

# JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Készítette: Éliás Dániel

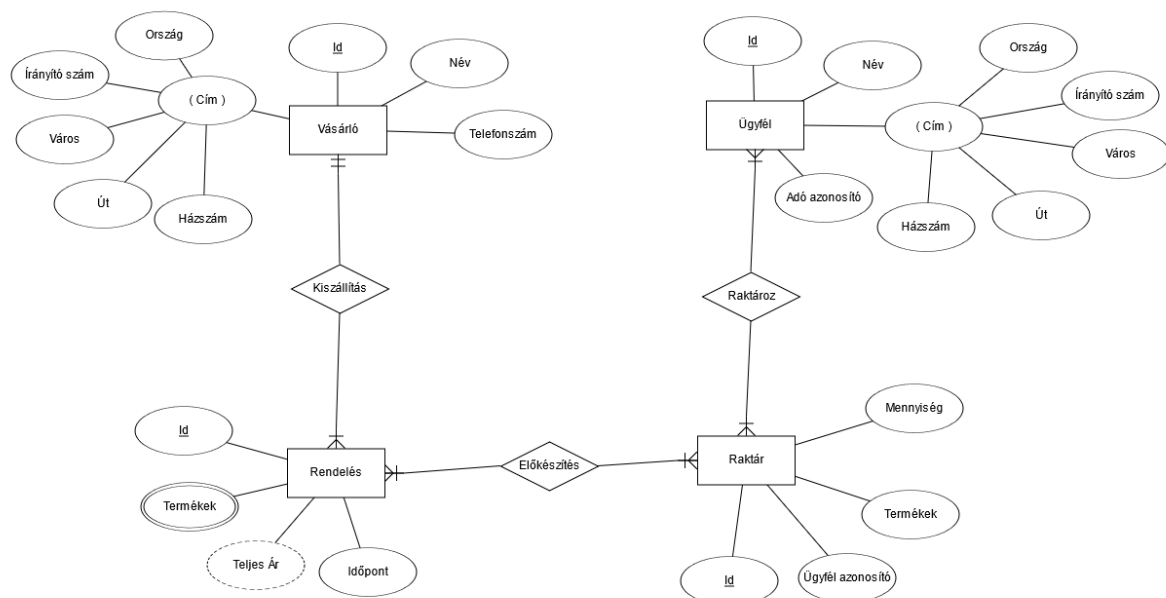
Neptun kód: W8YSQV

## A feladat leírása:

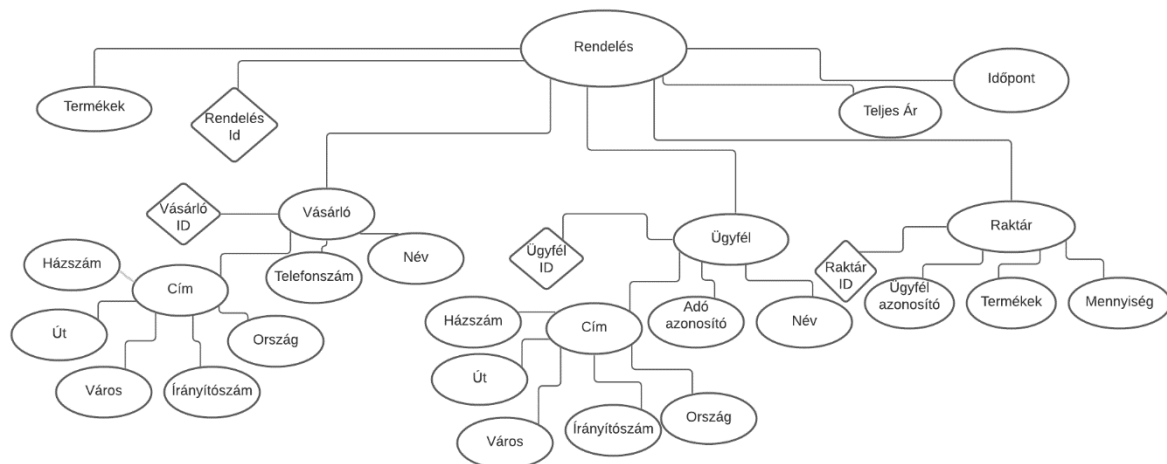
A feladatom egy logisztika cégnek az xml elkészítése. Egy logisztikai cég első számú feladata, hogy a saját ügyfeleinek a csomagjait tárolja és eljuttassa a vásárlóhoz különböző futárcégek segítségével mint például GLS,UPS vagy Sprinter és ez azért jó az ügyfélnek mert neki akkor nem kell foglalkozni a különböző szállítási nehézségekkel mint például hogy le kell fejleszteni egy külön applikációt amivel kommunikálni tud a különböző szállítók api-jaival vagy foglalkozni a szállítóval való leszerződésről. Amikor a cég ügyfele kap egy rendelést egy vásárlótól akkor megkapjuk az ügyféltől a vásárló adatait és azt is, hogy milyen terméket szeretne. Ekkor mi elkezdjük a raktárban a csomag előkészítését és küldünk vissza az ügyfélnek az elkészített xml-t ami tartalmazza a rendelés adatait amiben benne van a termékek neve és a rendelés időpontja és a rendelés teljes összege az ügyfél adatai ami a neve, címe és az adóazonosítója a raktárhoz tartozó adatok a termék neve mennyisége a terméknek és az ügyfél azonosítója ami azért kell mert a raktárban lehet például több fogkefe márka különböző ügyfelektől és még tartalmazza a Vásárló adatait nevét címét és telefonszámát. Amikor elkészült a csomag akkor oda adjuk a futárnak, aki kiszállítja a csomagot. Ez az XML azért jó mert egy visszaigazolást kap az ügyfél arról, hogy még mennyi terméke van a megvásárolt termékből és hogy ténylegesen jól adta meg a vásárló adatait.

## 1. feladat

### 1a) Az adatbázis ER modell:



## 1b) Az adatbázis konvertálása XDM modellre:



## 1c) Az XDM modell alapján XML dokumentum készítése:

```

<?xml version="1.0" encoding="UTF-8"?>
<rendelesek>
  <rendeles id="1">
    <termekek>Fogkefe</termekek>
    <teljesAr>400FT</teljesAr>
    <idopont>2021.10.27</idopont>
    <ugyfel id="1" rid="1">
      <nev>Philips</nev>
      <adoAzonosito>11221100</adoAzonosito>
      <cim>
        <orszag>Kína</orszag>
        <iranyitoszam>21210</iranyitoszam>
        <varos>Peking</varos>
        <ut>Yin út</ut>
        <hazszam>35</hazszam>
      </cim>
    </ugyfel>
    <raktar id="1" rid="1">
      <mennyiseg>12</mennyiseg>
      <termekek>Fogkefe</termekek>
      <ugyfelAzonosito>1</ugyfelAzonosito>
    </raktar>
    <vasarlo id="1" rid="1">
      <nev>Horváth Rozális</nev>
      <telefonszam>06302221234</telefonszam>
      <cim>
        <orszag>Magyarország</orszag>
        <iranyitoszam>6000</iranyitoszam>
        <varos>Kecskemét</varos>
        <ut>Ady Endre</ut>
        <hazszam>15</hazszam>
      </cim>
    </vasarlo>
  </rendeles>
  <rendeles id="2">
    <termekek>Zuhany rózsza</termekek>
    <teljesAr>1200FT</teljesAr>
    <idopont>2021.11.03</idopont>
    <ugyfel id="2" rid="2">
  
```

```

        <nev>Horváth és társa kft</nev>
        <adoAzonosito>01235476</adoAzonosito>
        <cim>
            <ország>Magyarország</ország>
            <iranyitoszam>3849</iranyitoszam>
            <varos>Forró</varos>
            <ut>Fő út</ut>
            <hazszam>111</hazszam>
        </cim>
    </ugyfel>
    <raktar id="23" rid="2">
        <mennyiseg>2</mennyiseg>
        <termek>Zuhany rózsza</termek>
        <ugyfelAzonosito>2</ugyfelAzonosito>
    </raktar>
    <vasarlo id="5" rid="2">
        <nev>Horváth Rozális</nev>
        <telefonszam>06302221234</telefonszam>
        <cim>
            <ország>Magyarország</ország>
            <iranyitoszam>6000</iranyitoszam>
            <varos>Kecskemét</varos>
            <ut>Ady Endre</ut>
            <hazszam>15</hazszam>
        </cim>
    </vasarlo>
</rendeles>
<rendeles id="3">
    <termek>Televízió</termek>
    <teljesAr>35000FT</teljesAr>
    <idopont>2021.11.21</idopont>
    <ugyfel id="1" rid="3">
        <nev>Philips</nev>
        <adoAzonosito>11221100</adoAzonosito>
        <cim>
            <ország>Kína</ország>
            <iranyitoszam>21210</iranyitoszam>
            <varos>Peking</varos>
            <ut>Yang út</ut>
            <hazszam>112</hazszam>
        </cim>
    </ugyfel>
    <raktar id="14" rid="3">
        <mennyiseg>15</mennyiseg>
        <termek>Televízió</termek>
        <ugyfelAzonosito>1</ugyfelAzonosito>
    </raktar>
    <vasarlo id="5" rid="3">
        <nev>Éliás Dániel</nev>
        <telefonszam>06306832678</telefonszam>
        <cim>
            <ország>Magyarország</ország>
            <iranyitoszam>3849</iranyitoszam>
            <varos>Forró</varos>
            <ut>Ady Endre</ut>
            <hazszam>1</hazszam>
        </cim>
    </vasarlo>
</rendeles>

```

</rendelesek>

## Az XML dokumentum alapján XMLSchema készítése (saját típusok):

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

  <xs:element name="rendelesek">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="rendeles" type="rendeles" type="rendeles"
maxOccurs="unbounded" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="rendeles">
    <xs:sequence>
      <xs:element name="termek" type="xs:string" />
      <xs:element name="teljesAr" type="xs:string" />
      <xs:element name="idopont" type="xs:string" />

      <xs:element name="ugyfel" type="ugyfel" type="ugyfel" maxOccurs="1" />

      <xs:element name="raktar" type="raktar" type="raktar" maxOccurs="1" />

      <xs:element name="vasarlo" type="vasarlo" type="vasarlo"
maxOccurs="1" />
    </xs:sequence>
    <xs:attribute name="id" type="xs:int" use="required" />
  </xs:complexType>
  <xs:complexType name="ugyfel">
    <xs:sequence>
      <xs:element name="nev" type="xs:string" />
      <xs:element name="adoAzonosito" type="xs:string" />
      <xs:element name="cim" type="cim" type="cim" />
    </xs:sequence>
    <xs:attribute name="id" type="xs:int" use="required"/>
    <xs:attribute name="rid" type="xs:int" use="required" />
  </xs:complexType>

  <xs:complexType name="raktar">
    <xs:sequence>
      <xs:element name="mennyiseg" type="xs:int" />
      <xs:element name="termek" type="xs:string" />
      <xs:element name="ugyfelAzonosito" type="xs:string" />
    </xs:sequence>
    <xs:attribute name="id" type="xs:int" use="required"/>
    <xs:attribute name="rid" type="xs:int" use="required" />
  </xs:complexType>

  <xs:complexType name="vasarlo">
    <xs:sequence>
      <xs:element name="nev" type="xs:string" />
      <xs:element name="telefonszam" type="xs:string" />
      <xs:element name="cim" type="cim" type="cim" />
    </xs:sequence>
    <xs:attribute name="id" type="xs:int" use="required"/>
    <xs:attribute name="rid" type="xs:int" use="required" />
  </xs:complexType>
</xs:schema>
```

```

</xs:complexType>

<xs:complexType name="cimtype">
  <xs:sequence>
    <xs:element name="ország" type="xs:string" />
    <xs:element name="iranyitoszam" type="xs:int" />
    <xs:element name="varos" type="xs:string" />
    <xs:element name="ut" type="xs:string" />
    <xs:element name="hazszam" type="xs:int" />
  </xs:sequence>
</xs:complexType>
</xs:schema>

```

## 2. feladat

### 2a) adatolvasás

```

public static void main(String[] args) throws SAXException, IOException,
ParserConfigurationException {
    // Létrehozás
    File xmlFile = new File("XMLW8YSQV.xml");
    // Builder + konvertálás
    DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
    DocumentBuilder dBuilder = factory.newDocumentBuilder();
    Document doc = dBuilder.parse(xmlFile);
    // Normalizálás
    doc.getDocumentElement().normalize();
    // Gyökérelem kiírása
    System.out.println("Gyökérelem: " +
doc.getDocumentElement().getNodeName());
    // rendezes elemek nList-be illesztése
    NodeList nList = doc.getElementsByTagName("rendeles");
    System.out.println("-----");
    // Kiírás ciklusa; nList/nNode-on végighaladva az elkészített
algoritmusokkal
    // kiírja a kért adatokat
    for (int i = 0; i < nList.getLength(); i++) {
        Node nNode = nList.item(i);
        if (nNode.getNodeType() == Node.ELEMENT_NODE) {
            Element elem = (Element) nNode;
            String id = elem.getAttribute("id");
            // Adatok átadása a node-okba
            Node node1 =
elem.getElementsByTagName("termekek").item(0);
            String termekek = node1.getTextContent();
            Node node2 =
elem.getElementsByTagName("teljesAr").item(0);
            String teljesAr = node2.getTextContent();
            Node node3 =
elem.getElementsByTagName("idopont").item(0);
            String idopont = node3.getTextContent();
            // Kiírás
            System.out.println("Rendeles ID: " + id);
            System.out.println("Termék: " + termekek);
            System.out.println("Teljes Ár: " + teljesAr);
            System.out.println("Rendelés időpontja: " + idopont);
            // Metódus hívások

```

```

        System.out.println(termekek + " Gyártója:\n");
        listUgyfel(doc, id);
        System.out.println(termekek + " Raktár adatai:\n");
        listRaktar(doc, id);
        System.out.println(termekek + " Vásárló adatai:\n");
        listVasarlo(doc, id);
        System.out.println("\n");
    }
}

// Ügyfél adatainak kiírása az előző módszer alapján
public static void listUgyfel(Document doc, String Rid) {
    NodeList nList = doc.getElementsByTagName("ugyfel");
    for (int i = 0; i < nList.getLength(); i++) {
        Node nNode = nList.item(i);
        if (nNode.getNodeType() == Node.ELEMENT_NODE) {
            Element elem = (Element) nNode;
            if (elem.getAttribute("rid").toString().equals(Rid)) {
                String id = elem.getAttribute("id");
                Node node1 =
elem.getElementsByTagName("nev").item(0);
                String nev = node1.getTextContent();
                Node node2 =
elem.getElementsByTagName("adoAzonosito").item(0);
                String adoAzonosito = node2.getTextContent();
                Node node3 =
elem.getElementsByTagName("cim").item(0);
                String cim = node3.getTextContent();
                System.out.println("Ügyfél id: " + id);
                System.out.println("Ügyfél neve: " + nev);
                System.out.println("Ügyfél adó azonosítója: " +
adoAzonosito);
                System.out.println("Ügyfél adó címe: " + cim +
"\n");
            }
        }
    }
}

// Raktár adatainak kiírása az előző módszer alapján
public static void listRaktar(Document doc, String Rid) {
    NodeList nList = doc.getElementsByTagName("raktar");
    for (int i = 0; i < nList.getLength(); i++) {
        Node nNode = nList.item(i);
        if (nNode.getNodeType() == Node.ELEMENT_NODE) {
            Element elem = (Element) nNode;
            if (elem.getAttribute("rid").toString().equals(Rid)) {
                String id = elem.getAttribute("id");
                Node node1 =
elem.getElementsByTagName("mennyiseg").item(0);
                String mennyiseg = node1.getTextContent();
                Node node2 =
elem.getElementsByTagName("termekek").item(0);
                String termekek = node2.getTextContent();
                Node node3 =
elem.getElementsByTagName("ugyfelAzonosito").item(0);
                String ugyfelAzonosito = node3.getTextContent();
                System.out.println("Raktár id: " + id);
            }
        }
    }
}

```

```

        System.out.println("Termék mennyiség: " +
mennyiseg);

        System.out.println("Termék neve: " + termekek);
        System.out.println("Ügyfél azonosítója: " +
ugyfelAzonosito + "\n");
    }
}

// Vásárló adatainak kiírása az előző módszer alapján
public static void listVasarlo(Document doc, String Rid) {
    NodeList nList = doc.getElementsByTagName("vasarlo");
    for (int i = 0; i < nList.getLength(); i++) {
        Node nNode = nList.item(i);
        if (nNode.getNodeType() == Node.ELEMENT_NODE) {
            Element elem = (Element) nNode;
            if (elem.getAttribute("rid").toString().equals(Rid)) {
                String id = elem.getAttribute("id");
                Node node1 =
elem.getElementsByTagName("nev").item(0);
                String nev = node1.getTextContent();
                Node node2 =
elem.getElementsByTagName("telefonszam").item(0);
                String telefonszam = node2.getTextContent();
                Node node3 =
elem.getElementsByTagName("cim").item(0);
                String cim = node3.getTextContent();
                System.out.println("Vasarló ID: " + id);
                System.out.println("Vásárló neve: " + nev);
                System.out.println("Vásárló telefonszáma: " +
telefonszam);

                System.out.println("Vásárló címe: " + telefonszam
+ "\n");
            }
        }
    }
}

```

## 2b) adatmódosítás

```

package hu.domparse.w8ysqv;

import java.io.File;
import java.io.IOException;
import java.util.Scanner;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerConfigurationException;
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 DomModifyW8YSQV {

    public static void main(String[] args) throws SAXException, IOException,
ParserConfigurationException, TransformerException {
        File xmlFile = new File("XMLW8YSQV.xml");
        File xmlFileMODIFIED = new File("XMLW8YSQVMODIFIED.xml");
        //Scanner nyitása a beolvasáshoz
        Scanner sc = new Scanner(System.in);
        //Builder + konvertálás
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        DocumentBuilder dBuilder = factory.newDocumentBuilder();
        Document doc = dBuilder.parse(xmlFile);
        //Normalizálás
        doc.getDocumentElement().normalize();
        //A rendeles elemek kiválasztása.
        NodeList nList = doc.getElementsByTagName("rendeles");
        //ciklus a változtatásokhoz
        for (int i = 0; i < nList.getLength(); i++) {
            Node nNode = nList.item(i);
            Element elem = (Element) nNode;
            //Termék bekérése
            Node node1 = elem.getElementsByTagName("termekek").item(0);
            String nev = node1.getTextContent();
            //Átadás a node-nak
            System.out.println("A jelenlegi termék neve:" + nev + "\n");
            System.out.println("Add meg az új termék nevét: \n");
            //Bekérés billentyűzetten
            String modifiedname = sc.next();
            //Beállítás node-on keresztül
            node1.setTextContent(modifiedname);
        }
        //Scanner zárása
        sc.close();
        //transformer és domsource használatával változtatjuk a fájlt
        TransformerFactory transformerFactory = TransformerFactory.newInstance();
        Transformer transformer = transformerFactory.newTransformer();
        DOMSource source = new DOMSource(doc);
        //A változtatás a result-ba került
        StreamResult result = new StreamResult(xmlFileMODIFIED);
        //Beírásra kerül a módosított fájlba a módosítás
        transformer.transform(source, result);
    }
}

```

### 3c) adatlekérdezés

```

package hu.domparse.w8ysqv;

import java.io.File;
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.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

public class DomQueryW8YSQV {

    public static void main(String[] args) throws SAXException, IOException,
ParserConfigurationException {
        // Létrehozás
        File xmlFile = new File("XMLW8YSQV.xml");
        // Builder + konvertálás
        DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
        DocumentBuilder dBuilder = factory.newDocumentBuilder();
        Document doc = dBuilder.parse(xmlFile);
        // Normalizálás
        doc.getDocumentElement().normalize();
        // Gyökérelem kiírása
        System.out.println("Gyökérelem: " +
doc.getDocumentElement().getNodeName());
        // rendeles elemek nList-be illesztése
        NodeList nList = doc.getElementsByTagName("rendeles");
        System.out.println("-----");
        // Kiírás ciklusa; nList/nNode-on végighaladva az elkészített
algoritmusokkal
        // kiírja a kért adatokat
        for (int i = 0; i < nList.getLength(); i++) {
            Node nNode = nList.item(i);
            if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                Element elem = (Element) nNode;
                String id = elem.getAttribute("id");
                // Adatok átadása a node-okba
                Node node1 =
elem.getElementsByTagName("termek").item(0);
                String termek = node1.getTextContent();
                Node node2 =
elem.getElementsByTagName("teljesAr").item(0);
                String teljesAr = node2.getTextContent();
                Node node3 =
elem.getElementsByTagName("idopont").item(0);
                String idopont = node3.getTextContent();
                // Az adott query feltétele
                if (termek.equals("Fogkefe")) {
                    // Kiírás
                    System.out.println("Rendelés ID: " + id);
                    System.out.println("Termék: " + termek);
                    System.out.println("Teljes Ár: " + teljesAr);
                    System.out.println("Rendelés időpontja: " + idopont);
                    // Metódus hívások
                    System.out.println(termek + " Gyártója:\n");
                    listUgyfel(doc, id);
                    System.out.println(termek + " Raktár adatai:\n");
                    listRaktar(doc, id);
                    System.out.println(termek + " Vásárló adatai:\n");
                }
            }
        }
    }
}

```

```

        listVasarlo(doc, id);
        System.out.println("\n");
    }
}

// Ügyfél adatainak kiírása az előző módszer alapján
public static void listUgyfel(Document doc, String Rid) {
    NodeList nList = doc.getElementsByTagName("ugyfel");
    for (int i = 0; i < nList.getLength(); i++) {
        Node nNode = nList.item(i);
        if (nNode.getNodeType() == Node.ELEMENT_NODE) {
            Element elem = (Element) nNode;
            if (elem.getAttribute("rid").toString().equals(Rid)) {
                String id = elem.getAttribute("id");
                Node node1 =
elem.getElementsByTagName("nev").item(0);
                String nev = node1.getTextContent();
                Node node2 =
elem.getElementsByTagName("adoAzonosito").item(0);
                String adoAzonosito = node2.getTextContent();
                Node node3 =
elem.getElementsByTagName("cim").item(0);
                String cim = node3.getTextContent();
                System.out.println("Ügyfél id: " + id);
                System.out.println("Ügyfél neve: " + nev);
                System.out.println("Ügyfél adó azonosítója: " +
adoAzonosito);
                System.out.println("Ügyfél adó címe: " + cim +
"\n");
            }
        }
    }
}

// Raktár adatainak kiírása az előző módszer alapján
public static void listRaktar(Document doc, String Rid) {
    NodeList nList = doc.getElementsByTagName("raktar");
    for (int i = 0; i < nList.getLength(); i++) {
        Node nNode = nList.item(i);
        if (nNode.getNodeType() == Node.ELEMENT_NODE) {
            Element elem = (Element) nNode;
            if (elem.getAttribute("rid").toString().equals(Rid)) {
                String id = elem.getAttribute("id");
                Node node1 =
elem.getElementsByTagName("mennyiseg").item(0);
                String mennyiseg = node1.getTextContent();
                Node node2 =
elem.getElementsByTagName("termekek").item(0);
                String termekek = node2.getTextContent();
                Node node3 =
elem.getElementsByTagName("ugyfelAzonosito").item(0);
                String ugyfelAzonosito = node3.getTextContent();
                System.out.println("Raktár id: " + id);
                System.out.println("Termék mennyiség: " +
mennyiseg);
                System.out.println("Termék neve: " + termekek);
            }
        }
    }
}

```

```

        System.out.println("Ügyfél azonosítója: " +
        ugyfelAzonosito + "\n");
    }
}

// Vásárló adatainak kiírása az előző módszer alapján
public static void listVasarlo(Document doc, String Rid) {
    NodeList nList = doc.getElementsByTagName("vasarlo");
    for (int i = 0; i < nList.getLength(); i++) {
        Node nNode = nList.item(i);
        if (nNode.getNodeType() == Node.ELEMENT_NODE) {
            Element elem = (Element) nNode;
            if (elem.getAttribute("rid").toString().equals(Rid)) {
                String id = elem.getAttribute("id");
                Node node1 =
elem.getElementsByTagName("nev").item(0);
                String nev = node1.getTextContent();
                Node node2 =
elem.getElementsByTagName("telefonszam").item(0);
                String telefonszam = node2.getTextContent();
                Node node3 =
elem.getElementsByTagName("cim").item(0);
                String cim = node3.getTextContent();
                System.out.println("Vasarló ID: " + id);
                System.out.println("Vásárló neve: " + nev);
                System.out.println("Vásárló telefonszáma: " +
                telefonszam);
                System.out.println("Vásárló címe: " + cim + "\n");
            }
        }
    }
}

```