatja meg
        NodeList ugyfelList = element.getElementsByTagName("Ugyfel");
        Element ugyfel = (Element) ugyfelList.item(0);

ugyfel.getElementsByTagName("vezeteknev").item(0).setTextContent("Kis");

ugyfel.getElementsByTagName("keresztnev").item(0).setTextContent("Gábor");
        ugyfel.setAttribute("ai_id", "4");

        // Az első Oktato fizetesét változtatja meg
        NodeList oktatoList = element.getElementsByTagName("Oktato");
        Element oktato = (Element) oktatoList.item(0);

oktato.getElementsByTagName("fizetes").item(0).setTextContent("350000");
        oktato.setAttribute("ai_id", "4");

        // Az első Auto markáját változtatja meg
        NodeList autoList = element.getElementsByTagName("Auto");
        Element auto = (Element) autoList.item(0);

auto.getElementsByTagName("marka").item(0).setTextContent("Chevrolet");

        // Az első Szerelo nevét változtatja meg
        NodeList szereloList = element.getElementsByTagName("Szerelo");
        Element szerelo = (Element) szereloList.item(0);
        szerelo.getElementsByTagName("nev").item(0).setTextContent("Kovács
Béla");

        // Az első cserealkatreszek cserealkatresz elemének tartalmát
változtatja meg
        NodeList cserealkatreszekList =
element.getElementsByTagName("cserealkatreszek");
        Element cserealkatreszek = (Element) cserealkatreszekList.item(0);

cserealkatreszek.getElementsByTagName("cserealkatresz").item(0).setTextContent
("kuplung");

    }
}
```

2c) DOM adat lekérdezés

Továbbra is a beolvasott elemeket használtam. Erre egy DomQueryU3ROFS osztályt hoztam létre. Összesen 6 lekérdezést készítettem.

- Összes autósiskola

- Összes Ugyfel adatainak kiíratása, akik egy bizonyos Autosiskola-hoz tartoznak
- Azoknak az Oktato-knak a neve és fizetése, akik bizonyos Autosiskola-ban tanítanak
- Az Auto elemek rendszam, tipus, és marka adatainak kiíratása
- Szerelők és az általuk szerelt autók, valamint a cserélt alkatrészek

DomQueryU3ROFS metódus forráskódja:

```
public class DomQueryU3ROFS {

    public static void QueryPrescribedDetails(String filePath) {
        Document doc = null;
        try {
            // Fájl beolvasása
            File inputFile = new File(filePath);
            // Ez létrehoz egy singleton objektumot, amely lehetővé teszi a
dokumentumok
            DocumentBuilderFactory dbFactory =
DocumentBuilderFactory.newInstance();
            // Ez a dokumentumépítő példányok létrehozására szolgál
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
            doc = dBuilder.parse(inputFile);
            doc.getDocumentElement().normalize();
        } catch (Exception e) {
            e.printStackTrace();
        }

        // Új szakasz kezdete a konzolon
        System.out.println();
        // Kiírja, hogy "Összes autósiskola:"
        System.out.println("Összes autósiskola:");
        // Lekéri az összes "Autosiskola" elemet az XML-ből
        NodeList autosiskolaList = doc.getElementsByTagName("Autosiskola");
        // Végigmegy az összes autósiskola elemen
        for (int i = 0; i < autosiskolaList.getLength(); i++) {
            Node node = autosiskolaList.item(i);
            // Ellenőrzi, hogy az elem tényleg elem típusú-e (nem szöveg vagy
komment)
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element autosiskola = (Element) node;
                // Kiírja az autósiskola ID-ját és nevét
                System.out.println("Autosiskola ID: " +
autosiskola.getAttribute("ai_id"));
                System.out.println("Név: " +
autosiskola.getElementsByTagName("nev").item(0).getTextContent());
            }
        }
```

```
        // Új szakasz kezdete a konzolon
        System.out.println();
        // Kiírja, hogy "Összes Ugyfel adatainak kiíratása, akik egy bizonyos
Autosiskola-hoz tartoznak"
        System.out.println("Összes Ugyfel adatainak kiíratása, akik egy
bizonyos Autosiskola-hoz tartoznak");
        // Lekéri az összes "Ugyfel" elemet az XML-ből
        NodeList ugyfelList = doc.getElementsByTagName("Ugyfel");
        // Végigmegy az összes ügyfél elemen
        for (int i = 0; i < ugyfelList.getLength(); i++) {
            Node node = ugyfelList.item(i);
            // Ellenőrzi, hogy az elem tényleg elem típusú-e és az "ai_id"
attribútum értéke "1"
            if (node.getNodeType() == Node.ELEMENT_NODE &&
"2".equals(((Element) node).getAttribute("ai_id"))) {
                Element ugyfel = (Element) node;
                // Kiírja az ügyfél nevét
                System.out.println("Ügyfél név: " +
ugyfel.getElementsByTagName("nev").item(0).getTextContent());
            }
        }

        // Új szakasz kezdete a konzolon
        System.out.println();
        // Kiírja, hogy "Azoknak az Oktato-knak a neve és fizetése, akik
bizonyos Autosiskola-ban tanítanak"
        System.out.println("Azoknak az Oktato-knak a neve és fizetése, akik
bizonyos Autosiskola-ban tanítanak");
        // Lekéri az összes "Oktato" elemet az XML-ből
        NodeList oktatoList = doc.getElementsByTagName("Oktato");
        // Végigmegy az összes oktató elemen
        for (int i = 0; i < oktatoList.getLength(); i++) {
            Node node = oktatoList.item(i);
            // Ellenőrzi, hogy az elem tényleg elem típusú-e és az "ai_id"
attribútum értéke "1"
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element oktato = (Element) node;
                if ("2".equals(oktato.getAttribute("ai_id"))) {
                    // Kiírja az oktató nevét és fizetését
                    System.out.println("Oktató neve: " +
oktato.getElementsByTagName("nev").item(0).getTextContent());
                    System.out.println("Fizetése: " +
oktato.getElementsByTagName("fizetes").item(0).getTextContent());
                }
            }
        }

        // Új szakasz kezdete a konzolon
```

```
        System.out.println();
        // Kiírja, hogy "Az Auto elemek rendszam, tipus, és marka adatainak
kiíratása"
        System.out.println("Az Auto elemek rendszam, tipus, és marka adatainak
kiíratása");
        // Lekéri az összes "Auto" elemet az XML-ből
        NodeList autoList = doc.getElementsByTagName("Auto");
        // Végigmegy az összes auto elemen
        for (int i = 0; i < autoList.getLength(); i++) {
            Node node = autoList.item(i);
            // Ellenőrzi, hogy az elem tényleg elem típusú-e
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element auto = (Element) node;
                // Kiírja az auto rendszámát, típusát, és márkáját
                System.out.println("Rendszám: " +
auto.getElementsByTagName("rendszam").item(0).getTextContent());
                System.out.println("Típus: " +
auto.getElementsByTagName("tipus").item(0).getTextContent());
                System.out.println("Márka: " +
auto.getElementsByTagName("marka").item(0).getTextContent());
            }
        }
        // Új szakasz kezdete a konzolon
        System.out.println();
        // Kiírja, hogy "Szerelők és az általuk szerelt autók, valamint a
cserélt alkatrészek:"
        System.out.println("Szerelők és az általuk szerelt autók, valamint a
cserélt alkatrészek:");

        // Lekéri az összes "Szerelo" elemet az XML-ből
        NodeList szereloList = doc.getElementsByTagName("Szerelo");
        for (int i = 0; i < szereloList.getLength(); i++) {
            Node szereloNode = szereloList.item(i);
            // Ellenőrzi, hogy az elem tényleg elem típusú-e
            if (szereloNode.getNodeType() == Node.ELEMENT_NODE) {
                Element szerelo = (Element) szereloNode;
                // Kiírja a szerelő nevét és az autó ID-ját, amit szerelt
                String szereloNev =
szerelo.getElementsByTagName("nev").item(0).getTextContent();
                String autoId = szerelo.getAttribute("au_id");

                System.out.println("Szerelő neve: " + szereloNev);
                System.out.println("Szerelt Auto ID: " + autoId);

                // Lekéri az összes "Auto" elemet az XML-ből
                for (int j = 0; j < autoList.getLength(); j++) {
                    Node autoNode = autoList.item(j);
                    // Ellenőrzi, hogy az elem tényleg elem típusú-e és az
auto ID egyezik-e
```

```
                    if (autoNode.getNodeType() == Node.ELEMENT_NODE &&
autoId.equals(((Element) autoNode).getAttribute("au_id"))) {
                        Element auto = (Element) autoNode;
                        // Kiírja az autó rendszámát, típusát és márkáját
                        System.out.println("Auto Rendszám: " +
auto.getElementsByTagName("rendszam").item(0).getTextContent());
                        System.out.println("Auto Típus: " +
auto.getElementsByTagName("tipus").item(0).getTextContent());
                        System.out.println("Auto Márka: " +
auto.getElementsByTagName("marka").item(0).getTextContent());
                    }
                }

                // Lekéri az összes "cserealkatreszek" elemet az XML-ből
                NodeList cserealkatreszekList =
doc.getElementsByTagName("cserealkatreszek");
                for (int k = 0; k < cserealkatreszekList.getLength(); k++) {
                    Node cserealkatreszekNode = cserealkatreszekList.item(k);
                    // Ellenőrzi, hogy az elem tényleg elem típusú-e és az
auto ID egyezik-e
                    if (cserealkatreszekNode.getNodeType() ==
Node.ELEMENT_NODE && autoId.equals(((Element)
cserealkatreszekNode).getAttribute("au_id"))) {
                        Element cserealkatreszek = (Element)
cserealkatreszekNode;
                        NodeList alkatresek =
cserealkatreszek.getElementsByTagName("cserealkatresz");
                        System.out.println("Cserélt alkatrészek:");
                        // Végigmegy a cserélt alkatrészek listáján
                        for (int l = 0; l < alkatresek.getLength(); l++) {
                            Node alkatresz = alkatresek.item(l);
                            // Ellenőrzi, hogy az elem tényleg elem típusú-e
                            if (alkatresz.getNodeType() == Node.ELEMENT_NODE)
{
                                // Kiírja az alkatrész nevét
                                System.out.println(" - " +
alkatresz.getTextContent());
                            }
                        }
                    }
                }
                // Új szakasz a konzolon
                System.out.println();
            }
        }
    }
}
```

Output:

```
Összes autósiskola:
Autosiskola ID: 1
Név: Go Car
Autosiskola ID: 2
Név: Guruljunk
Autosiskola ID: 3
Név: UNI

Összes Ugyfel adatainak kiíratása, akik egy bizonyos Autosiskola-hoz tartoznak
Ügyfél név:
            Gyáni
            Kevin


Azoknak az Oktato-knak a neve és fizetése, akik bizonyos Autosiskola-ban tanítanak
Oktató neve: Kiss János
Fizetése: 300000

Az Auto elemek rendszam, tipus, és marka adatainak kiíratása
Rendszám: ABC-111
Típus: Astra
Márka: opel
Rendszám: ABC-222
Típus: Focus
Márka: Ford
Rendszám: ABC-333
Típus: Corolla
Márka: Toyota

Szerelők és az általuk szerelt autók, valamint a cserélt alkatrészek:
Szerelő neve: Kovács Abdul
Szerelt Auto ID: 1
Auto Rendszám: ABC-111
Auto Típus: Astra
Auto Márka: opel
Cserélt alkatrészek:
 - fékbetét
 - féktárcsa

Szerelő neve: Kiss Adorján
Szerelt Auto ID: 2
Auto Rendszám: ABC-222
Auto Típus: Focus
Auto Márka: Ford
Cserélt alkatrészek:
 - motor
 - lengőkar

Szerelő neve: Nagy Ferenc
Szerelt Auto ID: 3
```

2d) DOM adatírás

WriteElementsToFileAndConsole metódus: Ez a metódus a dokumentum létrehozásáért és annak különböző elemekkel való feltöltéséért felelős. A DocumentBuilderFactory és DocumentBuilder segítségével hozza létre az üres XML dokumentumot, amelybe különböző elemeket ad hozzá.

Root Element: A U3ROFS_Autosiskolak nevű gyökérelemet hoz létre, amely az XML dokumentum alapját képezi. Ehhez attribútumokat is rendel, mint például a névtér és a séma helye.

Autósiskolák, ügyfelek, oktatók, autók, szerelők és cserealkatrészek hozzáadása: A kód tartalmaz különböző segédmetódusokat (addAutosiskola, addUgyfel, stb.), amelyekkel ezeket az entitásokat hozzáadja a dokumentumhoz. Ezek az entitások különböző attribútumokkal és gyermek elemekkel rendelkeznek.

createElement segédmetódus: Ez a metódus egy új XML elemet hoz létre a megadott névvel és értékkel. Hasznos, mert egyszerűsíti az elemek létrehozásának folyamatát.

Dokumentum kiírása: A Transformer osztály segítségével a kód kiírja az XML dokumentumot egy fájlba, valamint a konzolra is. A kiírás során figyelembe veszi az encodingot és az indentálást is, hogy az eredmény olvasható és jól formázott legyen.

printDocument, printNodeList, és printNode metódusok: Ezek a metódusok felelősek a dokumentum és annak egyes elemeinek kiírásáért. A printNode metódus rekurzívan járja be az XML elemeket, és gondoskodik a megfelelő behúzásokról.

DomWriteU3ROFS forráskódja:

```
public static void WriteElementsToFileAndConsole() {
        try {
            // Előkészítjük a dokumentumot
            DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
            // Ez a dokumentumépítő példányok létrehozására szolgál
            DocumentBuilder builder = factory.newDocumentBuilder();
            // Ez a dokumentum építésére szolgál
            Document doc = builder.newDocument();
            // Root Element létrehozása
            Element rootElement = doc.createElement("U3ROFS_Autosiskolak");
            rootElement.setAttribute("xmlns:xsi",
"http://www.w3.org/2001/XMLSchema-instance");
            rootElement.setAttribute("xsi:noNamespaceSchemaLocation",
"XMLSchemaU3ROFS.xsd");
            doc.appendChild(rootElement);
            // Autosiskolak létrehozása
            addAutosiskola(doc, rootElement, "1", "Go Car", "1111 Budapest,
Kossuth Lajos utca 1.",
                    Arrays.asList("06-70-123-4567", "06-30-123-4567"));
```

```
            addAutosiskola(doc, rootElement, "2", "Guruljunk", "1111 Budapest,
Petőfi utca 2.",
                    Arrays.asList("06-70-123-4567"));
            addAutosiskola(doc, rootElement, "3", "UNI", "3515, Miskolc,
Egyetem út 1",
                    Arrays.asList("06-70-123-4567", "06-30-123-4567"));
            // Ugyfel létrehozása
            addUgyfel(doc, rootElement, "1", "1", "1", "Nagy", "Máté",
                    Arrays.asList("06-70-123-4567", "06-30-123-4567"), "18",
"2003-01-01");
            addUgyfel(doc, rootElement, "2", "2", "2", "Gyáni", "Kevin",
Arrays.asList("06-70-123-4567"), "18",
                    "2003-01-01");
            addUgyfel(doc, rootElement, "3", "3", "3", "Kovács", "Ádám",
                    Arrays.asList("06-70-123-4567", "06-30-123-4567"), "18",
"2003-01-01");
            // Oktato létrehozása
            addOktato(doc, rootElement, "1", "1", "Kovács János", "300000",
                    Arrays.asList("06-70-123-4567", "06-30-123-4567"));
            addOktato(doc, rootElement, "2", "2", "Kiss János", "300000",
Arrays.asList("06-70-123-4567"));
            addOktato(doc, rootElement, "3", "3", "Nagy János", "300000",
                    Arrays.asList("06-70-123-4567", "06-30-123-4567"));
            // Auto létrehozása
            addAuto(doc, rootElement, "1", "1", "ABC-111", "Astra", "Opel");
            addAuto(doc, rootElement, "2", "2", "ABC-222", "Focus", "Ford");
            addAuto(doc, rootElement, "3", "3", "ABC-333", "Corolla",
"Toyota");
            // Szerelo létrehozása
            addSzerelo(doc, rootElement, "1", "1", "Kovács Abdul", "400000",
                    Arrays.asList("06-70-123-1111", "06-30-123-1112"));
            addSzerelo(doc, rootElement, "2", "2", "Kiss Adorján", "420000",
Arrays.asList("06-70-123-2222"));
            addSzerelo(doc, rootElement, "3", "3", "Nagy Ferenc", "450000",
                    Arrays.asList("06-70-123-3333", "06-30-123-3334"));

            // Cserealkatreszek létrehozása
            addCserealkatreszek(doc, rootElement, "1", "1",
Arrays.asList("fékbetét", "féktárcsa"));
            addCserealkatreszek(doc, rootElement, "2", "2",
Arrays.asList("motor", "lengőkar"));
            addCserealkatreszek(doc, rootElement, "3", "3",
Arrays.asList("szélvédő"));

            // Dokumentum mentése
            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");

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

    private static void addAutosiskola(Document doc, Element rootElement,
String ai_id, String nev, String cim,
            List<String> telefonok) {
        Element autosiskola = doc.createElement("Autosiskola");
        autosiskola.setAttribute("ai_id", ai_id);

        Element nevElement = createElement(doc, "nev", nev);
        Element cimElement = createElement(doc, "cim", cim);
        autosiskola.appendChild(nevElement);
        autosiskola.appendChild(cimElement);

        for (String telefon : telefonok) {
            Element telefonElement = createElement(doc, "telefon", telefon);
            autosiskola.appendChild(telefonElement);
        }

        rootElement.appendChild(autosiskola);
    }

    private static void addUgyfel(Document doc, Element rootElement, String
u_id, String ai_id, String o_id,
            String vezeteknev, String keresztnev, List<String> telefonok,
String kor, String szuletesiDatum) {
        Element ugyfel = doc.createElement("Ugyfel");
        ugyfel.setAttribute("u_id", u_id);
        ugyfel.setAttribute("ai_id", ai_id);
        ugyfel.setAttribute("o_id", o_id);

        Element nevElement = doc.createElement("nev");
        Element vezeteknevElement = createElement(doc, "vezeteknev",
vezeteknev);
        Element keresztnevElement = createElement(doc, "keresztnev",
keresztnev);
        nevElement.appendChild(vezeteknevElement);
        nevElement.appendChild(keresztnevElement);

        ugyfel.appendChild(nevElement);
```

```
        for (String telefon : telefonok) {
            Element telefonElement = createElement(doc, "telefon", telefon);
            ugyfel.appendChild(telefonElement);
        }

        Element korElement = createElement(doc, "kor", kor);
        Element szuletesiDatumElement = createElement(doc, "szuletesi_datum",
szuletesiDatum);

        ugyfel.appendChild(korElement);
        ugyfel.appendChild(szuletesiDatumElement);

        rootElement.appendChild(ugyfel);
    }

    private static void addOktato(Document doc, Element rootElement, String
o_id, String ai_id, String nev,
            String fizetes, List<String> telefonok) {
        Element oktato = doc.createElement("Oktato");
        oktato.setAttribute("o_id", o_id);
        oktato.setAttribute("ai_id", ai_id);

        Element nevElement = createElement(doc, "nev", nev);
        Element fizetesElement = createElement(doc, "fizetes", fizetes);

        oktato.appendChild(nevElement);
        oktato.appendChild(fizetesElement);

        for (String telefon : telefonok) {
            Element telefonElement = createElement(doc, "telefon", telefon);
            oktato.appendChild(telefonElement);
        }

        rootElement.appendChild(oktato);
    }

    private static void addAuto(Document doc, Element rootElement, String
au_id, String o_id, String rendszam,
            String tipus, String marka) {
        Element auto = doc.createElement("Auto");
        auto.setAttribute("au_id", au_id);
        auto.setAttribute("o_id", o_id);

        Element rendszamElement = createElement(doc, "rendszam", rendszam);
        Element tipusElement = createElement(doc, "tipus", tipus);
        Element markaElement = createElement(doc, "marka", marka);

        auto.appendChild(rendszamElement);
        auto.appendChild(tipusElement);
```

```
        auto.appendChild(markaElement);

        rootElement.appendChild(auto);
    }

    private static void addSzerelo(Document doc, Element rootElement, String
sz_id, String au_id, String nev,
            String fizetes, List<String> telefonok) {
        Element szerelo = doc.createElement("Szerelo");
        szerelo.setAttribute("sz_id", sz_id);
        szerelo.setAttribute("au_id", au_id);

        Element nevElement = createElement(doc, "nev", nev);
        Element fizetesElement = createElement(doc, "fizetes", fizetes);

        szerelo.appendChild(nevElement);
        szerelo.appendChild(fizetesElement);

        for (String telefon : telefonok) {
            Element telefonElement = createElement(doc, "telefon", telefon);
            szerelo.appendChild(telefonElement);
        }

        rootElement.appendChild(szerelo);
    }

    private static void addCserealkatreszek(Document doc, Element rootElement,
String au_id, String sz_id,
            List<String> alkatreszek) {
        Element cserealkatreszek = doc.createElement("cserealkatreszek");
        cserealkatreszek.setAttribute("au_id", au_id);
        cserealkatreszek.setAttribute("sz_id", sz_id);

        for (String alkatresz : alkatreszek) {
            Element alkatreszElement = createElement(doc, "cserealkatresz",
alkatresz);
            cserealkatreszek.appendChild(alkatreszElement);
        }

        rootElement.appendChild(cserealkatreszek);
    }

    private static void printDocument(Document doc) {
        try {
            // Fájlba írás
            File outputFile = new File("XML_U3ROFS2.xml");
            // Írás a konzolra
            PrintWriter writer = new PrintWriter(new FileWriter(outputFile,
true));
```

```
            // Kiírja az XML főgyökér elemét a konzolra és fájlba
            Element rootElement = doc.getDocumentElement();
            String rootName = rootElement.getTagName();
            // A gyökér elem attribútumainak kiírása
            StringJoiner rootAttributes = new StringJoiner(" ");
            // A gyökér elem attribútumainak lekérése
            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");
            // A gyökér elem alatti elemek lekérése
            NodeList autosiskolaList =
doc.getElementsByTagName("Autosiskola");
            NodeList ugyfelList = doc.getElementsByTagName("Ugyfel");
            NodeList oktatoList = doc.getElementsByTagName("Oktato");
            NodeList autoList = doc.getElementsByTagName("Auto");
            NodeList szereloList = doc.getElementsByTagName("Szerelo");
            NodeList cserealkatreszekList =
doc.getElementsByTagName("cserealkatreszek");

            printNodeList(autosiskolaList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(ugyfelList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(oktatoList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(autoList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(szereloList, writer);
            System.out.println("");
            writer.println("");
            printNodeList(cserealkatreszekList, writer);

            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, 1, writer);
            System.out.println("");
            writer.println("");
        }
    }

    private static void printNode(Node node, int indent, PrintWriter writer) {
        // Ha az elem típusa ELEMENT_NODE, akkor kiírjuk az elem nevét és
attribútumait
        if (node.getNodeType() == Node.ELEMENT_NODE) {
            Element element = (Element) node;
            String nodeName = element.getTagName();
            StringJoiner attributes = new StringJoiner(" ");
            NamedNodeMap attributeMap = element.getAttributes();
            // Kiírjuk az elem nevét és attribútumait
            for (int i = 0; i < attributeMap.getLength(); i++) {
                Node attribute = attributeMap.item(i);
                attributes.add(attribute.getNodeName() + "=\"" +
attribute.getNodeValue() + "\"");
            }

            // Kiírjuk az elem nevét és attribútumait
            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 + ">");
        }
    }

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

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