JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat: Futóbolt

Készítette: **Karczub Roland**

Neptunkód: **KJSPMW**

Dátum: **2023.10.25.**

Miskolc,2023

Tartalomjegyzék:

A feladat leírása:

A féléves beadandó feladatom egy futással foglalkozó sportbolt, ahol interneten keresztül lehet főképp futáshoz termékeket rendelni. Több raktár tárolja az adott termékeket, és több alkalmazott szerepel benne. Az említett adatokat ahhoz, hogy nyilvántartsam, öt egyedet hoztam létre, amik a következőek:

- Vevő
- Rendelés
- Termék
- Raktár
- Alkalmazott

Elsősorban a **Vevő** egyedet szeretném bemutatni, ez tartalmazza a rendelő adatait, pontosítva egészen a címen belül az irányítószám, település, házszám, utcáig. Magába foglalja a telefonszámát, ami egy több értékű tulajdonságként lett létrehozva, nevét, email-címét és a személyi számát, ami ebben az egyedben megkapta az elsődleges kulcsot, mivel ez alapján van beazonosítva a vevő, ez az a tulajdonság, amiből minden vevőnek különböző van.

A második a sorban a **Rendelés** egyed, ami magába foglalja a rendeléseket, pontosabban egy Rendelés_ID-t, ami a konkrét rendelés száma, a nyomon követés céljából, ez maga az elsődleges kulcs. Tartalmazza még az egyed, a rendelés árát, a dátumot, amikor a rendelés leadásra került, illetve a rendelés típusát, hogy készpénzzel vagy bankkártyával fizetett a vevő. Egy vevő csak egy rendelést tud leadni.

A következő a **Termék** egyed, ami összeköttetésben áll a Rendelés egyeddel, és egy rendeléshez egyértelműen több termék tartozik. Tartalmazza a termék három féle típusát, ami a cipő, ruha, és kiegészítőket foglalja magába, illetve egy Termék_ID-t, ami az egyedben az elsődleges kulcs szerepét kapta meg, ez az ID a termékeket jelöli, mindegyik termék külön ID-val rendelkezik.

A beérkezett rendeléseket egy **Raktár** egyeddel kapcsolom össze, ahol több rendelés több raktárhoz tartozhat, tehát, hogyha az adott termék nincsen az egyik raktárban, viszont egy másikban készleten lehet. A Raktár

tartalmazza a termék típusát, a Raktár_ID-t, ami az elsődleges kulcs, ez tartja nyilván külön a raktárakat. Szerepel még a darabszám, hogy a rendelt termékből hány darab érhető el, illetve a raktárban lévő termék árát birtokolja még.

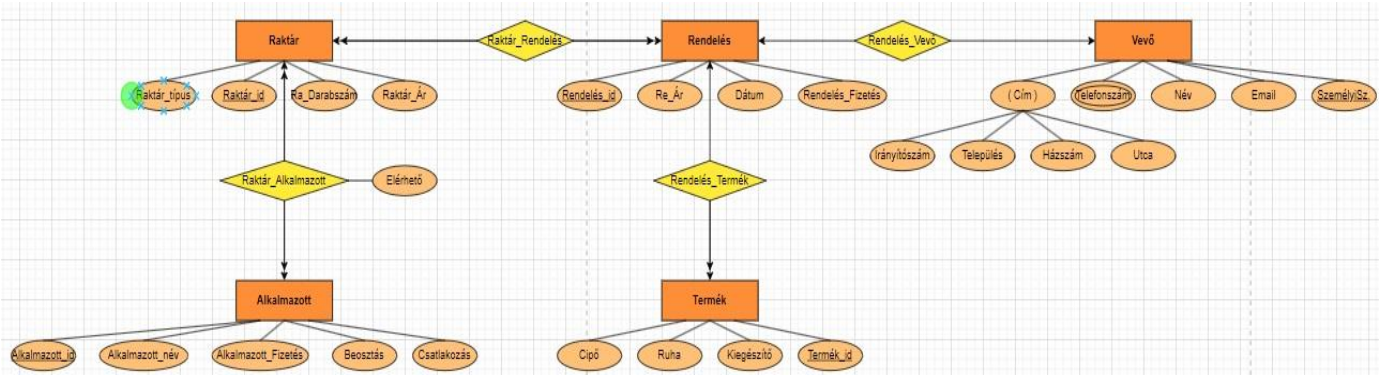
Az végső a sorban, az **Alkalmazott** egyed, ami több-több kapcsolatban áll a Raktárral, tehát több raktárhoz, több alkalmazott tartozik. Az Alkalmazott egyedben szerepel az Alkalmazott_ID, ami elsődleges kulcs, és ez az alkalmazottak egyedi azonosítója. Tartalmazza még az alkalmazott nevét,

fizetését, beosztását, illetve a csatlakozás dátumát, ami egyfajta tapasztalatot is lefed, hogy ha az alkalmazott már régebb óta dolgozik ott, akkor nagy valószínűséggel nagyobb tapasztalattal is rendelkezik, mint az, aki nemrég kezdett.

1.feladat:

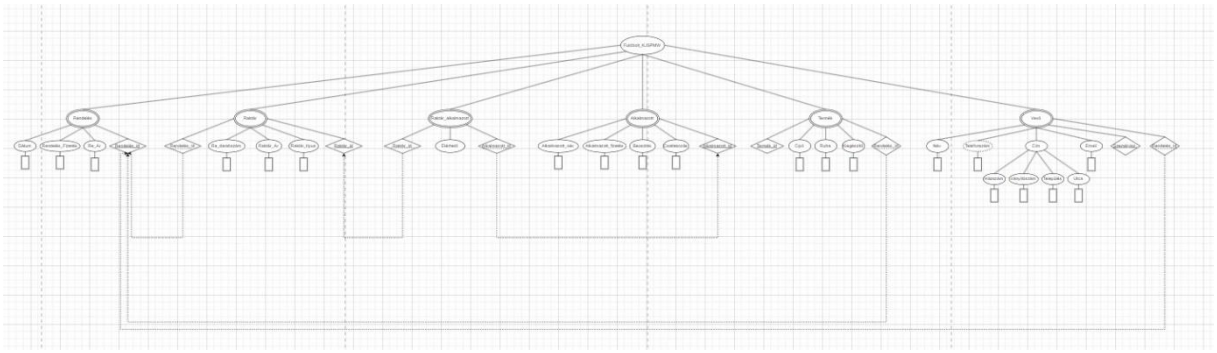
1a) Az adatbázis ER modell tervezése:

A Draw.io nevű szoftverrel készítettem el az ER-modellt. Először létrehoztam a különböző egyedeket, melyeket téglalappal jelöltem, majd ezek tulajdonságait elipszis alakzattal ábrázoltam. Ezt követte az egyedek közötti kapcsolatok kialakítása, ahol megfelelő élekkel kötöttem össze az egyedeket a kapcsolatokat szimbolizáló vonalakkal. A többértékű tulajdonságokat dupla falú elipszisekkel ábrázoltam. Figyeltem arra, hogy mindenféle kapcsolattípust érvényesítsek, és betartottam a minimum egyedszámot, amint azt a feladat kiírása előírta.



1b) Az adatbázis konvertálása XDM modellre:

A konvertáláskor figyelembe kell venni az ER modell során definiált kapcsolatokat, azok típusait (1:1, 1:N, N:M), illetve az entitások elsődleges kulcsait is. Minden egy-több kapcsolat esetében ahhoz az elsődleges kulcshoz kerül a szaggatott nyíl, ahol az ER modellben a többszerepel. Az ,,Eat" és a ,,Favor" kapcsolatokat kivéve, mindenhol 1:N kapcsolat szerepel az ER modellben, így az XDM mindenhol majdnem hasonlóan fog kinézni. Az N:M kapcsolat esetében egy új modellt veszünk fel, tulajdonsággal és primary key-el együtt természetesen, ahonnan a nyilakat a fő entitásokhoz húzzuk. A többágú tulajdonságok itt is több tulajdonsággal rendelkeznek, a többérték¶ tulajdonságok itt nem kapnak külön modellt. Az XDM modell gyökéreleme: Futobolt _KJSPMW



1c) Az XDM modell alapján XML dokumentum készítése:

Az XDM modell létrehozása után nekiláttam az XML dokumentum kialakításának, amely tükrözi a modellt. Mivel minden elem többször is előfordulhat, az előírásoknak megfelelően igyekeztem legalább három példányt létrehozni mindegyikből. A cím tulajdonság strukturáltan megjelenik az XML dokumentumban, amely gyerekelemek segítségével írja le a vevőnek az adatait.

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

    <!--Rendelés-->

    <rendeles rendeles_id="1">
        <datum>2021-10-01</datum>
        <rendeles_fizetes>utánvét</rendeles_fizetes>
        <rendeles_ar>12900</rendeles_ar>
    </rendeles>
    <rendeles rendeles_id="2">
        <datum>2020-12-24</datum>
        <rendeles_fizetes>kártya</rendeles_fizetes>
        <rendeles_ar>45230</rendeles_ar>
    </rendeles>
    <rendeles rendeles_id="3">
        <datum>2023-08-07</datum>
        <rendeles_fizetes>utánvét</rendeles_fizetes>
        <rendeles_ar>6700</rendeles_ar>
    </rendeles>

    <!--Raktár-->

    <raktar raktar_id="11" rendeles_id="1">
        <ra_darabszam>10</ra_darabszam>
        <raktar_ar>1000</raktar_ar>
        <raktar_tipus>új</raktar_tipus>
    </raktar>
    <raktar raktar_id="12" rendeles_id="2">
        <ra_darabszam>10</ra_darabszam>
        <raktar_ar>1000</raktar_ar>
        <raktar_tipus>új</raktar_tipus>
    </raktar>
    <raktar raktar_id="13" rendeles_id="3">
        <ra_darabszam>10</ra_darabszam>
        <raktar_ar>1000</raktar_ar>
        <raktar_tipus>új</raktar_tipus>
    </raktar>

    <!--Raktár-alkalmazott-->

    <raktar_alkalmazott raktar_id="11" alkalmazott_id = "41">
```

```
    <elerheto>igen</elerheto>
</raktar_alkalmazott>
<raktar_alkalmazott raktar_id="12" alkalmazott_id = "42">
    <elerheto>nem</elerheto>
</raktar_alkalmazott>
<raktar_alkalmazott raktar_id="13" alkalmazott_id = "43">
    <elerheto>igen</elerheto>
</raktar_alkalmazott>

<!--Alkalmazott-->

<alkalmazott alkalmazott_id = "41">
    <alkalmazott_nev>Ádám</alkalmazott_nev>
    <alkalmazott_fizetes>1000000</alkalmazott_fizetes>
    <alkalmazott_beosztas>segédmunkás</alkalmazott_beosztas>
    <alkalmazott_csatlakozas>2010-02-01</alkalmazott_csatlakozas>
</alkalmazott>
<alkalmazott alkalmazott_id = "42">
    <alkalmazott_nev>Béla</alkalmazott_nev>
    <alkalmazott_fizetes>500000</alkalmazott_fizetes>
    <alkalmazott_beosztas>vezető</alkalmazott_beosztas>
    <alkalmazott_csatlakozas>2018-01-23</alkalmazott_csatlakozas>
</alkalmazott>
<alkalmazott alkalmazott_id = "43">
    <alkalmazott_nev>Dani</alkalmazott_nev>
    <alkalmazott_fizetes>245000</alkalmazott_fizetes>
    <alkalmazott_beosztas>igazgató</alkalmazott_beosztas>
    <alkalmazott_csatlakozas>2011-06-12</alkalmazott_csatlakozas>
</alkalmazott>

<!--Termék-->

<termek termek_id = "51" rendeles_id = "1">
    <termek_cipo>félcipő</termek_cipo>
    <termek_ruha>póló</termek_ruha>
    <termek_kiegeszito>pulzuspánt</termek_kiegeszito>
</termek>
<termek termek_id = "52" rendeles_id = "2">
    <termek_cipo>bakancs</termek_cipo>
    <termek_ruha>kabát</termek_ruha>
    <termek_kiegeszito>óra</termek_kiegeszito>
</termek>
<termek termek_id = "53" rendeles_id = "3">
    <termek_cipo>csizma</termek_cipo>
    <termek_ruha>szoknya</termek_ruha>
    <termek_kiegeszito>karkötő</termek_kiegeszito>
</termek>

<!--Vevő-->
```

```xml
    <vevo id= "101" rendeles_id = "1">
        <nev>Lajos</nev>
        <telefonszam>06304567656</telefonszam>
        <cim>
            <iranyitoszam>1106</iranyitoszam>
            <telepules>Budapest</telepules>
            <utca>Árpád út</utca>
            <hazszam>14</hazszam>
        </cim>
        <email>kedvesbela@gmail.com</email>
        <szemelyigazolvany_szam>12345678</szemelyigazolvany_szam>
    </vevo>
    <vevo id= "102" rendeles_id = "2">
        <nev>Józsi</nev>
        <telefonszam>06301234567</telefonszam>
        <telefonszam>06304343312</telefonszam>
        <cim>
            <iranyitoszam>3900</iranyitoszam>
            <telepules>Szerencs</telepules>
            <utca>Magyar utca</utca>
            <hazszam>21</hazszam>
        </cim>
        <email>nagyjozsi@gmail.com</email>
        <szemelyigazolvany_szam>38264823</szemelyigazolvany_szam>
    </vevo>
    <vevo id= "103" rendeles_id = "3">
        <nev>Attila</nev>
        <telefonszam>06308766623</telefonszam>
        <cim>
            <iranyitoszam>3905</iranyitoszam>
            <telepules>Monok</telepules>
            <utca>Szabadság utca</utca>
            <hazszam>03</hazszam>
        </cim>
        <email>bereczkattila@gmail.com</email>
        <szemelyigazolvany_szam>32468734</szemelyigazolvany_szam>
    </vevo>
</Futobolt_KJSPMW>
```

1d) Az XML dokumentum alapján XMLSchema készítése:

Az XML dokumentum teljes struktúrájának létrehozása érdekében készítettem egy XMLSchema-t. Ez a séma leírja azokat az elemeket, amelyekre szükség van a dokumentum felépítéséhez, valamint meghatározza ezek típusait. Emellett szigorúan rögzíti bizonyos elemek számosságát, ami elengedhetetlen a struktúra megfelelő kialakításához. A XMLSchema definiálja az elsődleges és idegenkulcsokat is, amelyek összekapcsolják az elemeket, és így lehetővé teszik az egyszerűbb lekérdezéseket a dokumentumban.

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

<!--Egyszerű típusok kigyűjtése-->

<xs:element name = "datum" type="xs:date"/>
<xs:element name = "fizetes" type="xs:string"/>
<xs:element name = "ar" type="xs:integer"/>
<xs:element name = "darabszam" type="xs:integer"/>
<xs:element name = "tipus" type="xs:string"/>
<xs:element name = "alkalmazott_nev" type="xs:string"/>
<xs:element name = "alkalmazott_fizetes" type="xs:integer"/>
<xs:element name = "alkalmazott_beosztas" type="xs:string"/>
<xs:element name = "alkalmazott_csatlakozas" type="xs:date"/>
<xs:element name = "termek_tipus" type="xs:string"/>
<xs:element name = "email" type="xs:string"/>
<xs:element name = "telefonszam" type="TelefonszamTipus"/>

<xs:simpleType name = "telefonszamTipus">
    <xs:restriction base = "xs:string">
        <xs:pattern value = "[0-9]{2}-[0-9]{2}-[0-9]{2}"/>
    </xs:restriction>
</xs:simpleType>

<!--Komplex típusok kigyűjtése-->

<xs:complexType name = "rendelesTipus">
    <xs:sequence>
        <xs:element ref = "datum"/>
        <xs:element ref = "fizetes"/>
        <xs:element ref = "ar"/>
    </xs:sequence>
    <xs:attribute name = "rendeles_id" type="xs:integer" use="required"/>
</xs:complexType>

<xs:complexType name = "raktarTipus">
    <xs:sequence>
        <xs:element ref = "darabszam"/>
        <xs:element ref = "ar"/>
```

```
        <xs:element ref = "tipus"/>
    </xs:sequence>
    <xs:attribute name = "raktar_id" type="xs:integer" use="required"/>
    <xs:attribute name = "rendeles_id" type="xs:integer" use="required"/>
</xs:complexType>

<xs:complexType name = "raktarAlkalmazottTipus">
    <xs:sequence>
        <xs:element ref = "elerheto"/>
    </xs:sequence>
    <xs:attribute name = "raktar_id" type="xs:integer" use="required"/>
    <xs:attribute name = "alkalmazott_id" type="xs:integer" use="required"/>
</xs:complexType>

<xs:complexType name = "alkalmazottTipus">
    <xs:sequence>
        <xs:element ref = "alkalmazott_nev"/>
        <xs:element ref = "alkalmazott_fizetes"/>
        <xs:element ref = "alkalmazott_beosztas"/>
        <xs:element ref = "alkalmazott_csatlakozas"/>
    </xs:sequence>
    <xs:attribute name = "alkalmazott_id" type="xs:integer" use="required"/>
</xs:complexType>

<xs:complexType name = "termekTipus">
    <xs:sequence>
        <xs:element ref = "termek_tipus"/>
        <xs:element ref = "ar"/>
    </xs:sequence>
    <xs:attribute name = "termek_id" type="xs:integer" use="required"/>
    <xs:attribute name = "rendeles_id" type="xs:integer" use="required"/>
</xs:complexType>

<xs:complexType name = "vevokTipus">
    <xs:sequence>
        <xs:element ref = "email"/>
        <xs:element ref = "telefonszam"/>
        <xs:element ref = "nev"/>
        <xs:element name = "cim">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name = "telepules" type="xs:string"/>
                    <xs:element name = "utca" type="xs:string"/>
                    <xs:element name = "hazszam" type="xs:integer"/>
                    <xs:element name = "iranyitoszam" type="xs:integer"/>
                </xs:sequence>
            </xs:complexType>
    </xs:sequence>
    <xs:attribute name = "vevo_id" type="xs:integer" use="required"/>
```

```
    <xs:attribute name = "rendeles_id" type="xs:integer" use="required"/>
</xs:complexType>

<!--Gyökér elem definiálása-->

<xs:element name="futobolt_KJSPMW">
    <xs:complexType>
        <xs:sequence>
            <xs:element name = "rendeles" type="RendelesTipus"
maxOccurs="unbounded" minOccurs="0"/>
            <xs:element name = "raktar" type="RaktarTipus"
maxOccurs="unbounded" minOccurs="0"/>
            <xs:element name = "raktar_alkalmazott"
type="RaktarAlkalmazottTipus" maxOccurs="unbounded" minOccurs="0"/>
            <xs:element name = "alkalmazott" type="AlkalmazottTipus"
maxOccurs="unbounded" minOccurs="0"/>
            <xs:element name = "termek" type="TermekTipus"
maxOccurs="unbounded" minOccurs="0"/>
            <xs:element name = "vevo" type="VevokTipus" maxOccurs="unbounded"
minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>

    <!--Elsődleges kulcsok kigyűjtése-->

    <xs:key name = "rendeles_kulcs">
        <xs:selector xpath = "rendeles"/>
        <xs:field xpath = "@rendeles_id"/>
    </xs:key>

    <xs:key name = "raktar_kulcs">
        <xs:selector xpath = "raktar"/>
        <xs:field xpath = "@raktar_id"/>
    </xs:key>

    <xs:key name = "alkalmazott_kulcs">
        <xs:selector xpath = "alkalmazott"/>
        <xs:field xpath = "@alkalmazott_id"/>
    </xs:key>

    <xs:key name = "termek_kulcs">
        <xs:selector xpath = "termek"/>
        <xs:field xpath = "@termek_id"/>
    </xs:key>

    <xs:key name = "vevo_kulcs">
        <xs:selector xpath = "vevo"/>
        <xs:field xpath = "@vevo_id"/>
    </xs:key>
```

```
    <!--Idegen kulcsok kigyűjtése-->

    <xs:keyref name = "raktar_rendeles" refer = "rendeles_kulcs">
        <xs:selector xpath = "raktar"/>
        <xs:field xpath = "@rendeles_id"/>
    </xs:keyref>

    <xs:keyref name = "raktar_alkalmazott_raktar" refer = "raktar_kulcs">
        <xs:selector xpath = "raktar_alkalmazott"/>
        <xs:field xpath = "@raktar_id"/>
    </xs:keyref>

    <xs:keyref name = "raktar_alkalmazott_alkalmazott" refer =
"alkalmazott_kulcs">
        <xs:selector xpath = "raktar_alkalmazott"/>
        <xs:field xpath = "@alkalmazott_id"/>
    </xs:keyref>

    <xs:keyref name = "termek_rendeles" refer = "rendeles_kulcs">
        <xs:selector xpath = "termek"/>
        <xs:field xpath = "@rendeles_id"/>
    </xs:keyref>

    <xs:keyref name = "vevo_rendeles" refer = "rendeles_kulcs">
        <xs:selector xpath = "vevo"/>
        <xs:field xpath = "@rendeles_id"/>
    </xs:keyref>

    <!--Egy-egy kapcsolat kigyűjtése-->

    <xs:unique name = "rendeles_vevo_unique">
        <xs:selector xpath = "vevo"/>
        <xs:field xpath = "@rendeles_id"/>
    </xs:unique>

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

2.feladat:

2a) Adatolvasás:

A Java programom adatolvasó részéhez először beolvasom a dokumentumot a 1. feladat mappából. Ezután különböző függvények segítségével struktúrált formában kiírom az információkat a konzolra. A struktúráltság érdekében létrehoztam egyedi formázó függvényeket, amelyek a beolvasott dokumentumot, valamint a kimeneti fájlt kapják bemenetként. Minden egyes elemhez elkészítettem egy read függvényt, amely lekéri az adott elem összes alkotóelemét, ezeket megfelelő nevű változókban tárolja, majd ezekből felépíti a struktúrát. A struktúrát a printToFileAndConsole függvény hozza létre, amely a bemeneti elemet String-ként, a konzol elérési útját (System.out), valamint a kimeneti fájlt kapja meg bemenetként. A kimeneti fájl neve: XMLKJSPMW2.xml.

```
package hu.domparse.kjspmw;

import java.io.File;
import java.io.PrintStream;
import java.io.PrintWriter;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;

public class DomReadKJSPMW {
    public static void main(String args[]) {
        try {
            // DocumentFactory inicializálása
            DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();

            // DocumentBuilder inicializálása
            DocumentBuilder builder = factory.newDocumentBuilder();

            File file = new File("..\\..\\ElsoFeladat\\XMLKJSPMW.xml");

            // Dokumentum betöltése
            Document doc = builder.parse(file);
            doc.getDocumentElement().normalize();

            // Kimeneti fájl inicializálása
            PrintWriter outfile = new PrintWriter(new
File("..\\XMLKJSPMW2.xml"), "UTF-8");

            // XML adatok kiírása
            printToFileAndConsole("<?xml version=\"1.0\" encoding=\"UTF-
8\"?>", System.out, outfile);
            printToFileAndConsole(
```

```
                    "<Futobolt_KJSPMW
xmlns:xs=\"http://www.w3.org/2001/XMLSchema-instance\"
xs:noNamespaceSchemaLocation=\"XMLSchemaKJSPMW.xsd\">",
                    System.out, outfile);

            // Rendeles beolvasása
            readRendeles(doc, outfile);

            // Raktárak beolvasása
            readRaktarak(doc, outfile);

            // RaktarAlkalmazott kapcsolatok beolvasása
            readRaktarAlkalmazott(doc, outfile);

            // Alkalmazottak beolvasása
            readAlkalmazott(doc, outfile);

            // Termékek beolvasása
            readTermekek(doc, outfile);

            // Vevők beolvasása
            readVevok(doc, outfile);

            // XML gyökérelem lezárása
            printToFileAndConsole("</Futobolt_KJSPMW>", System.out, outfile);

            // Kimeneti fájl lezárása
            outfile.close();

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

    // Kiíró metódus
    private static void printToFileAndConsole(final String msg, PrintStream
console, PrintWriter file) {
        console.println(msg);
        file.println(msg);
    }

    // Eleme kiírás formázó metódus
    private static void printElement(String elementName, String content,
PrintWriter file) {
        printToFileAndConsole("        <" + elementName + ">" + content + "</"
+ elementName + ">", System.out,
                file);
    }
```

```
    // Cím kiírás formázó metódus
    private static void printCim(Element cimElement, PrintWriter file) {
        printToFileAndConsole("            <" + cimElement.getNodeName() +
">", System.out, file);
        printToFileAndConsole(
                "                <" +
cimElement.getElementsByTagName("iranyitoszam").item(0).getNodeName()
                        + ">" +
cimElement.getElementsByTagName("iranyitoszam").item(0).getTextContent() +
"</"
                        +
cimElement.getElementsByTagName("iranyitoszam").item(0).getNodeName() + ">",
                System.out, file);
        printToFileAndConsole(
                "                <" +
cimElement.getElementsByTagName("telepules").item(0).getNodeName()
                        + ">" +
cimElement.getElementsByTagName("telepules").item(0).getTextContent() + "</"
                        +
cimElement.getElementsByTagName("telepules").item(0).getNodeName() + ">",
                System.out, file);
        printToFileAndConsole(
                "                <" +
cimElement.getElementsByTagName("utca").item(0).getNodeName() + ">"
                        +
cimElement.getElementsByTagName("utca").item(0).getTextContent() + "</"
                        +
cimElement.getElementsByTagName("utca").item(0).getNodeName() + ">",
                System.out, file);
        printToFileAndConsole("                <" +
cimElement.getElementsByTagName("hazszam").item(0).getNodeName()
                + ">" +
cimElement.getElementsByTagName("hazszam").item(0).getTextContent() + "</"
                +
cimElement.getElementsByTagName("hazszam").item(0).getNodeName() + ">",
System.out, file);
        printToFileAndConsole("            </" + cimElement.getNodeName() +
">", System.out, file);
    }

    // Rendeléseket beolvasó metódus
    private static void readRendeles(Document document, PrintWriter file) {
        NodeList rendelesList = document.getElementsByTagName("rendeles");
        for (int temp = 0; temp < rendelesList.getLength(); temp++) {
            Node node = rendelesList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element rendelesElement = (Element) node;
                String datum =
rendelesElement.getElementsByTagName("datum").item(0).getTextContent();
```

```
                String rendelesId =
rendelesElement.getAttribute("rendeles_id");
                String fizetes =
rendelesElement.getElementsByTagName("rendeles_fizetes").item(0).getTextConten
t();
                String ar =
rendelesElement.getElementsByTagName("rendeles_ar").item(0).getTextContent();

                printToFileAndConsole("    <rendeles rendeles_id=\"" +
rendelesId + "\">", System.out, file);
                printElement("datum", datum, file);
                printElement("rendeles_fizetes", fizetes, file);
                printElement("rendeles_ar", ar, file);
                printToFileAndConsole("    </rendeles>", System.out, file);
            }
        }
    }

    // Raktárakat beolvasó metódus
    private static void readRaktarak(Document document, PrintWriter file) {
        NodeList raktarList = document.getElementsByTagName("raktar");
        for (int temp = 0; temp < raktarList.getLength(); temp++) {
            Node node = raktarList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element raktarElement = (Element) node;
                String raktar_id = raktarElement.getAttribute("raktar_id");
                String rendeles_id =
raktarElement.getAttribute("rendeles_id");
                String ra_darabszam =
raktarElement.getElementsByTagName("ra_darabszam").item(0).getTextContent();
                String raktar_ar =
raktarElement.getElementsByTagName("raktar_ar").item(0).getTextContent();
                String raktar_tipus =
raktarElement.getElementsByTagName("raktar_tipus").item(0).getTextContent();

                printToFileAndConsole("    <raktar raktar_id=\"" + raktar_id +
"\" rendeles_id=\""
                        + rendeles_id + "\">", System.out, file);
                printElement("ra_darabszam", ra_darabszam, file);
                printElement("raktar_ar", raktar_ar, file);
                printElement("raktar_tipus", raktar_tipus, file);
                printToFileAndConsole("    </raktar>", System.out, file);
            }

        }
    }

    // Raktar-Alkalmazott kapcsolatokat beolvasó metódus
```

```
    private static void readRaktarAlkalmazott(Document document, PrintWriter
file) {
        NodeList raktarAlkalmazottList =
document.getElementsByTagName("raktar_alkalmazott");
        for (int temp = 0; temp < raktarAlkalmazottList.getLength(); temp++) {
            Node node = raktarAlkalmazottList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element raktarAlkalmazottElement = (Element) node;
                String elerheto = raktarAlkalmazottElement
                        .getElementsByTagName("elerheto").item(0).getTextConte
nt();
                String raktar_id =
raktarAlkalmazottElement.getAttribute("raktar_id");
                String alkalmazott_id =
raktarAlkalmazottElement.getAttribute("alkalmazott_id");

                printToFileAndConsole("    <raktar_alkalmazott raktar_id=\"" +
raktar_id + "\" alkalmazott_id=\""
                        + alkalmazott_id + "\">", System.out, file);
                printToFileAndConsole("          <elerheto>" + elerheto
                        + "</elerheto>", System.out, file);
                printToFileAndConsole("    </raktar_alkalmazott>", System.out,
file);
            }
        }
    }

    // Alkalmazott beolvasó metódus
    private static void readAlkalmazott(Document document, PrintWriter file) {
        NodeList alkalmazottList =
document.getElementsByTagName("alkalmazott");
        for (int temp = 0; temp < alkalmazottList.getLength(); temp++) {
            Node node = alkalmazottList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element alkalmazottElement = (Element) node;
                String alkalmazott_nev =
alkalmazottElement.getElementsByTagName("alkalmazott_nev").item(0)
                        .getTextContent();
                String alkalmazott_id =
alkalmazottElement.getAttribute("alkalmazott_id");
                String alkalmazott_fizetes =
alkalmazottElement.getElementsByTagName("alkalmazott_fizetes").item(0)
                        .getTextContent();
                String alkalmazott_beosztas =
alkalmazottElement.getElementsByTagName("alkalmazott_beosztas").item(0)
                        .getTextContent();
                String alkalmazott_csatlakozas =
alkalmazottElement.getElementsByTagName("alkalmazott_csatlakozas")
                        .item(0).getTextContent();
```

```
                printToFileAndConsole("    <alkalmazott alkalmazott_id=\"" +
alkalmazott_id + "\">", System.out,
                        file);
                printElement("alkalmazott_nev", alkalmazott_nev, file);
                printElement("alkalmazott_fizetes", alkalmazott_fizetes,
file);
                printElement("alkalmazott_beosztas", alkalmazott_beosztas,
file);
                printElement("alkalmazott_csatlakozas",
alkalmazott_csatlakozas, file);
                printToFileAndConsole("    </alkalmazott>", System.out, file);
            }
        }
    }

    // Termékeket beolvasó metódus
    private static void readTermekek(Document document, PrintWriter file) {
        NodeList TermekekList = document.getElementsByTagName("termek");
        for (int temp = 0; temp < TermekekList.getLength(); temp++) {
            Node node = TermekekList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element TermekElement = (Element) node;
                String termek_id = TermekElement.getAttribute("termek_id");
                String rendeles_id =
TermekElement.getAttribute("rendeles_id");
                String termek_cipo =
TermekElement.getElementsByTagName("termek_cipo").item(0).getTextContent();
                String termek_ruha =
TermekElement.getElementsByTagName("termek_ruha").item(0).getTextContent();
                String termek_kiegeszito =
TermekElement.getElementsByTagName("termek_kiegeszito").item(0)
                        .getTextContent();

                printToFileAndConsole("    <termek termek_id=\"" + termek_id +
"\" rendeles_id=\""
                        + rendeles_id + "\">", System.out, file);
                printElement("termek_cipo", termek_cipo, file);
                printElement("termek_ruha", termek_ruha, file);
                printElement("termek_kiegeszito", termek_kiegeszito, file);
                printToFileAndConsole("    </termek>", System.out, file);

            }
        }
    }

    // Vevőket beolvasó metódus
    private static void readVevok(Document document, PrintWriter file) {
        NodeList vevokList = document.getElementsByTagName("vevo");
```

```
        for (int temp = 0; temp < vevokList.getLength(); temp++) {
            Node node = vevokList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element vevoElement = (Element) node;
                String vevo_id = vevoElement.getAttribute("id");
                String rendeles_id = vevoElement.getAttribute("rendeles_id");
                String vevo_nev =
vevoElement.getElementsByTagName("nev").item(0).getTextContent();
                String vevo_email =
vevoElement.getElementsByTagName("email").item(0).getTextContent();
                NodeList vevo_telefon =
vevoElement.getElementsByTagName("telefonszam");
                Element vevo_cim = (Element)
vevoElement.getElementsByTagName("cim").item(0);
                String vevo_szemelyi =
vevoElement.getElementsByTagName("szemelyigazolvany_szam").item(0)
                        .getTextContent();
                printToFileAndConsole("    <vevo id=\"" + vevo_id + "\"
rendeles_id=\""
                        + rendeles_id + "\">", System.out, file);
                printElement("nev", vevo_nev, file);
                for (int i = 0; i < vevo_telefon.getLength(); i++) {
                    printElement("telefonszam",
vevo_telefon.item(i).getTextContent(), file);
                }
                printCim(vevo_cim, file);
                printElement("email", vevo_email, file);
                printElement("szemelyigazolvany_szam", vevo_szemelyi, file);
                printToFileAndConsole("    </vevo>", System.out, file);

            }
        }
    }
}
```

2b) Adatmódosítás:

Az adatmódosító függvényben először betöltjük az első feladat XML dokumentumát, majd annak elemeit módosítjuk. Módosításra kerülnek az alkalmazott nevei ,illetve lekérem az összes terméket, és módosítom a termékeknek a ruha típusát. A módosított dokumentumot végül kiírom a konzolra.

```
package hu.domparse.kjspmw;

import java.io.IOException;
import java.io.StringWriter;
import java.util.Random;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.transform.OutputKeys;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerException;
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.xml.sax.SAXException;

public class DomModifyKJSPMW {
    public static void main(String args[]) {
        try {
            // DocumentBuilder inicializálása
            DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
            DocumentBuilder builder = factory.newDocumentBuilder();
            // Dokumentum beolvasása
            Document document =
builder.parse("..\\..\\ElsoFeladat\\XMLKJSPMW.xml");

            // Dokumentum módosítása
            modifyNodes(document);

            // Dokumentum kiírása a konzolra a módosítás után
            printXML(document);

        } catch (ParserConfigurationException | SAXException | IOException e)
{
            e.printStackTrace();
        }
    }
```

```
    private static void printXML(Document document) {
        try {
            // TransformerFactory és Transformer osztályok példányosítása
            TransformerFactory transformerFactory =
TransformerFactory.newInstance();
            Transformer transformer = transformerFactory.newTransformer();

            // Behúzás beállítása a transformerben
            transformer.setOutputProperty(OutputKeys.INDENT, "yes");

            // StringWriter osztály példányosítása, amiben eltároljuk a
dokumentumot
            StringWriter stringWriter = new StringWriter();

            // Dokumentum string-gé alakítása
            transformer.transform(new DOMSource(document), new
StreamResult(stringWriter));

            // Dokumentum kiírása a konzolra
            System.out.println(stringWriter.toString());
        } catch (TransformerException e) {
            e.printStackTrace();
        }
    }

    private static void modifyNodes(Document document) {
        // Lekéri az összes Alkalmazott node-ot
        NodeList alkalmazottNodeList =
document.getElementsByTagName("alkalmazott");

        // Végigiterál a node-okon
        for (int i = 0; i < alkalmazottNodeList.getLength(); i++) {
            Node node = alkalmazottNodeList.item(i);

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

                // Lekéri a Nev node-ot
                Node alkalmazottNevNode =
element.getElementsByTagName("alkalmazott_nev").item(0);

                // Módosítja a node értékét
                alkalmazottNevNode.setTextContent("alkalmazott" + (i + 1));
            }
        }
        // Lekéri az összes raktarAlkalmazott node-ot
        NodeList raktarAlkalmazottNodeList =
document.getElementsByTagName("raktar_alkalmazott");
```

```
        // Végigiterál a node-okon
        for (int i = 0; i < raktarAlkalmazottNodeList.getLength(); i++) {
            Node node = raktarAlkalmazottNodeList.item(i);

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

                Random random = new Random();

                int randomErtek = random.nextInt(5);
                // Módosítja az attribútum értékét
                element.setAttribute("raktar_id",
String.valueOf(randomErtek));
            }
        }

        // Lekéri az összes termek node-ot
        NodeList termekNodeList = document.getElementsByTagName("termek");

        // Végigiterál a node-okon
        for (int i = 0; i < termekNodeList.getLength(); i++) {
            Node node = termekNodeList.item(i);

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

                // Lekéri a Nev node-ot
                Node termekRuhaNode =
element.getElementsByTagName("termek_ruha").item(0);

                // Módosítja a node értékét
                termekRuhaNode.setTextContent("termek" + (i + 1));
            }
        }

    }
}
```

2c) Adatlekérdezés:

A feladatban készítettem három darab lekérdezést, van benne bonyolultabb és egyszerűbb is.

Az első lekérdezésben egy egyszerűvel kezdtem, lekértem az összes vevő nevét.

A második lekérdezésben a Raktár-Alkalmazott kapcsolatot kérem le, ahol elsőnek egy vizsgálattal megnézem, hogy az ID-k megegyeznek e, és ha igen, akkor megkapom a raktárnak a típusát, illetve az alkalmazott nevét.

A harmadik lekérdezésben pedig lekérdezem az összes vevőt, nekik a címűket, és megnézem ki lakik Szerencsen, és csak azokat íratom ki, akik Szerencsen laknak, és ilyenkor kiírom a vevő minden adatát.

```
package hu.domparse.kjspmw;

import java.io.IOException;

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

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

public class DomQueryKJSPMW {
    public static void main(String args[]) {
        try {
            // DocumentBuilder inicializálása
            DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
            DocumentBuilder builder = factory.newDocumentBuilder();
            // Dokumentum beolvasása
            Document document =
builder.parse("..\\..\\ElsoFeladat\\XMLKJSPMW.xml");

            // Üres sor beszúrása a konzolon, a jobb olvashatóság érdekében
            System.out.println();

            System.out.println("Összes vevő nevének lekérdezése:");
            // Összes vevő nevének lekérdezése
            NodeList vevoList = document.getElementsByTagName("vevo");
            // Végigiterál az összes Aruhaz Node-on
            for (int i = 0; i < vevoList.getLength(); i++) {
                Node node = vevoList.item(i);
                // Megnézi, hogy az elem elem típusú-e
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element vevo = (Element) node;
```

```
                    // Kiírja az vevő nevét
                    System.out.println(" Vevő neve: "
                            +
vevo.getElementsByTagName("nev").item(0).getTextContent());
                }
            }

            // Üres sor beszúrása a konzolon, a jobb olvashatóság érdekében
            System.out.println();

            System.out.println("Raktár-Alkalmazott kapcsolat lekérdezése:");

            // Raktár-alkalmazott kapcsolat lekérdezése
            NodeList raktarAlkalmazottKapcsolatList =
document.getElementsByTagName("raktar_alkalmazott");

            // Összes alkalmazott lekérdezése
            NodeList alkalmazottList =
document.getElementsByTagName("alkalmazott");

            // Összes raktár lekérdezése
            NodeList raktarList = document.getElementsByTagName("raktar");

            // Végigiterál az összes Raktar-Alkalmazott Node-on
            for (int i = 0; i < raktarAlkalmazottKapcsolatList.getLength();
i++) {
                Node node = raktarAlkalmazottKapcsolatList.item(i);

                // Inicializál egy stringet, ami a kiírandó sor lesz
                String kiirtSor = "";

                // Megnézi, hogy az elem elem típusú-e
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element raktarAlkalmazottElement = (Element) node;

                    // Végigiterál az összes alkalmazotton
                    for (int j = 0; j < alkalmazottList.getLength(); j++) {
                        Node alkalmazottNode = alkalmazottList.item(j);

                        // Megnézi, hogy az elem elem típusú-e
                        if (alkalmazottNode.getNodeType() ==
Node.ELEMENT_NODE) {
                            Element alkalmazott = (Element) alkalmazottNode;

                            // Megnézi, hogy az alkalmazott ID-ja megegyezik-e
a raktarAlkalmazott
                            // alkalmazott ID-jával az
                            // raktár-alkalmazott kapcsolatban
```

```
                        if (alkalmazott.getAttribute("alkalmazott_id")
                                .equals(raktarAlkalmazottElement.getAttrib
ute("alkalmazott_id"))) {
                            // Hozzáadja a kiírandó sorhoz az alkalmazott
nevét
                            kiirtSor += "Alkalmazott neve: "
                                    +
alkalmazott.getElementsByTagName("alkalmazott_nev").item(0).getTextContent()
                                    + " ";
                        }
                    }
                }

                // Végigiterál az összes raktáron
                for (int j = 0; j < raktarList.getLength(); j++) {
                    Node raktarNode = raktarList.item(j);
                    // Megnézi, hogy az elem elem típusú-e
                    if (raktarNode.getNodeType() == Node.ELEMENT_NODE) {
                        Element raktar = (Element) raktarNode;

                        // Megnézi, hogy a raktar ID-ja megegyezik-e a
raktarAlkalmazott raktar ID-jával
                        // az
                        // RaktárAlkalmazott kapcsolatban
                        if (raktar.getAttribute("raktar_id")
                                .equals(raktarAlkalmazottElement.getAttrib
ute("raktar_id"))) {
                            // Hozzáadja a kiírandó sorhoz a raktar
tipusat
                            kiirtSor += "Raktar Tipus: "
                                    +
raktar.getElementsByTagName("raktar_tipus").item(0).getTextContent();
                        }
                    }
                }

                // Kiírja a kapcsolatban álló alkalmazottak nevét és
raktárak tipusát
                System.out.println(kiirtSor);
            }
        }

        // Üres sor beszúrása a konzolon, a jobb olvashatóság érdekében
        System.out.println();

        System.out.println("Összes Szerencsen élő vevő lekérdezése:");
        // Összes vevő lekérdezése
        vevoList = document.getElementsByTagName("vevo");
        // Végigiterál az összes vevo Node-on
```

```
            for (int i = 0; i < vevoList.getLength(); i++) {
                Node node = vevoList.item(i);
                // Megnézi, hogy az elem elem típusú-e
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    Element vevo = (Element) node;
                    // Lekéri a címet
                    Node cimNode = vevo.getElementsByTagName("cim").item(0);
                    Element cim = (Element) cimNode;
                    // Lekéri a települést
                    Node telepulesNode =
cim.getElementsByTagName("telepules").item(0);

                    // Ha a vevő Szerencsen lakik, akkor kiírja az vevő
telefonszámát, nevét és
                    // címét
                    if (telepulesNode.getTextContent().equals("Szerencs")) {
                        System.out.println(" Vevő neve: "
                                +
vevo.getElementsByTagName("nev").item(0).getTextContent() + " Címe: "
                                +
cim.getElementsByTagName("iranyitoszam").item(0).getTextContent() + ", "
                                +
cim.getElementsByTagName("telepules").item(0).getTextContent() + ", "
                                +
cim.getElementsByTagName("utca").item(0).getTextContent() + ", "
                                +
cim.getElementsByTagName("hazszam").item(0).getTextContent() + " Telefonszam:
" +
                                vevo.getElementsByTagName("telefonszam").item(
0).getTextContent());
                    }
                }
            }

            // Üres sor beszúrása a konzolon, a jobb olvashatóság érdekében
            System.out.println();

        } catch (ParserConfigurationException | SAXException | IOException e)
{
            e.printStackTrace();
        }
    }
}
```

2d) Adatírás:

Ebben a feladatban rendeléseket, raktárakat, raktár-alkalmazott kapcsolatot, alkalmazottakat, termékeket, és vevőket hozok létre, és ezeket struktúrált formában jelenítem meg a konzolon, valamint fájlba. Itt a dokumentumot nem olvasom be, hanem saját függvények segítségével építem fel. Az "add" kezdetű függvények kiegészítik a dokumentumot a megfelelő elemekkel, tetszőleges névvel és további tulajdonságokkal. A "read" kezdetű függvények paraméterként kapják meg a feltöltött dokumentumot, majd kiírják a konzolra és fájlba. A kimeneti fájl neve: XMLKJSPMW3.xml.

```
package hu.domparse.kjspmw;

import java.io.File;
import java.io.PrintStream;
import java.io.PrintWriter;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;

public class DomWriteKJSPMW {
    public static void main(String args[]) {
        try {
            // DocumentFactory inicializálása
            DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();

            // DocumentBuilder inicializálása
            DocumentBuilder builder = factory.newDocumentBuilder();

            // Létrehozza a dokumentumot
            Document doc = builder.newDocument();

            // Gyökér elem létrehozása
            Element rootElement = doc.createElement("Futobolt_KJSPMW");

            rootElement.setAttribute("xmlns:xsi",
"http://www.w3.org/2001/XMLSchema-instance");
            rootElement.setAttribute("xsi:noNamespaceSchemaLocation",
"XMLSchemaKJSPMW.xsd");

            // Gyökér elem hozzáadása a dokumentumhoz
            doc.appendChild(rootElement);

            // Rendelések hozzáadása
            addRendeles(doc, rootElement, "1", "2020-11-13", "Utánvét",
"11500");
```

```
            addRendeles(doc, rootElement, "2", "2022-12-10", "Utánvét",
"5000");
            addRendeles(doc, rootElement, "3", "2021-01-02", "Kártya",
"7810");
            addRendeles(doc, rootElement, "4", "2021-05-09", "Utánvét",
"6788");
            addRendeles(doc, rootElement, "5", "2023-09-10", "Kártya",
"23000");

            // Raktárak hozzáadása
            addRaktar(doc, rootElement, "1", "1", "34", "1500", "új");
            addRaktar(doc, rootElement, "1", "2", "12", "500", "használt");
            addRaktar(doc, rootElement, "3", "3", "45", "60540", "új");
            addRaktar(doc, rootElement, "4", "4", "23", "12300", "régi");
            addRaktar(doc, rootElement, "5", "5", "67", "21000", "új");

            // Rakatár-Alkalmazott kapcsolatok hozzáadása
            addRaktarAlkalmazott(doc, rootElement, "1", "1", "igen");
            addRaktarAlkalmazott(doc, rootElement, "1", "2", "nem");
            addRaktarAlkalmazott(doc, rootElement, "2", "3", "igen");
            addRaktarAlkalmazott(doc, rootElement, "3", "4", "igen");
            addRaktarAlkalmazott(doc, rootElement, "3", "5", "nem");

            // Alkalmazottak hozzáadása
            addAlkalmazott(doc, rootElement, "1", "Kiss József", "350000",
"vezető", "2019-01-01");
            addAlkalmazott(doc, rootElement, "2", "Nagy Béla", "100000",
"segédmunkás", "2016-03-23");
            addAlkalmazott(doc, rootElement, "3", "Kovács Péter", "80000",
"segédmunkás", "2013-12-11");
            addAlkalmazott(doc, rootElement, "4", "Horváth Tibor", "760000",
"tulajdonos", "2010-05-30");
            addAlkalmazott(doc, rootElement, "5", "Kun Erika", "450000",
"adminisztrátor", "2012-10-05");

            // Termékek hozzáadása
            addTermek(doc, rootElement, "1", "1", "félcipő", "póló",
"pulzuspánt");
            addTermek(doc, rootElement, "2", "2", "cipő", "pulóver",
"sportóra");
            addTermek(doc, rootElement, "3", "3", "félcipő", "kabát",
"csősál");
            addTermek(doc, rootElement, "4", "4", "bakancs", "nadrág",
"kesztyű");
            addTermek(doc, rootElement, "1", "1", "félcipő", "aláfelső",
"compress szár");

            // Vevők hozzáadása
```

```
            addVevo(doc, rootElement, "1", "1", "Pista", "06-30-540-3344",
1106, "Budapest", "Kossuth Lajos utca", "12",
                    "kedvespista@gmail.com", "12312445");
            addVevo(doc, rootElement, "2", "2", "Józsi", "06-30-567-3455",
3900, "Szerencs", "Rózsa utca",
                    "34",
                    "jozsivagyok@gmail.com", "25468356");
            addVevo(doc, rootElement, "3", "3", "Attila", "06-20-345-3455",
3900, "Szerencs", "Damjanich utca",
                    "56",
                    "attila112@gmail.com", "34957543");
            addVevo(doc, rootElement, "4", "4", "Péter", "06-20-324-6788",
3903, "Bekecs", "Honvéd út", "23",
                    "nagypeter@gmail.com", "23497532");
            addVevo(doc, rootElement, "5", "5", "János", "06-70-566-3435",
3905, "Monok", "Szabadság utca", "78",
                    "janoskiss@gmail.com", "12343253");

            // Dokumentum kiírása, mentése
            File outputFile = new File("..\\XMLKJSPMW3.xml");

            PrintWriter file = new PrintWriter(outputFile, "UTF-8");

            printHeader(doc, file);

            readRendeles(doc, file);

            readRaktarak(doc, file);

            readRaktarAlkalmazott(doc, file);

            readAlkalmazott(doc, file);

            readTermekek(doc, file);

            readVevok(doc, file);

            printToFileAndConsole("</Futobolt_KJSPMW>", System.out, file);

            file.close();

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

    /* Kiíró, adatfeldolgozó rész kezdete */

    // Kiíró metódus
```

```
    private static void printToFileAndConsole(final String msg, PrintStream
console, PrintWriter file) {
        console.println(msg);
        file.println(msg);
    }

    // Fejrész elkészítő metódus
    private static void printHeader(Document doc, PrintWriter file) {
        printToFileAndConsole("<?xml version=\"1.0\" encoding=\"UTF-8\"?>",
System.out, file);
        printToFileAndConsole(
                "<Futobolt_KJSPMW xmlns:xs=\"http://www.w3.org/2001/XMLSchema-
instance\" xs:noNamespaceSchemaLocation=\"XMLSchemaKJSPMW.xsd\">",
                System.out, file);
    }

    // Eleme kiírás formázó metódus
    private static void printElement(String elementName, String content,
PrintWriter file) {
        printToFileAndConsole("        <" + elementName + ">" + content + "</"
+ elementName + ">", System.out,
                file);
    }

    // Cím kiírás formázó metódus
    private static void printCim(Element cimElement, PrintWriter file) {
        printToFileAndConsole("            <" + cimElement.getNodeName() +
">", System.out, file);
        printToFileAndConsole(
                "                <" +
cimElement.getElementsByTagName("iranyitoszam").item(0).getNodeName()
                        + ">" +
cimElement.getElementsByTagName("iranyitoszam").item(0).getTextContent() +
"</"
                        +
cimElement.getElementsByTagName("iranyitoszam").item(0).getNodeName() + ">",
                System.out, file);
        printToFileAndConsole(
                "                <" +
cimElement.getElementsByTagName("telepules").item(0).getNodeName()
                        + ">" +
cimElement.getElementsByTagName("telepules").item(0).getTextContent() + "</"
                        +
cimElement.getElementsByTagName("telepules").item(0).getNodeName() + ">",
                System.out, file);
        printToFileAndConsole(
                "                <" +
cimElement.getElementsByTagName("utca").item(0).getNodeName() + ">"
```

```
                        +
cimElement.getElementsByTagName("utca").item(0).getTextContent() + "</"
                        +
cimElement.getElementsByTagName("utca").item(0).getNodeName() + ">",
                System.out, file);
        printToFileAndConsole("                <" +
cimElement.getElementsByTagName("hazszam").item(0).getNodeName()
                + ">" +
cimElement.getElementsByTagName("hazszam").item(0).getTextContent() + "</"
                +
cimElement.getElementsByTagName("hazszam").item(0).getNodeName() + ">",
System.out, file);
        printToFileAndConsole("            </" + cimElement.getNodeName() +
">", System.out, file);
    }

    // Rendeléseket beolvasó metódus
    private static void readRendeles(Document document, PrintWriter file) {
        NodeList rendelesList = document.getElementsByTagName("rendeles");
        for (int temp = 0; temp < rendelesList.getLength(); temp++) {
            Node node = rendelesList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element rendelesElement = (Element) node;
                String datum =
rendelesElement.getElementsByTagName("datum").item(0).getTextContent();
                String rendelesId =
rendelesElement.getAttribute("rendeles_id");
                String fizetes =
rendelesElement.getElementsByTagName("rendeles_fizetes").item(0).getTextConten
t();
                String ar =
rendelesElement.getElementsByTagName("rendeles_ar").item(0).getTextContent();

                printToFileAndConsole("    <rendeles rendeles_id=\"" +
rendelesId + "\">", System.out, file);
                printElement("datum", datum, file);
                printElement("rendeles_fizetes", fizetes, file);
                printElement("rendeles_ar", ar, file);
                printToFileAndConsole("    </rendeles>", System.out, file);
            }
        }
    }

    // Raktárakat beolvasó metódus
    private static void readRaktarak(Document document, PrintWriter file) {
        NodeList raktarList = document.getElementsByTagName("raktar");
        for (int temp = 0; temp < raktarList.getLength(); temp++) {
            Node node = raktarList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
```

```
                Element raktarElement = (Element) node;
                String raktar_id = raktarElement.getAttribute("raktar_id");
                String rendeles_id =
raktarElement.getAttribute("rendeles_id");
                String ra_darabszam =
raktarElement.getElementsByTagName("ra_darabszam").item(0).getTextContent();
                String raktar_ar =
raktarElement.getElementsByTagName("raktar_ar").item(0).getTextContent();
                String raktar_tipus =
raktarElement.getElementsByTagName("raktar_tipus").item(0).getTextContent();

                printToFileAndConsole("    <raktar raktar_id=\"" + raktar_id +
"\" rendeles_id=\""
                        + rendeles_id + "\">", System.out, file);
                printElement("ra_darabszam", ra_darabszam, file);
                printElement("raktar_ar", raktar_ar, file);
                printElement("raktar_tipus", raktar_tipus, file);
                printToFileAndConsole("    </raktar>", System.out, file);
            }

        }
    }

    // Raktar-Alkalmazott kapcsolatokat beolvasó metódus
    private static void readRaktarAlkalmazott(Document document, PrintWriter
file) {
        NodeList raktarAlkalmazottList =
document.getElementsByTagName("raktar_alkalmazott");
        for (int temp = 0; temp < raktarAlkalmazottList.getLength(); temp++) {
            Node node = raktarAlkalmazottList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element raktarAlkalmazottElement = (Element) node;
                String elerheto = raktarAlkalmazottElement
                        .getElementsByTagName("elerheto").item(0).getTextConte
nt();
                String raktar_id =
raktarAlkalmazottElement.getAttribute("raktar_id");
                String alkalmazott_id =
raktarAlkalmazottElement.getAttribute("alkalmazott_id");

                printToFileAndConsole("    <raktar_alkalmazott raktar_id=\"" +
raktar_id + "\" alkalmazott_id=\""
                        + alkalmazott_id + "\">", System.out, file);
                printToFileAndConsole("          <elerheto>" + elerheto
                        + "</elerheto>", System.out, file);
                printToFileAndConsole("    </raktar_alkalmazott>", System.out,
file);
            }
        }
```

```
    }

    // Alkalmazott beolvasó metódus
    private static void readAlkalmazott(Document document, PrintWriter file) {
        NodeList alkalmazottList =
document.getElementsByTagName("alkalmazott");
        for (int temp = 0; temp < alkalmazottList.getLength(); temp++) {
            Node node = alkalmazottList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element alkalmazottElement = (Element) node;
                String alkalmazott_nev =
alkalmazottElement.getElementsByTagName("alkalmazott_nev").item(0)
                        .getTextContent();
                String alkalmazott_id =
alkalmazottElement.getAttribute("alkalmazott_id");
                String alkalmazott_fizetes =
alkalmazottElement.getElementsByTagName("alkalmazott_fizetes").item(0)
                        .getTextContent();
                String alkalmazott_beosztas =
alkalmazottElement.getElementsByTagName("alkalmazott_beosztas").item(0)
                        .getTextContent();
                String alkalmazott_csatlakozas =
alkalmazottElement.getElementsByTagName("alkalmazott_csatlakozas")
                        .item(0).getTextContent();

                printToFileAndConsole("    <alkalmazott alkalmazott_id=\"" +
alkalmazott_id + "\">", System.out,
                        file);
                printElement("alkalmazott_nev", alkalmazott_nev, file);
                printElement("alkalmazott_fizetes", alkalmazott_fizetes,
file);
                printElement("alkalmazott_beosztas", alkalmazott_beosztas,
file);
                printElement("alkalmazott_csatlakozas",
alkalmazott_csatlakozas, file);
                printToFileAndConsole("    </alkalmazott>", System.out, file);
            }
        }
    }

    // Termékeket beolvasó metódus
    private static void readTermekek(Document document, PrintWriter file) {
        NodeList TermekekList = document.getElementsByTagName("termek");
        for (int temp = 0; temp < TermekekList.getLength(); temp++) {
            Node node = TermekekList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element TermekElement = (Element) node;
                String termek_id = TermekElement.getAttribute("termek_id");
```

```
                String rendeles_id =
TermekElement.getAttribute("rendeles_id");
                String termek_cipo =
TermekElement.getElementsByTagName("termek_cipo").item(0).getTextContent();
                String termek_ruha =
TermekElement.getElementsByTagName("termek_ruha").item(0).getTextContent();
                String termek_kiegeszito =
TermekElement.getElementsByTagName("termek_kiegeszito").item(0)
                        .getTextContent();

                printToFileAndConsole("    <termek termek_id=\"" + termek_id +
"\" rendeles_id=\""
                        + rendeles_id + "\">", System.out, file);
                printElement("termek_cipo", termek_cipo, file);
                printElement("termek_ruha", termek_ruha, file);
                printElement("termek_kiegeszito", termek_kiegeszito, file);
                printToFileAndConsole("    </termek>", System.out, file);

            }
        }
    }

    // Vevőket beolvasó metódus
    private static void readVevok(Document document, PrintWriter file) {
        NodeList vevokList = document.getElementsByTagName("vevo");
        for (int temp = 0; temp < vevokList.getLength(); temp++) {
            Node node = vevokList.item(temp);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                Element vevoElement = (Element) node;
                String vevo_id = vevoElement.getAttribute("id");
                String rendeles_id = vevoElement.getAttribute("rendeles_id");
                String vevo_nev =
vevoElement.getElementsByTagName("nev").item(0).getTextContent();
                String vevo_email =
vevoElement.getElementsByTagName("email").item(0).getTextContent();
                NodeList vevo_telefon =
vevoElement.getElementsByTagName("telefonszam");
                Element vevo_cim = (Element)
vevoElement.getElementsByTagName("cim").item(0);
                String vevo_szemelyi =
vevoElement.getElementsByTagName("szemelyigazolvany_szam").item(0)
                        .getTextContent();
                printToFileAndConsole("    <vevo id=\"" + vevo_id + "\"
rendeles_id=\""
                        + rendeles_id + "\">", System.out, file);
                printElement("nev", vevo_nev, file);
                for (int i = 0; i < vevo_telefon.getLength(); i++) {
                    printElement("telefonszam",
vevo_telefon.item(i).getTextContent(), file);
```

```
                }
                printCim(vevo_cim, file);
                printElement("email", vevo_email, file);
                printElement("szemelyigazolvany_szam", vevo_szemelyi, file);
                printToFileAndConsole("    </vevo>", System.out, file);

            }
        }
    }
    /* Hozzáadó rész kezdete */

    // Rendelések hozzáadása

    private static void addRendeles(Document doc, Element rootElement, String
rendeles_id, String datum,
            String rendeles_fizetes, String rendeles_ar) {
        Element rendeles = doc.createElement("rendeles");
        rendeles.setAttribute("rendeles_id", rendeles_id);

        Element datumElement = createElementAndAddToDoc(doc, "datum", datum);
        Element fizetesElement = createElementAndAddToDoc(doc,
"rendeles_fizetes", rendeles_fizetes);
        Element arElement = createElementAndAddToDoc(doc, "rendeles_ar",
rendeles_ar);

        rendeles.appendChild(datumElement);
        rendeles.appendChild(fizetesElement);
        rendeles.appendChild(arElement);

        rootElement.appendChild(rendeles);
    }

    // Raktár hozzáadása
    private static void addRaktar(Document doc, Element rootElement, String
raktar_id, String rendeles_id,
            String ra_darabszam, String raktar_ar, String raktar_tipus) {
        Element raktar = doc.createElement("raktar");
        raktar.setAttribute("raktar_id", raktar_id);
        raktar.setAttribute("rendeles_id", rendeles_id);

        Element raDarabszamElement = createElementAndAddToDoc(doc,
"ra_darabszam", ra_darabszam);
        Element raktarArElement = createElementAndAddToDoc(doc, "raktar_ar",
raktar_ar);
        Element raktarTipusElement = createElementAndAddToDoc(doc,
"raktar_tipus", raktar_tipus);

        raktar.appendChild(raDarabszamElement);
        raktar.appendChild(raktarArElement);
```

```
        raktar.appendChild(raktarTipusElement);

        rootElement.appendChild(raktar);
    }

    // Raktár-Alkalmazott kapcsolat hozzáadása
    private static void addRaktarAlkalmazott(Document doc, Element
rootElement, String raktar_id,
            String alkalmazott_id, String elerheto) {
        Element raktarAlkalmazott = doc.createElement("raktar_alkalmazott");
        raktarAlkalmazott.setAttribute("raktar_id", raktar_id);
        raktarAlkalmazott.setAttribute("alkalmazott_id", alkalmazott_id);

        Element elerhetoElement = createElementAndAddToDoc(doc, "elerheto",
elerheto);

        raktarAlkalmazott.appendChild(elerhetoElement);

        rootElement.appendChild(raktarAlkalmazott);
    }

    // Alkalmazott hozzáadása
    private static void addAlkalmazott(Document doc, Element rootElement,
String alkalmazott_id, String alkalmazott_nev,
            String alkalmazott_fizetes, String alkalmazott_beosztas, String
alkalmazott_csatlakozas) {
        Element alkalmazott = doc.createElement("alkalmazott");
        alkalmazott.setAttribute("alkalmazott_id", alkalmazott_id);

        Element alkalmazottNevElement = createElementAndAddToDoc(doc,
"alkalmazott_nev", alkalmazott_nev);
        Element alkalmazottFizetesElement = createElementAndAddToDoc(doc,
"alkalmazott_fizetes", alkalmazott_fizetes);
        Element alkalmazottBeosztasElement = createElementAndAddToDoc(doc,
"alkalmazott_beosztas",
                alkalmazott_beosztas);
        Element alkalmazottCsatlakozasElement = createElementAndAddToDoc(doc,
"alkalmazott_csatlakozas",
                alkalmazott_csatlakozas);

        alkalmazott.appendChild(alkalmazottNevElement);
        alkalmazott.appendChild(alkalmazottFizetesElement);
        alkalmazott.appendChild(alkalmazottBeosztasElement);
        alkalmazott.appendChild(alkalmazottCsatlakozasElement);

        rootElement.appendChild(alkalmazott);
    }

    // Termék hozzáadása
```

```
    private static void addTermek(Document doc, Element rootElement, String
termek_id, String rendeles_id,
            String termek_cipo, String termek_ruha, String termek_kiegeszito)
{
        Element termek = doc.createElement("termek");
        termek.setAttribute("termek_id", termek_id);
        termek.setAttribute("rendeles_id", rendeles_id);

        Element termekCipoElement = createElementAndAddToDoc(doc,
"termek_cipo", termek_cipo);
        Element termekRuhaElement = createElementAndAddToDoc(doc,
"termek_ruha", termek_ruha);
        Element termekKiegeszitoElement = createElementAndAddToDoc(doc,
"termek_kiegeszito", termek_kiegeszito);

        termek.appendChild(termekCipoElement);
        termek.appendChild(termekRuhaElement);
        termek.appendChild(termekKiegeszitoElement);

        rootElement.appendChild(termek);
    }

    // Vevő hozzáadása
    private static void addVevo(Document doc, Element rootElement, String
vevo_id, String rendeles_id, String nev,
            String telefon, int iranyitoszam, String telepules, String utca,
String hazszam,
            String email,
            String szemelyi) {
        Element vevo = doc.createElement("vevo");
        vevo.setAttribute("id", vevo_id);
        vevo.setAttribute("rendeles_id", rendeles_id);

        Element nevElement = createElementAndAddToDoc(doc, "nev", nev);
        Element telefonElement = createElementAndAddToDoc(doc, "telefonszam",
telefon);
        Element cimElement = doc.createElement("cim");
        Element iranyitoszamElement = createElementAndAddToDoc(doc,
"iranyitoszam", String.valueOf(iranyitoszam));
        Element telepulesElement = createElementAndAddToDoc(doc, "telepules",
telepules);
        Element utcaElement = createElementAndAddToDoc(doc, "utca", utca);
        Element hazszamElement = createElementAndAddToDoc(doc, "hazszam",
hazszam);
        Element emailElement = createElementAndAddToDoc(doc, "email", email);
        Element szemelyiElement = createElementAndAddToDoc(doc,
"szemelyigazolvany_szam", szemelyi);

        vevo.appendChild(nevElement);
```

```
        vevo.appendChild(telefonElement);
        vevo.appendChild(cimElement);
        cimElement.appendChild(iranyitoszamElement);
        cimElement.appendChild(telepulesElement);
        cimElement.appendChild(utcaElement);
        cimElement.appendChild(hazszamElement);
        vevo.appendChild(emailElement);
        vevo.appendChild(szemelyiElement);

        rootElement.appendChild(vevo);
    }

    // Elem létrehozása és dokumentumhoz adása
    private static Element createElementAndAddToDoc(Document doc, String name,
String value) {
        Element element = doc.createElement(name);
        element.appendChild(doc.createTextNode(value));
        return element;
    }

}
```