JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Gyártócég nyilvántartás

Készítette: **Szöllősi János**

Neptunkód: **BC6X4X**

Dátum: **2023.11.27.**

Tartalomjegyzék

[A feladat leírása: 3](#_Toc152158987)

[Az egyedek és a köztük lévő relációk: 3](#_Toc152158988)

[1. feladat 4](#_Toc152158989)

[1a) Az adatbázis ER modellje: 4](#_Toc152158990)

[1b) Az adatbázis konvertálása XDM modellre: 4](#_Toc152158991)

[1c) Az XDM modell alapján XML dokumentum készítése: 5](#_Toc152158992)

[1d) Az XML dokumentum alapján XMLSchema készítése (saját típusok, ref, key, keyref, speciális elemek): 11](#_Toc152158993)

[2. feladat 19](#_Toc152158994)

[2a) Adatolvasás: 19](#_Toc152158995)

[2b) Adatlekérdezés: 22](#_Toc152158996)

[2c) Adatmódosítás: 27](#_Toc152158997)

# A feladat leírása:

A féléves feladatomban bérgyártó cégek adatbázisát készítem el. A beadandóm a cégek adatait tartja számon, mint például a cég helye, termékei, dolgozói, gyártási információk.

A feladatom ötletét a munkahelyem (Jabil Circuit Magyarország Kft.) adta leegyszerűsítve.

## Az egyedek és a köztük lévő relációk:

A cég és település egyedek közötti reláció N:M, kötelező típusú kapcsolat, mivel egy cég több településen is elhelyezkedhet, illetve egy településhez töb cég is tartozhat.

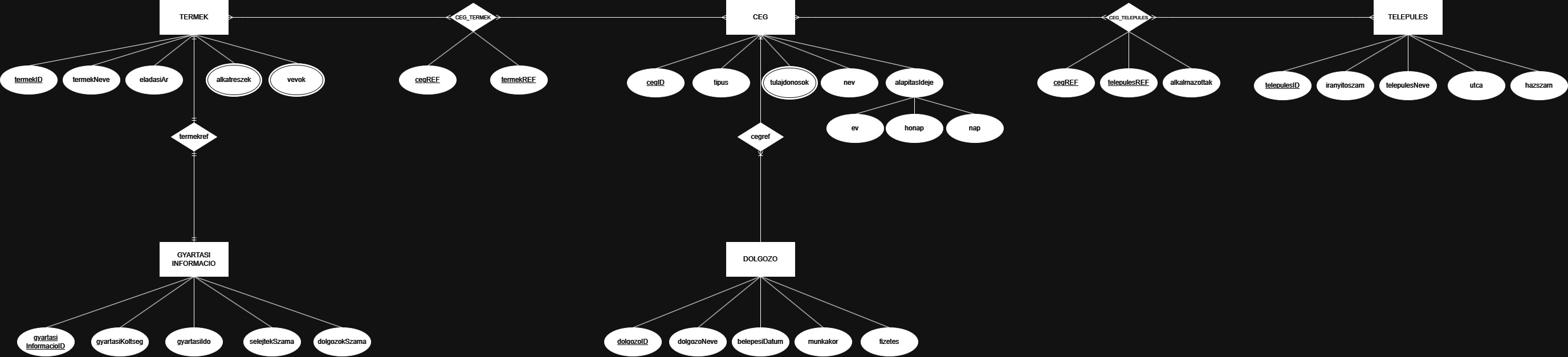
A cég és termék egyedek közötti reláció N:M, kötelező típusú kapcsolat, mivel egy cég több terméket is gyárthat, illetve egy terméket több cég is gyárthatja.

A cég és dolgozó egyedek közötti reláció 1:N, kötelező típusú kapcsolat, mivel egy céghez több dolgozó is tartozhat, de egy dolgozó egyszerre csak egy cégnél dolgozhat.

A termék és gyártási Információ egyedek közötti reláció 1:1, kötelező típusú kapcsolat, mivel egy termékhez csak egy gyártási információ tartozhat.

# 1. feladat

## 1a) Az adatbázis ER modellje:



## 1b) Az adatbázis konvertálása XDM modellre:

A black and white image of a group of white objects

Description automatically generated

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

<gyartocegek>

<!-- Cégek -->

<ceg cegID="1">

<nev>ABC Zrt</nev>

<tipus>Zrt</tipus>

<alapitasIdeje>

<ev>2010</ev>

<honap>02</honap>

<nap>15</nap>

</alapitasIdeje>

<tulajdonosok>

<tulajdonos>Kovács István</tulajdonos>

<tulajdonos>Nagy Mária</tulajdonos>

<tulajdonos>Szabó János</tulajdonos>

</tulajdonosok>

</ceg>

<ceg cegID="2">

<nev>DEF Kft</nev>

<tipus>Kft</tipus>

<alapitasIdeje>

<ev>2015</ev>

<honap>06</honap>

<nap>10</nap>

</alapitasIdeje>

<tulajdonosok>

<tulajdonos>Kiss Andrea</tulajdonos>

<tulajdonos>Nagy Gábor</tulajdonos>

<tulajdonos>Kovács Anikó</tulajdonos>

</tulajdonosok>

</ceg>

<ceg cegID="3">

<nev>GHI Bt</nev>

<tipus>Bt</tipus>

<alapitasIdeje>

<ev>2018</ev>

<honap>11</honap>

<nap>25</nap>

</alapitasIdeje>

<tulajdonosok>

<tulajdonos>Nagy Péter</tulajdonos>

<tulajdonos>Kiss Judit</tulajdonos>

<tulajdonos>Kovács Bence</tulajdonos>

</tulajdonosok>

</ceg>

<!-- Települések -->

<telepules telepulesID="1">

<iranyitoszam>1098</iranyitoszam>

<telepulesNeve>Pécs</telepulesNeve>

<utca>Király utca</utca>

<hazszam>25</hazszam>

</telepules>

<telepules telepulesID="2">

<iranyitoszam>1024</iranyitoszam>

<telepulesNeve>Budapest</telepulesNeve>

<utca>Alkotás utca</utca>

<hazszam>8</hazszam>

</telepules>

<telepules telepulesID="3">

<iranyitoszam>3300</iranyitoszam>

<telepulesNeve>Eger</telepulesNeve>

<utca>Széchenyi utca</utca>

<hazszam>12</hazszam>

</telepules>

<!-- Cég-Település kapcsolatok -->

<ceg\_telepules cegREF="1" telepulesREF="1" >

<alkalmazottak>5000</alkalmazottak>

</ceg\_telepules>

<ceg\_telepules cegREF="2" telepulesREF="2" >

<alkalmazottak>4000</alkalmazottak>

</ceg\_telepules>

<ceg\_telepules cegREF="3" telepulesREF="3" >

<alkalmazottak>3000</alkalmazottak>

</ceg\_telepules>

<!-- Termékek -->

<termek termekID="1">

<termekNeve>Laptop XYZ</termekNeve>

<eladasiAr>1500</eladasiAr>

<alkatreszek>

<alkatresz>Memória</alkatresz>

<alkatresz>Processzor</alkatresz>

<alkatresz>Tárhely</alkatresz>

</alkatreszek>

<vevok>

<vevo>Andrásné</vevo>

<vevo>Kiss Géza</vevo>

<vevo>Nagy Balázs</vevo>

</vevok>

</termek>

<termek termekID="2">

<termekNeve>Smartphone Plus</termekNeve>

<eladasiAr>800</eladasiAr>

<alkatreszek>

<alkatresz>Kijelző</alkatresz>

<alkatresz>Akku</alkatresz>

<alkatresz>Kamera</alkatresz>

</alkatreszek>

<vevok>

<vevo>Nagy Katalin</vevo>

<vevo>Kiss József</vevo>

<vevo>Szabó Anna</vevo>

</vevok>

</termek>

<termek termekID="3">

<termekNeve>Asztali Számítógép Pro</termekNeve>

<eladasiAr>2500</eladasiAr>

<alkatreszek>

<alkatresz>Processzor</alkatresz>

<alkatresz>RAM</alkatresz>

<alkatresz>GPU</alkatresz>

</alkatreszek>

<vevok>

<vevo>Kiss Péter</vevo>

<vevo>Nagy Zoltán</vevo>

<vevo>Szabó Eszter</vevo>

</vevok>

</termek>

<!-- Cég-Termék kapcsolatok -->

<ceg\_termek cegREF="1" termekREF="1" />

<ceg\_termek cegREF="2" termekREF="2" />

<ceg\_termek cegREF="3" termekREF="3" />

<!-- Gyártási Információk -->

<gyartasiInformacio gyartasiInformacioID="1" termekREF="1">

<gyartasiKoltseg>800</gyartasiKoltseg>

<gyartasiIdo>24</gyartasiIdo>

<selejtekSzama>3%</selejtekSzama>

<dolgozokSzama>25</dolgozokSzama>

</gyartasiInformacio>

<gyartasiInformacio gyartasiInformacioID="2" termekREF="2">

<gyartasiKoltseg>400</gyartasiKoltseg>

<gyartasiIdo>25</gyartasiIdo>

<selejtekSzama>2%</selejtekSzama>

<dolgozokSzama>15</dolgozokSzama>

</gyartasiInformacio>

<gyartasiInformacio gyartasiInformacioID="3" termekREF="3">

<gyartasiKoltseg>1200</gyartasiKoltseg>

<gyartasiIdo>26</gyartasiIdo>

<selejtekSzama>1%</selejtekSzama>

<dolgozokSzama>30</dolgozokSzama>

</gyartasiInformacio>

<!-- Dolgozók -->

<dolgozo dolgozoID="1" cegREF="1">

<dolgozoNeve>Kis Anikó</dolgozoNeve>

<belepesiDatum>2021-08-05</belepesiDatum>

<munkakor>Rendszergazda</munkakor>

<fizetes>3500</fizetes>

</dolgozo>

<dolgozo dolgozoID="2" cegREF="2">

<dolgozoNeve>Nagy Gábor</dolgozoNeve>

<belepesiDatum>2022-01-15</belepesiDatum>

<munkakor>Programozó</munkakor>

<fizetes>4200</fizetes>

</dolgozo>

<dolgozo dolgozoID="3" cegREF="3">

<dolgozoNeve>Szabó Mónika</dolgozoNeve>

<belepesiDatum>2023-05-20</belepesiDatum>

<munkakor>HR Menedzser</munkakor>

<fizetes>3800</fizetes>

</dolgozo>

</gyartocegek>

## 1d) Az XML dokumentum alapján XMLSchema készítése (saját típusok, ref, key, keyref, speciális elemek):

<?xml version="1.0" encoding="UTF-8"?>

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

<!-- Elemek, tulajdonságok -->

<xs:element name="nev" type="xs:string" />

<xs:element name="vevo" type="xs:string" />

<xs:element name="alkatresz" type="xs:string" />

<xs:element name="tipus" type="xs:string" />

<xs:element name="telepulesNeve" type="xs:string" />

<xs:element name="utca" type="xs:string" />

<xs:element name="hazszam" type="xs:integer" />

<xs:element name="termekNeve" type="xs:string" />

<xs:element name="eladasiAr" type="xs:integer" />

<xs:element name="gyartasiIdo" type="xs:integer" />

<xs:element name="gyartasiKoltseg" type="xs:integer" />

<xs:element name="dolgozokSzama" type="xs:integer" />

<xs:element name="dolgozoNeve" type="xs:string" />

<xs:element name="munkakor" type="xs:string" />

<xs:element name="fizetes" type="xs:integer" />

<xs:element name="alkalmazottak" type="xs:integer" />

<xs:attribute name="cegID" type="xs:integer" />

<xs:attribute name="cegREF" type="xs:integer" />

<xs:attribute name="telepulesID" type="xs:integer" />

<xs:attribute name="telepulesREF" type="xs:integer" />

<xs:attribute name="termekID" type="xs:integer" />

<xs:attribute name="termekREF" type="xs:integer" />

<xs:attribute name="gyartasiInformacioID" type="xs:integer" />

<xs:attribute name="dolgozoID" type="xs:integer" />

<!-- Egyszerű típusok -->

<xs:simpleType name="ev\_type">

<xs:restriction base="xs:string">

<xs:pattern value="(19|20)\d+\d+"></xs:pattern>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="honap\_type">

<xs:restriction base="xs:string">

<xs:pattern value="(0[1-9]|1[012])"></xs:pattern>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="nap\_type">

<xs:restriction base="xs:string">

<xs:pattern value="(0[1-9]|[12][0-9]|3[01])"></xs:pattern>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="datum\_type">

<xs:restriction base="xs:string">

<xs:pattern value="(19|20)\d\d-(0[1-9]|1[012])-(0[1-9]|[12][0-9]|3[01])"></xs:pattern>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="iranyitoszam\_type">

<xs:restriction base="xs:string">

<xs:length value="4" />

<xs:pattern value="([0-9])\*" />

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="nev\_type">

<xs:restriction base="xs:string">

<xs:pattern value="(\D\*\s+\D+)(\D\*\s\*)" />

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="selejtekSzama\_type">

<xs:restriction base="xs:string">

<xs:pattern value="(\d\*%)"></xs:pattern>

</xs:restriction>

</xs:simpleType>

<!-- Komplex típusok -->

<xs:complexType name="alapitasIdeje\_type">

<xs:sequence>

<xs:element name="ev" type="ev\_type"/>

<xs:element name="honap" type="honap\_type"/>

<xs:element name="nap" type="nap\_type"/>

</xs:sequence>

</xs:complexType>

<xs:complexType name="tulajdonosok\_type">

<xs:sequence>

<xs:element name="tulajdonos" type="nev\_type" maxOccurs="unbounded"/>

</xs:sequence>

</xs:complexType>

<xs:complexType name="alkatreszek\_type">

<xs:sequence>

<xs:element ref="alkatresz" maxOccurs="unbounded"/>

</xs:sequence>

</xs:complexType>

<xs:complexType name="vevok\_type">

<xs:sequence>

<xs:element ref="vevo" maxOccurs="unbounded"/>

</xs:sequence>

</xs:complexType>

<xs:complexType name="ceg\_type">

<xs:sequence>

<xs:element name="nev" type="nev\_type"/>

<xs:element ref="tipus"/>

<xs:element name="alapitasIdeje" type="alapitasIdeje\_type"/>

<xs:element name="tulajdonosok" type="tulajdonosok\_type"/>

</xs:sequence>

<xs:attribute ref="cegID" use="required"/>

</xs:complexType>

<xs:complexType name="telepules\_type">

<xs:sequence>

<xs:element name="iranyitoszam" type="iranyitoszam\_type"/>

<xs:element ref="telepulesNeve"/>

<xs:element ref="utca"/>

<xs:element ref="hazszam"/>

</xs:sequence>

<xs:attribute ref="telepulesID" use="required"/>

</xs:complexType>

<xs:complexType name="termek\_type">

<xs:sequence>

<xs:element ref="termekNeve"/>

<xs:element ref="eladasiAr"/>

<xs:element name="alkatreszek" type="alkatreszek\_type"/>

<xs:element name="vevok" type="vevok\_type"/>

</xs:sequence>

<xs:attribute ref="termekID" use="required"/>

</xs:complexType>

<xs:complexType name="gyartasiInformacio\_type">

<xs:sequence>

<xs:element ref="gyartasiKoltseg"/>

<xs:element ref="gyartasiIdo"/>

<xs:element name="selejtekSzama" type="selejtekSzama\_type"/>

<xs:element ref="dolgozokSzama"/>

</xs:sequence>

<xs:attribute ref="gyartasiInformacioID" use="required"/>

<xs:attribute ref="termekREF" use="required"/>

</xs:complexType>

<xs:complexType name="dolgozo\_type">

<xs:sequence>

<xs:element name="dolgozoNeve" type="nev\_type"/>

<xs:element name="belepesiDatum" type="datum\_type"/>

<xs:element ref="munkakor"/>

<xs:element ref="fizetes"/>

</xs:sequence>

<xs:attribute ref="dolgozoID" use="required"/>

<xs:attribute ref="cegREF" use="required"/>

</xs:complexType>

<!-- Kapcsolótáblák -->

<xs:complexType name="ceg\_termek\_type">

<xs:attribute ref="cegREF" use="required"/>

<xs:attribute ref="termekREF" use="required"/>

</xs:complexType>

<xs:complexType name="ceg\_telepules\_type">

<xs:sequence>

<xs:element ref="alkalmazottak"/>

</xs:sequence>

<xs:attribute ref="cegREF" use="required"/>

<xs:attribute ref="telepulesREF" use="required"/>

</xs:complexType>

<!-- Gyartócégek -->

<xs:element name="gyartocegek">

<xs:complexType>

<xs:sequence>

<xs:element name="ceg" type="ceg\_type" maxOccurs="unbounded"/>

<xs:element name="telepules" type="telepules\_type" maxOccurs="unbounded"/>

<xs:element name="ceg\_telepules" type="ceg\_telepules\_type" maxOccurs="unbounded"/>

<xs:element name="termek" type="termek\_type" maxOccurs="unbounded"/>

<xs:element name="ceg\_termek" type="ceg\_termek\_type" maxOccurs="unbounded"/>

<xs:element name="gyartasiInformacio" type="gyartasiInformacio\_type" maxOccurs="unbounded"/>

<xs:element name="dolgozo" type="dolgozo\_type" maxOccurs="unbounded"/>

</xs:sequence>

</xs:complexType>

<!-- Kulcsok -->

<xs:unique name="cegID">

<xs:selector xpath="ceg"/>

<xs:field xpath="@cegID"/>

</xs:unique>

<xs:unique name="telepulesID">

<xs:selector xpath="telepules"/>

<xs:field xpath="@telepulesID"/>

</xs:unique>

<xs:unique name="termekID">

<xs:selector xpath="termek"/>

<xs:field xpath="@termekID"/>

</xs:unique>

<xs:unique name="gyartasiInformacioID">

<xs:selector xpath="gyartasiInformacio"/>

<xs:field xpath="@gyartasiInformacioID"/>

</xs:unique>

<xs:unique name="dolgozoID">

<xs:selector xpath="dolgozo"/>

<xs:field xpath="@dolgozoID"/>

</xs:unique>

<!-- Kulcshivatkozások (idegen kulcsok) -->

<xs:keyref name="ceg\_FK1" refer="cegID">

<xs:selector xpath="dolgozo"></xs:selector>

<xs:field xpath="@cegREF"></xs:field>

</xs:keyref>

<xs:keyref name="termek\_FK1" refer="termekID">

<xs:selector xpath="gyartasiInformacio"></xs:selector>

<xs:field xpath="@termekREF"></xs:field>

</xs:keyref>

</xs:element>

</xs:schema>

# 2. feladat

## 2a) Adatolvasás:

package hu.domparse.BC6X4X;

import org.w3c.dom.Document;

import org.w3c.dom.Node;

import org.w3c.dom.NodeList;

import org.w3c.dom.Text;

import org.xml.sax.SAXException;

import javax.xml.parsers.DocumentBuilder;

import javax.xml.parsers.DocumentBuilderFactory;

import javax.xml.parsers.ParserConfigurationException;

import java.io.File;

import java.io.IOException;

public class DomReadBC6X4X {

public static void main(String[] args) {

File xmlFile = new File("XMLBC6X4X.xml");

Document doc = introduceFile(xmlFile);

if (doc == null) {

System.out.println("The document is null");

System.exit(-1);

} else {

doc.getDocumentElement().normalize();

System.out.println("Root: " + doc.getDocumentElement().getNodeName());

}

NodeList nodeList = doc.getDocumentElement().getChildNodes();

String indent = "";

listData(nodeList, indent);

}

public static Document introduceFile(File xmlFile){

Document doc = null;

try{

DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

DocumentBuilder dbBuilder = dbFactory.newDocumentBuilder();

doc = dbBuilder.parse(xmlFile);

} catch (ParserConfigurationException | SAXException | IOException e) {

e.printStackTrace();

}

return doc;

}

public static void listData(NodeList nodeList, String indent){

indent += "\t";

if(nodeList != null) {

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

Node node = nodeList.item(i);

if (node.getNodeType() == Node.ELEMENT\_NODE && !node.getTextContent().trim().isEmpty()) {

System.out.println(indent + "{" + node.getNodeName() + "}:");

NodeList nodeList\_new = node.getChildNodes();

listData(nodeList\_new, indent);

} else if (node instanceof Text){

String value = node.getNodeValue().trim();

if (value.isEmpty()){

continue;

}

System.out.println(indent + node.getTextContent());

}

}

}

}

}

## 2b) Adatlekérdezés:

package hu.domparse.BC6X4X;

import org.w3c.dom.Document;

import org.w3c.dom.Node;

import org.w3c.dom.NodeList;

import org.w3c.dom.Text;

import org.xml.sax.SAXException;

import javax.xml.parsers.DocumentBuilder;

import javax.xml.parsers.DocumentBuilderFactory;

import javax.xml.parsers.ParserConfigurationException;

import java.io.File;

import java.io.IOException;

public class DomQueryBC6X4X {

public static void main(String[] args) {

File xmlFile = new File("XMLBC6X4X.xml");

Document doc = introduceFile(xmlFile);

if (doc == null) {

System.out.println("The document is null");

System.exit(-1);

} else {

doc.getDocumentElement().normalize();

System.out.println("Root: " + doc.getDocumentElement().getNodeName());

}

//Kiírja azokat a cégeket, amelyeket 2017 előtt alapítottak

NodeList alapitas = doc.getDocumentElement().getElementsByTagName("ceg");

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

NodeList query = alapitas.item(i).getChildNodes();

for (int j = 0; j < query.getLength(); j++) {

if (query.item(j).getNodeName().equals("alapitasIdeje")){

NodeList query2 = query.item(j).getChildNodes();

for (int k = 0; k < query2.getLength(); k++) {

if (query2.item(k).getNodeName().equals("ev")&& Integer.parseInt(query2.item(k).getTextContent()) <2017) {

listData(alapitas.item(i).getChildNodes(), "");

}

}

}

}

}

System.out.println("-------------------------------------------------------------");

//Termékek kiírsa, ahol az eladási ár több, mint 1000$

NodeList termek = doc.getDocumentElement().getElementsByTagName("termek");

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

NodeList query = termek.item(i).getChildNodes();

for (int j = 0; j < query.getLength(); j++) {

if (query.item(j).getNodeName().equals("eladasiAr") && Integer.parseInt(query.item(j).getTextContent()) > 1000){

listData(termek.item(i).getChildNodes(), "");

}

}

}

System.out.println("-------------------------------------------------------------");

//Programozó dolgozók adatainak kiírása

NodeList programozok = doc.getDocumentElement().getElementsByTagName("dolgozo");

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

NodeList query = programozok.item(i).getChildNodes();

for (int j = 0; j < query.getLength(); j++) {

if (query.item(j).getNodeName().equals("munkakor") && query.item(j).getTextContent().equals("Programozó")){

listData(programozok.item(i).getChildNodes(), "");

}

}

}

System.out.println("-------------------------------------------------------------");

//Kiírja azoknak a termékeknek az adatait, amik pontosan 3 alkatrésszel rendelkeznek

NodeList termekek = doc.getDocumentElement().getElementsByTagName("termek");

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

NodeList query = termekek.item(i).getChildNodes();

for (int j = 0; j < query.getLength(); j++) {

if (query.item(j).getNodeName().equals("alkatreszek")){

NodeList query2 = query.item(j).getChildNodes();

if((query2.getLength()-1)/2==3) {

listData(termekek.item(i).getChildNodes(), "");

}

}

}

}

System.out.println("-------------------------------------------------------------");

//Kiírja azokat a dolgozókat, akiknek a fizetőse osztható 3-al

NodeList dolgozok = doc.getDocumentElement().getElementsByTagName("dolgozo");

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

NodeList query = dolgozok.item(i).getChildNodes();

for (int j = 0; j < query.getLength(); j++) {

if (query.item(j).getNodeName().equals("fizetes") && Integer.parseInt(query.item(j).getTextContent())%3==0){

listData(dolgozok.item(i).getChildNodes(), "");

}

}

}

}

public static Document introduceFile(File xmlFile){

Document doc = null;

try {

DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

DocumentBuilder dbBuilder = dbFactory.newDocumentBuilder();

doc = dbBuilder.parse(xmlFile);

} catch (ParserConfigurationException | SAXException | IOException e) {

e.printStackTrace();

}

return doc;

}

public static void listData(NodeList nodeList, String indent){

indent += "\t";

if (nodeList != null) {

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

Node node = nodeList.item(i);

if (node.getNodeType() == Node.ELEMENT\_NODE && !node.getTextContent().trim().isEmpty()) {

System.out.println(indent + "{" + node.getNodeName() + "}:");

NodeList nodeList\_new = node.getChildNodes();

listData(nodeList\_new, indent);

} else if (node instanceof Text){

String value = node.getNodeValue().trim();

if (value.isEmpty()){

continue;

}

System.out.println(indent + node.getTextContent());

}

}

}

}

}

## 2c) Adatmódosítás:

package hu.domparse.BC6X4X;

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 DomModifyBC6X4X {

public static void main(String[] args) {

File xmlFile = new File("XMLBC6X4X.xml");

Document doc = introduceFile(xmlFile);

if (doc == null) {

System.out.println("The document is null");

System.exit(-1);

} else {

doc.getDocumentElement().normalize();

System.out.println("Root: " + doc.getDocumentElement().getNodeName());

}

//Cég alapítás évének csökkentése 10 évvel

NodeList evek = doc.getDocumentElement().getElementsByTagName("ev");

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

evek.item(i).setTextContent(Integer.toString(Integer.parseInt(evek.item(i).getTextContent())-10));

}

evek = doc.getDocumentElement().getElementsByTagName("ceg");

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

listData(evek.item(i).getChildNodes(), "");

}

System.out.println("-------------------------------------------------------------");

//Dolgozók fizetésének növelése a duplájára

NodeList dolgozok = doc.getDocumentElement().getElementsByTagName("fizetes");

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

dolgozok.item(i).setTextContent(Integer.toString(Integer.parseInt(dolgozok.item(i).getTextContent())\*2));

}

dolgozok = doc.getDocumentElement().getElementsByTagName("dolgozo");

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

listData(dolgozok.item(i).getChildNodes(), "");

}

System.out.println("-------------------------------------------------------------");

//Termékek eladási árának növelése 200-al

NodeList termekek = doc.getDocumentElement().getElementsByTagName("eladasiAr");

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

termekek.item(i).setTextContent(Integer.toString(Integer.parseInt(termekek.item(i).getTextContent())+200));

}

termekek = doc.getDocumentElement().getElementsByTagName("termek");

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

listData(termekek.item(i).getChildNodes(), "");

}

System.out.println("-------------------------------------------------------------");

//Dolgozók munkakörének módosítása összeszerelőre

NodeList munkakorok = doc.getDocumentElement().getElementsByTagName("munkakor");

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

munkakorok.item(i).setTextContent("Összeszerelő");

}

munkakorok = doc.getDocumentElement().getElementsByTagName("dolgozo");

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

listData(munkakorok.item(i).getChildNodes(), "");

}

System.out.println("-------------------------------------------------------------");

//Minden cég tulajdonosához hozzáadom a saját nevemet

NodeList tulajdonosok = doc.getDocumentElement().getElementsByTagName("tulajdonosok");

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

Node newNode = tulajdonosok.item(i).appendChild(doc.createElement("tulajdonos"));

newNode.setTextContent("Szöllősi János");

}

tulajdonosok = doc.getDocumentElement().getElementsByTagName("ceg");

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

listData(tulajdonosok.item(i).getChildNodes(), "");

}

}

public static Document introduceFile(File xmlFile){

Document doc = null;

try{

DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

DocumentBuilder dbBuilder = dbFactory.newDocumentBuilder();

doc = dbBuilder.parse(xmlFile);

} catch (ParserConfigurationException | SAXException | IOException e) {

e.printStackTrace();

}

return doc;

}

public static void listData(NodeList nodeList, String indent){

indent += "\t";

if(nodeList != null) {

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

Node node = nodeList.item(i);

if (node.getNodeType() == Node.ELEMENT\_NODE && !node.getTextContent().trim().isEmpty()) {

System.out.println(indent + "{" + node.getNodeName() + "}:");

NodeList nodeList\_new = node.getChildNodes();

listData(nodeList\_new, indent);

} else if (node instanceof Text){

String value = node.getNodeValue().trim();

if (value.isEmpty()){

continue;

}

System.out.println(indent + node.getTextContent());

}

}

}

}

}