

JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat:
Banki rendszer

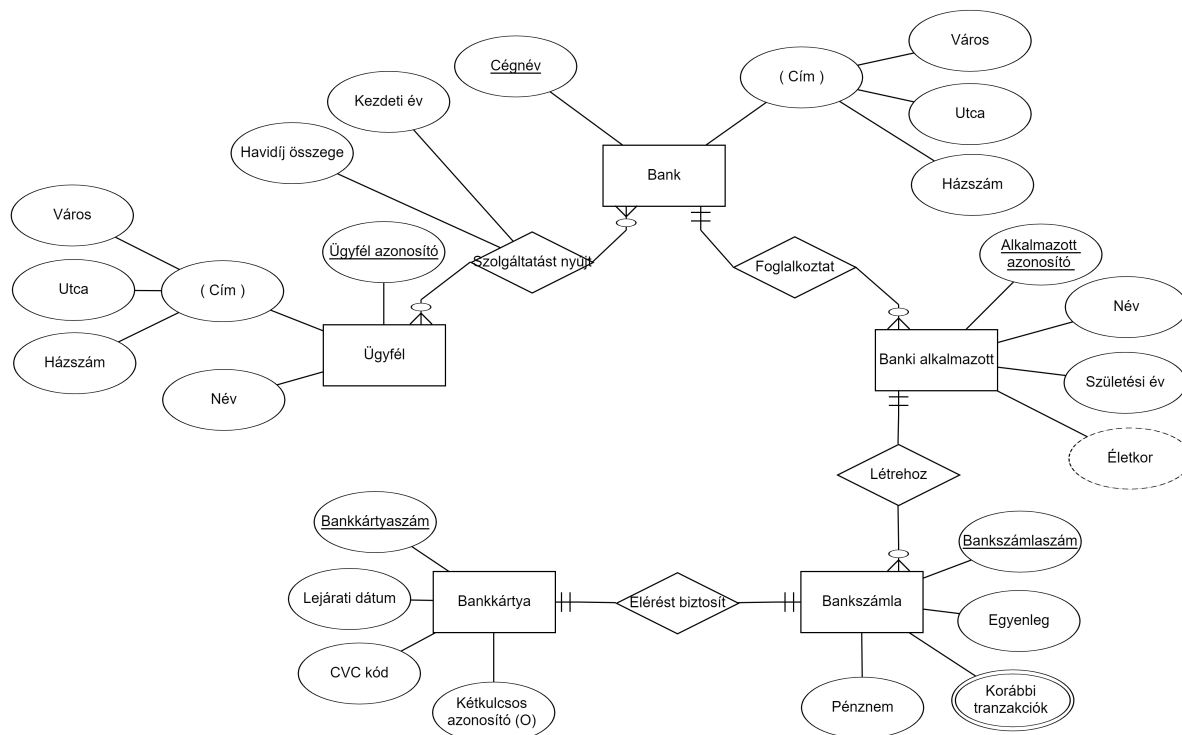
Készítette: Zán Martin Márk
Neptunkód: N16IAF

A feladat leírása: A modellem a bankok működését jellemző adatokat ír le.

- **Bank:** Maga a bank, a legközpontibb elem. Ez a gyökérellem, neki vannak alkalmazottjai, számlái, ügyfelei. Azonosítja őt a neve, és ezentúl van címe amely városból, utcából és házszámból áll.
- **Ügyfél:** Szolgáltatást kap a banktól. Őt egy ügyfélazonosító azonosítja, ezen-túl van neve és ugyanolyan összetett címe, mint a banknak.
- **Banki alkalmazott:** A bankban dolgozó emberek összessége. Egy alkalmazott azonosító azonosítja, van neve, születési éve, és számíthatunk életkort neki.
- **Bankszámla:** A banki alkalmazottak hozhatják létre. Bankszámlaszám azo-nosítja. Egyenlege, pénzneme és korábbi tranzakciói írják le. Korábbi tranz-akciói többértékű, azaz több is lehet belőle.
- **Bankkártya:** Ez biztosítja az elérést a bankszámlához. Bankkártyaszám azo-nosítja. Lejárat dátuma, CVC kódja írja le, és opcionálisan tartozhat hozzá kétkulcsos azonosítás használatául szolgáló azonosító kód.
- **Szolgáltatást nyújt:** A szolgáltatást a bank és az ügyfél között adja meg, melynek van kezdeti éve és havidíjának egy összege.

1. feladat

1a) Az adatbázis ER modell:

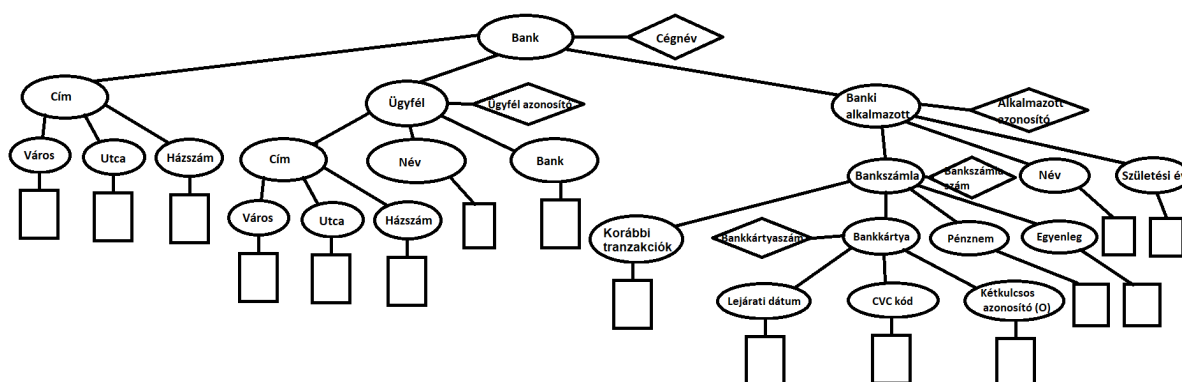


1b) Az adatbázis konvertálása XDM modellre:

Az órai jegyzetben lévő leírást követve konvertáltam ER modellem XDM modellre.

Így az egyedekből elemek, a tulajdonságokból gyerekelemek, a kulcs tulajdonságokból attribútumok és a lényegi egyértékű tulajdonságokból is attribútumok lettek.

A kapcsolatokat a következőképpen valósítottam meg: 1:N kapcsolatból szülő-gyermek kapcsolat, N:M kapcsolatból 2 darab szülő-gyermek kapcsolat.



1c) Az XDM modell alapján XML dokumentum készítése:

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

```
<bank cegnev="OTP" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="XMLSchemaN16IAF.xsd">
```

```
<cim>
```

```
<varos>Miskolc</varos>
```

```
<utca>Uitz Béla</utca>
```

```
<hazszam>6</hazszam>
```

```
</cim>
```

```
<ugyfel ugyfel_azonosito="1">
```

```
<cim>
```

```
<varos>Úzd</varos>
```

```
<utca>József Attila</utca>
```

```
<hazszam>215</hazszam>
```

```
</cim>
```

```
<nev>Kulcsár Ádám</nev>
```

```
<bank>OTP</bank>
```

```
</ugyfel>
```

<ugyfel ugyfel_azonosito="2">
<cim>
<varos>Miskolc</varos>
<utca>Vörösmarty Mihály</utca>
<hazszam>454</hazszam>
</cim>
<nev>Szabó Henrietta</nev>
<bank>OTP</bank>
<bank>ERSTE</bank>
</ugyfel>

<banki_alkalmazott alkalmazott_azonosito="1">
<nev>Visegrádi Iván</nev>
<szuletesi_ev>1988</szuletesi_ev>

<bankszamla bankszamla_szam="1343598542375482">
<egyenleg>540</egyenleg>
<penznem>EUR</penznem>
<korabbi_tranzakciok>-400</korabbi_tranzakciok>
<korabbi_tranzakciok>200</korabbi_tranzakciok>

<bankkartya bankkartya_szam="5012443244328394">
<lejarati_datum>2022-08-30</lejarati_datum>
<cvckod>632</cvckod>
<ketkulcsos_azonosito>7892</ketkulcsos_azonosito>
</bankkartya>

</bankszamla>

</banki_alkalmazott>

<banki_alkalmazott alkalmazott_azonosito="2">
<nev>Galambos Lajos</nev>
<szuletesi_ev>1965</szuletesi_ev>

<bankszamla bankszamla_szam="1343698541375482">
<egyenleg>2200000</egyenleg>
<penznem>KRW</penznem>

<bankkartya bankkartya_szam="1912443244328394">
<lejarati_datum>2024-11-21</lejarati_datum>
<cvckod>112</cvckod>
<ketkulcsos_azonosito>1234</ketkulcsos_azonosito>
</bankkartya>

</bankszamla>

<bankszamla bankszamla_szam="1343498344375482">

<egyenleg>-54</egyenleg>

<penznem>HUF</penznem>

<korabbi_tranzakciok>-54</korabbi_tranzakciok>

<bankkartya bankkartya_szam="1612475254308394">

<lejarati_datum>2019-11-21</lejarati_datum>

<cvckod>748</cvckod>

</bankkartya>

</bankszamla>

</banki_alkalmazott>

</bank>

1d) 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">

  <!-- Gyokerelem -->
  <xs:element name="bank">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="cim" type="cim_tipus"/>
        <xs:element name="ugyfel" type="ugyfel_tipus" maxOccurs="unbounded"/>
        <xs:element name="banki_alkalmazott"
          type="banki_alkalmazott_tipus" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="cegnev" type="xs:string" use="required"/>
    </xs:complexType>

    <!-- Kulcsok -->
    <xs:key name="cegnev">
      <xs:selector xpath="bank"/>
      <xs:field xpath="@cegnev"/>
    </xs:key>

    <xs:key name="ugyfel_azonosito">
      <xs:selector xpath="ugyfel"/>
      <xs:field xpath="@ugyfel_azonosito"/>
    </xs:key>

    <xs:key name="alkalmazott_azonosito">
      <xs:selector xpath="banki_alkalmazott"/>
      <xs:field xpath="@alkalmazott_azonosito"/>
    </xs:key>

    <xs:key name="bankszamla_szam">
      <xs:selector xpath="bankszamla"/>
      <xs:field xpath="@bankszamla_szam"/>
    </xs:key>

    <xs:key name="bankkartya_szam">
      <xs:selector xpath="bankkartya"/>
      <xs:field xpath="@bankkartya_szam"/>
    </xs:key>

  </xs:element>
```

```

<!-- Egyszeru tipusok -->
<xs:simpleType name="penznem_tipus">
<xs:restriction base="xs:string">
<xs:enumeration value="HUF"/>
<xs:enumeration value="EUR"/>
<xs:enumeration value="GBP"/>
<xs:enumeration value="CHF"/>
<xs:enumeration value="USD"/>
<xs:enumeration value="KRW"/>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="cvc_kod_tipus">
<xs:restriction base="xs:int">
<xs:minInclusive value="100"/>
<xs:maxInclusive value="999"/>
</xs:restriction>
</xs:simpleType>

<!-- Komplex tipusok -->
<xs:complexType name="cim_tipus">
<xs:sequence>
<xs:element name="varos" type="xs:string"/>
<xs:element name="utca" type="xs:string"/>
<xs:element name="hazszam" type="xs:unsignedInt"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="ugyfel_tipus">
<xs:sequence>
<xs:element name="cim" type="cim_tipus"/>
<xs:element name="nev" type="xs:string"/>
<xs:element name="bank" type="xs:string" maxOccurs="unbounded"/>
</xs:sequence>
<xs:attribute name="ugyfel_azonosito" type="xs:int" use="required"/>
</xs:complexType>

<xs:complexType name="banki_alkalmazott_tipus">
<xs:sequence>
<xs:element name="nev" type="xs:string"/>
<xs:element name="szuletesi_ev" type="xs:gYear"/>
<xs:element name="bankszamla" type="bankszamla_tipus" maxOccurs="unbounded"/>
</xs:sequence>

```

```

<xs:attribute name="alkalmazott_azonosito" type="xs:int" use="required"/>
</xs:complexType>

<xs:complexType name="bankszamla_tipus">
<xs:sequence>
<xs:element name="egyenleg" type="xs:int"/>
<xs:element name="penznem" type="penznem_tipus"/>
<xs:element name="korabbi_tranzakciok" type="xs:int"
  minOccurs="0" maxOccurs="30"/>
<xs:element name="bankkartya" type="bankkartya_tipus" maxOccurs="unbounded"/>
</xs:sequence>
<xs:attribute name="bankszamla_szam" type="xs:decimal" use="required"/>
  <!-- Itt csak azert szukseges decimal, mert az int-be nem fer bele -->
</xs:complexType>

<xs:complexType name="bankkartya_tipus">
<xs:sequence>
<xs:element name="lejarati_datum" type="xs:date"/>
<xs:element name="cvckod" type="cvc_kod_tipus"/>
<xs:element name="ketkulcsos_azonosito" type="xs:int" minOccurs="0"/>
</xs:sequence>
<xs:attribute name="bankkartya_szam" type="xs:decimal" use="required"/>
</xs:complexType>

</xs:schema>

```


2. feladat

2a) adatolvasás - DOMReadN16IAF.java

Az alábbi kód beolvassa az ügyfelek adatait, majd kiírja azokat a konzolra.

```
package hu.domparse.N16IAF;

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.*;
import org.xml.sax.SAXException;

public class DomReadN16IAF {

    public static void main(String[] args) throws
        ParserConfigurationException, SAXException, IOException {
        // TODO Auto-generated method stub

        // File elérésének biztosítása, gyár létrehozása, DocumentBuilder
        létrehozása és maga a dokumentum (dom objektum) létrehozása.
        File xmlFile = new File("D:\\Programming\\Github\\N16IAF_XMLGyak
        \\XMLTaskN16IAF\\XMLN16IAF.xml");
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        DocumentBuilder dBuilder = factory.newDocumentBuilder();
        Document doc = dBuilder.parse(xmlFile);

        // A dokumentum normalizálása.
        doc.getDocumentElement().normalize();

        // Kiírjuk a gyökérelemet.
        System.out.println("Gyökérelem: " + doc.getDocumentElement().getNodeName());

        // Lekérdezzük az "ugyfel"-eket, először egy listába téve azokat.
        NodeList nList = doc.getElementsByTagName("ugyfel");

        // Majd ezen a listán végigmegyünk.
        for (int i = 0; i < nList.getLength(); i++) {
            Node nNode = nList.item(i);

            // Kiíratjuk a nevüket a node-oknak.
            System.out.println("\nJelenlegi elem: " + nNode.getNodeName());

            // Ha ez egy elem akkor kiírjuk róluk az adatokat.
```

```

if (nNode.getNodeType() == Node.ELEMENT_NODE) {
    Element elem = (Element) nNode;

    // Attribútum (jellemző) lekérdezése.
    String uid = elem.getAttribute("ugyfel_azonosito");

    // Név lekérdezése.
    Node node1 = elem.getElementsByTagName("nev").item(0);
    String name = node1.getTextContent();

    // Itt for ciklus szükséges, mert több bank is tartozhat egy ügyfélhez.
    String bank = "";
    Node node2;
    for ( int j = 0; j < elem.getElementsByTagName("bank").getLength() ; j++ ) {
        node2 = elem.getElementsByTagName("bank").item(j);
        bank = bank + " " + node2.getTextContent();
    }

    // Címüket lekerdezzük
    Node node3 = elem.getElementsByTagName("cim").item(0);
    String adress = node3.getTextContent();

    // Majd a legvégén kiíratjuk azokat.
    System.out.printf("Ügyfél azonosító: %s%n", uid);
    System.out.printf("Név: %s%n", name);
    System.out.printf("Bank: %s%n", bank);
    System.out.printf("Cím: %s%n", adress);
    }
    }
    }
}

```

A futtatás eredménye:

Gyökérelem: bank

Jelenlegi elem: ügyfel

Ügyfél azonosító: 1

Név: Kulcsár Ádám

Bank: OTP

Cím:

Úzd

József Attila

215

Jelenlegi elem: ügyfel

Ügyfél azonosító: 2

Név: Szabó Henrietta

Bank: OTP ERSTE

Cím:

Miskolc

Vörösmarty Mihály

454

2b) adatlekérdezés - DOMQueryN16IAF.java

Az alábbi kód 3 lekérdezést végez:

- Lekérdezi azon ügyfelek nevét melyek "Ózd" városában élnek.
- Lekérdezi azon bankszámlák számait, melyeken tartozás van a bank felé (negatív az egyenlegük)
- Lekérdezi azon banki alkalmazottakat, akik már bocsátottak ki bankkártyát.

```
package hu.domparse.N16IAF;

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.TransformerException;

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 DomQueryNeptunkod {

    public static void main(String[] args) throws
        ParserConfigurationException, SAXException, IOException {
        // TODO Auto-generated method stub

        // File elérésének biztosítása, gyár létrehozása, DocumentBuilder
        // létrehozása és maga a dokumentum (dom objektum) létrehozása.
        File xmlFile = new File("D:\\Programming\\Github\\N16IAF_XMLGyak\\XMLTaskN16IAF\\XMLN16IAF.xml");
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        DocumentBuilder dBuilder = factory.newDocumentBuilder();
        Document doc = dBuilder.parse(xmlFile);

        // A dokumentum normalizálása.
        doc.getDocumentElement().normalize();

        // Kiírjuk a gyökérelemet.
        System.out.println("Gyökérelem: " + doc.getDocumentElement().getNodeName());
```

```

// -----

// Kiíratom azokat az ügyfelek nevét akik Űzdon laknak
System.out.println("Űzdon elő ügyfelek:");

// Lekérdezzük az "ugyfel"-eket, először egy listába téve azokat.
NodeList nList = doc.getElementsByTagName("ugyfel");

// Majd ezen a listán végigmegyünk.
for (int i = 0; i < nList.getLength(); i++) {
Node nNode = nList.item(i);

// Ha ez egy elem akkor kiíjuk róluk az adatokat.
if (nNode.getNodeType() == Node.ELEMENT_NODE) {
Element elem = (Element) nNode;

// Név lekérdezése.
Node node1 = elem.getElementsByTagName("nev").item(0);
String name = node1.getTextContent();

// Városukat lekerdezzük
Node node3 = elem.getElementsByTagName("varos").item(0);
String adress = node3.getTextContent();

if ("Űzd".equals(adress)) {
// Majd a legvégén kiíratjuk azokat.
System.out.printf("Név: %s%n", name);
}

}

System.out.println("\n");

// -----

// Kiíratom azon bankszámlaszámokat, melyeken tartozás van a bank felé.
System.out.println("Negatív egyenlegű bankszámlák:");

nList = doc.getElementsByTagName("bankszamlaszo");

for (int i = 0; i < nList.getLength(); i++) {
Node nNode = nList.item(i);

```

```

// Ha ez egy elem akkor kiíjuk róluk az adatokat.
if (nNode.getNodeType() == Node.ELEMENT_NODE) {
    Element elem = (Element) nNode;

    // Bankszámlaszám lekérdezés
    String uid = elem.getAttribute("bankszamla_szam");

    // Egyenlegeket lekérdezzük.
    Node node3 = elem.getElementsByTagName("egyenleg").item(0);
    String money = node3.getTextContent();

    if (Integer.valueOf(money) < 0) {
        // Majd a legvégén kiíratjuk azokat.
        System.out.printf("Bankszámlaszám: %s%n", uid);
    }

}

System.out.println("\n");

// -----

// Kiíratom azon banki alkalmazottakat, akik már adtak ki bankkártyát.
System.out.println("Szorgalmas dolgozók (akik már adtak ki bankkártyát):");

nList = doc.getElementsByTagName("banki_alkalmazott");

for (int i = 0; i < nList.getLength(); i++) {
    Node nNode = nList.item(i);

    // Ha ez egy elem akkor kiíjuk róluk az adatokat.
    if (nNode.getNodeType() == Node.ELEMENT_NODE) {
        Element elem = (Element) nNode;

        // Név lekérdezése.
        Node node1 = elem.getElementsByTagName("nev").item(0);
        String name = node1.getTextContent();

        // Bankszámlaszámokat lekérdezzük.
        Node node3 = elem.getElementsByTagName("bankszamla").item(0);

        if (node3 != null) {
            // Majd a legvégén kiíratjuk a neveket.
            System.out.printf("Név: %s%n", name);
        }
    }
}

```

}

}

}

}

}

A futtatás eredménye:

Gyökérelem: bank

Üzdon elő ügyfelek:

Név: Kulcsár Ádám

Negatív egyenlegű bankszámlák:

Bankszámlaszám: 1343498344375482

Szorgalmas dolgozók (akik már adtak ki bankkártyát):

Név: Visegrádi Iván

Név: Galambos Lajos

2c) adatmódosítás - DOMModifyN16IAF.java

Az alábbi kód 3 módosítást végez:

- Minden bankszámlára 5% kamatot ír jóvá/számol fel.
- Egy megadott bankszámla valutáját módosítja EUR-ról GBP-re és módosítja a rajta lévő összeget is körülbelül az EUR/GBP árfolyamnak megfelelően.
- Kitörli a bank címét.

```
package hu.domparse.N16IAF;

import java.io.File;
import java.io.IOException;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
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.NamedNodeMap;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

public class DomModifyN16IAF {

    public static void main(String[] args) throws ParserConfigurationException,
        SAXException, IOException, TransformerException {
        // TODO Auto-generated method stub

        // File elérésének biztosítása, gyár létrehozása, DocumentBuilder
        létrehozása és maga a dokumentum (dom objektum) létrehozása.
        File xmlFile = new File("D:\\Programming\\Github\\N16IAF_XMLGyak
        \\XMLTaskN16IAF\\XMLN16IAF.xml");
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        DocumentBuilder dBuilder = factory.newDocumentBuilder();
        Document doc = dBuilder.parse(xmlFile);

        // A dokumentum normalizálása.
        doc.getDocumentElement().normalize();
```

```

// Kiírjuk a gyökérelemet.
System.out.println("Gyökérelem: " + doc.getDocumentElement().getNodeName());

// -----

// Minden bankszámlára 5% kamatot írunk jóvá/számolunk fel
System.out.println("5% kamat jóváírás:");

NodeList nList = doc.getElementsByTagName("bankszaml");

for (int i = 0; i < nList.getLength(); i++) {
    Node nNode = nList.item(i);

    // Ha ez egy elem akkor kiírjuk róluk az adatokat.
    if (nNode.getNodeType() == Node.ELEMENT_NODE) {
        Element elem = (Element) nNode;

        // Egyenlegeket lekérdezzük.
        Node node3 = elem.getElementsByTagName("egyenleg").item(0);
        String money = node3.getTextContent();

        // Majd a legvégén kiíratjuk azokat.
        System.out.printf("Egyenleg jóváírás előtt: %s%n", money);

        node3.setTextContent(String.valueOf(Math.round(Double.valueOf(money) * 1.05)));

        money = node3.getTextContent();
        System.out.printf("Egyenleg jóváírás után: %s%n", money);

    }
}

System.out.println("\n");

// -----

// A 1343598542375482 bankszámlaszámú bankszámla valutáját GBP-re
// módosítjuk (és a pénzének összegét is átváltjuk abba).
System.out.println("Valuta módosítása.");

nList = doc.getElementsByTagName("bankszaml");

for (int i = 0; i < nList.getLength(); i++) {
    Node nNode = nList.item(i);

```

```

// Ha ez egy elem akkor kiíjuk róluk az adatokat.
if (nNode.getNodeType() == Node.ELEMENT_NODE) {
    Element elem = (Element) nNode;

    // Bankszámlaszám lekérdezés
    String uid = elem.getAttribute("bankszamla_szam");

    if (uid.equals("1343598542375482")) {

        Node node2 = elem.getElementsByTagName("penznem").item(0);
        node2.setTextContent("GBP");
        Node node3 = elem.getElementsByTagName("egyenleg").item(0);
        node3.setTextContent(String.valueOf
            (Math.round( Double.valueOf(node3.getTextContent()) * 0.85) ));
        // 1 EUR ~ 0.85 GBP
    }

}

System.out.println("\n");

// -----

        // Bank címenek torlese
        System.out.println("Bank címenek torlese.");

        Node bank = doc.getFirstChild();
        NodeList child = bank.getChildNodes();

        for (int i = 0; i < child.getLength(); i++) {
            Node nNode = child.item(i);

            // Ha ez egy elem akkor kiíjuk róluk az adatokat.
            if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                Element elem = (Element) nNode;

                if("cim".equals(elem.getNodeName())) {
                    bank.removeChild(elem);
                }

            }
        }

```

```
System.out.println("\n");
```

```
// -----
```

```
// Módosított XML kiíratása
```

```
TransformerFactory transformerFactory = TransformerFactory.newInstance();
```

```
    Transformer transformer = transformerFactory.newTransformer();
```

```
    DOMSource source = new DOMSource(doc);
```

```
    System.out.println("Módosított XML");
```

```
    StreamResult consoleResult = new StreamResult(System.out);
```

```
    transformer.transform(source, consoleResult);
```

```
}
```

```
}
```

A futtatás eredménye:

Gyökérelem: bank
5% kamat jóváírás:
Egyenleg jóváírás előtt: 540
Egyenleg jóváírás után: 567
Egyenleg jóváírás előtt: 2200000
Egyenleg jóváírás után: 2310000
Egyenleg jóváírás előtt: -54
Egyenleg jóváírás után: -57

Valuta módosítása.

Bank címenek torlese.

Modositott XML

```
<?xml version="1.0" encoding="utf-8" standalone="no"?><bank xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

```
<ugyfel ugyfel_azonosito="1">
  <cim>
    <varos>Ázsd</varos>
    <utca>Jázsef Attila</utca>
    <hazszam>215</hazszam>
  </cim>
  <nev>Kulcsr Adm</nev>
  <bank>OTP</bank>
</ugyfel>
```

```
<ugyfel ugyfel_azonosito="2">
  <cim>
    <varos>Miskolc</varos>
    <utca>Vársmarty Mihály</utca>
    <hazszam>454</hazszam>
  </cim>
  <nev>Szabai Henrietta</nev>
  <bank>OTP</bank>
  <bank>ERSTE</bank>
</ugyfel>
```

```
<banki_alkalmazott alkalmazott_azonosito="1">
  <nev>Visegrádi Iván</nev>
  <szuletesi_ev>1988</szuletesi_ev>
```

<bankszamla bankszamla_szam="1343598542375482">
<egyenleg>482</egyenleg>
<penznem>GBP</penznem>
<korabbi_tranzakciok>-400</korabbi_tranzakciok>
<korabbi_tranzakciok>200</korabbi_tranzakciok>

<bankkartya bankkartya_szam="5012443244328394">
<lejarati_datum>2022-08-30</lejarati_datum>
<cvckod>632</cvckod>
<ketkulcsos_azonosito>7892</ketkulcsos_azonosito>
</bankkartya>

</bankszamla>

</banki_alkalmazott>

<banki_alkalmazott alkalmazott_azonosito="2">
<nev>Galambos Lajos</nev>
<szuletesi_ev>1965</szuletesi_ev>

<bankszamla bankszamla_szam="1343698541375482">
<egyenleg>2310000</egyenleg>
<penznem>KRW</penznem>

<bankkartya bankkartya_szam="1912443244328394">
<lejarati_datum>2024-11-21</lejarati_datum>
<cvckod>112</cvckod>
<ketkulcsos_azonosito>1234</ketkulcsos_azonosito>
</bankkartya>

</bankszamla>

<bankszamla bankszamla_szam="1343498344375482">
<egyenleg>-57</egyenleg>
<penznem>HUF</penznem>
<korabbi_tranzakciok>-54</korabbi_tranzakciok>

<bankkartya bankkartya_szam="1612475254308394">
<lejarati_datum>2019-11-21</lejarati_datum>
<cvckod>748</cvckod>
</bankkartya>

</bankszamla>

</banki_alkalmazott>
</bank>