JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Gyártócég nyilvántartás

Készítette: **Szöllősi János**
Neptunkód: **BC6X4X**
Dátum: **2023.11.27.**

Tartalomjegyzék

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 tulajdonságai:

1. Ceg:
    a. cegID: A ceg egyed elsődleges kulcsa.
    b. tipus: A ceg tipusa, pl: KFT, ZRT.
    c. nev: A ceg neve.
    d. tulajdonosok: A tulajdonosok listája (többértékű tulajdonság)
    e. alapitasIdeje: A ceg alapításának ideje (összetett tulajdonság):
        i. ev: A ceg alapítás idejének éve.
        ii. honap: A ceg alapítás idejének hónapja
        iii. nap: A ceg alapítás idejének napja.
2. Telepules:
    a. telepulesID: A telepules egyed elsődlehes kulcsa.
    b. iranyitoszam: A telepules irányítószáma.
    c. telepulesNeve: A telepules megnevezése.
    d. utca: Utca neve ahol a cég található.
    e. házszám: Házszam ahol a cég található.
3. Ceg_telepules:
    a. cegREF: A cegID-re mutató idegen kulcs.
    b. telepulesREF: A telepulesID-re mutató idegen kulcs. Ez kapcsolja össze a cégeket a településekkel.
    c. alkalmazottak: Ezen a településen található cég általán alkalmazott dolgozók száma.
4. Dolgozo:
    a. dolgozoID: A dolgozo egyed elsődleges kulcsa.
    b. dolgozoNeve: A dolgozó neve.
    c. belepesiDatum: Dolgozó belépésének a dátuma (pl: 2023-10-10)
    d. munkakor: A dolgozó munkakörének megnevezése (pl: Programozó)
    e. fizetes: A dolgozó havi fizetése.
5. Termek:
    a. termekID: A termek egyed elsődleges kulcsa.
    b. termekNeve: A termék megnevezése.
    c. eladasiAr: A termék értéke.
    d. alkatreszek: A termékbe szerelt alkatrészek listája (többértékű tulajdonság)
    e. vevok: A terméket megvásárló cégek/magánszemélyek listája (többértékű tulajdonság)
6. Ceg_termek:
    a. cegREF: A cegID-re mutató idegen kulcs.
    b. termekREF: A termekID-re mutató idegen kulcs. Ez kapcsolja össze a cégeket a termékekkel.
7. GyartasiInformacio:
    a. gyartasiInformacioID: A GyartasiInformacio egyed elsődleges kulcsa.

b. gyartasiKoltseg: Az adott termék legyártására szükséges pénz (pl 50 $)
c. gyartasiIdo: Az adott termék legyártására szükséges idő (pl 20 sec)
d. selejtekSzama: A termék gyártása alatt előfordulő selejtek száma százalékban megadva
e. dolgozokSzama: A termék gyártásához szükséges emberi dolgozók száma.

1. feladat

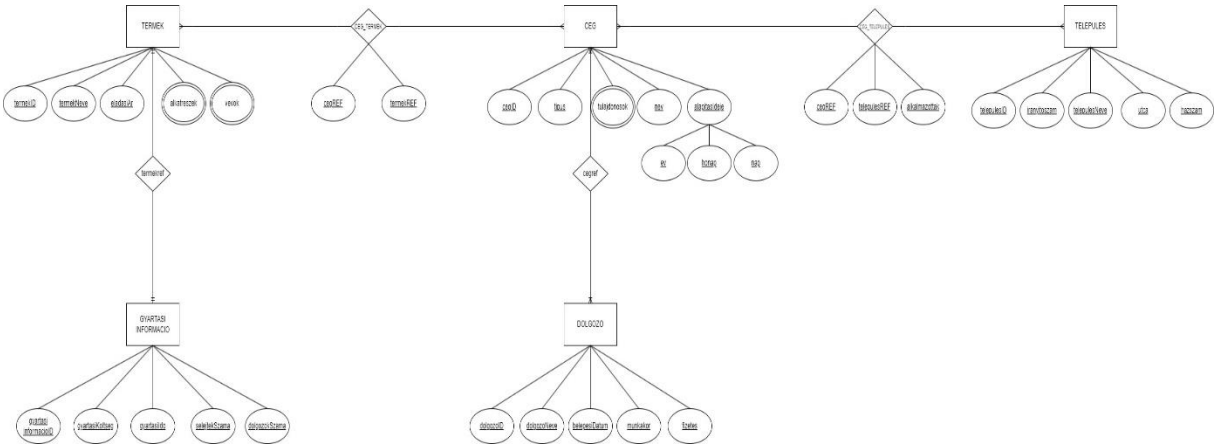
1a) Az adatbázis ER modellje:

- 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.

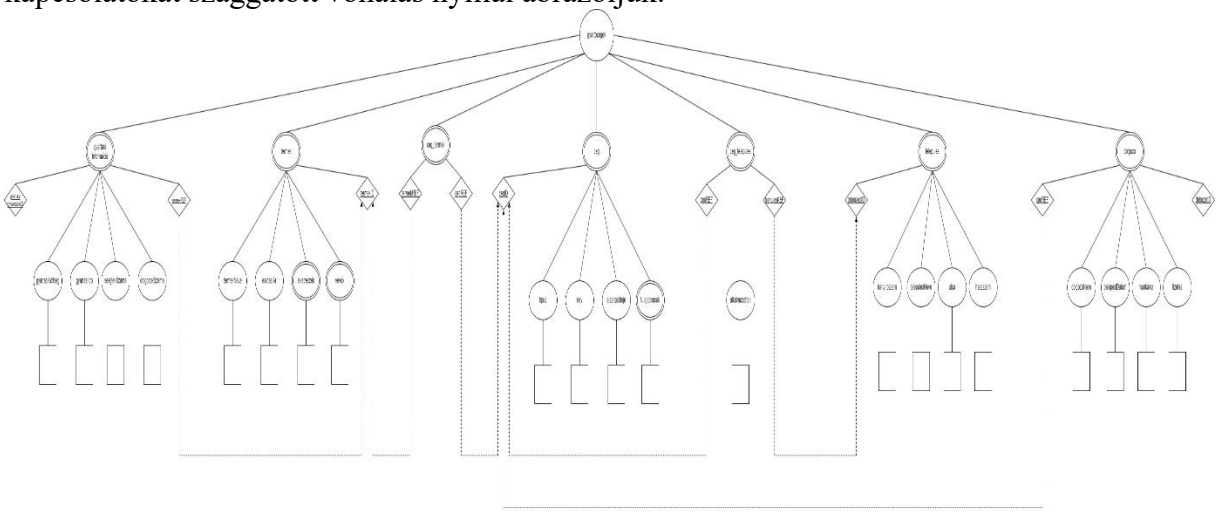
- A cég és dolgozó egyedek közötti reláció 1:N, kötelező típusú kapcsolat, mivel egy céghez több dolgozó is tartozhat, de egy dolgozó egyszerre csak egy cégnél dolgozhat.

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



1b) Az adatbázis konvertálása XDM modellre:

Az XDM modell használata során három különböző jelölést alkalmazunk: ellipszist, rombuszt és téglalapot. Az ellipszis reprezentálja az elemeket, minden egyedi entitásból létrejön egy elem, beleértve a tulajdonságokat is. A rombusz az attribútumokat ábrázolja, melyek a kulcsfontosságú tulajdonságokból erednek. A téglalapok pedig a szöveget reprezentálják. Dupla ellipszissel jelöljük a többértékű elemeket. Az idegenkulcsok és a kulcsok közötti kapcsolatokat szaggatott vonalas nyíllal ábrázoljuk.



1c) Az XDM modell alapján XML dokumentum készítése:

Az XML dokumentum az XDM modell alapján került kidolgozásra. A gyökérelem a „gyartocegek” megnevezést kapta a feladat leírása alapján. A gyökérelem után hoztám létre a gyermekelemeket.

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>

<gyartocegek xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="XMLSchemaBC6X4X.xsd">
   <!-- 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):

A XML dokumentumhoz egy validációt elősegítő séma elkészítése volt szükséges. Első lépésként az egyszerű típusokat gyűjtöttem össze, majd meghatároztam a saját típusokat. Ezt követően kialakítottam a komplex típusokat minden egyes elemre, és létrehoztam elsődleges és idegen kulcsokat. Végül megvalósítottam az egyedek közötti kapcsolatokat (pl: 1:1, 1:N, N:M). A pontos validálás érdekében rengeteg helyen patternt, illetve regex-et használtam.

```
<?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="evTipus">
		<xs:restriction base="xs:string">
			<xs:pattern value="(19|20)\d+\d+"></xs:pattern>
		</xs:restriction>
	</xs:simpleType>
```

```
	<xs:simpleType name="honapTipus">
		<xs:restriction base="xs:string">
			<xs:pattern value="(0[1-9]|1[012])"></xs:pattern>
		</xs:restriction>
	</xs:simpleType>

	<xs:simpleType name="napTipus">
		<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="datumTipus">
		<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="iranyitoszamTipus">
		<xs:restriction base="xs:string">
			<xs:length value="4" />
			<xs:pattern value="([0-9])*" />
		</xs:restriction>
	</xs:simpleType>

	<xs:simpleType name="nevTipus">
		<xs:restriction base="xs:string">
			<xs:pattern value="(\D*\s+\D+)(\D*\s*)" />
		</xs:restriction>
	</xs:simpleType>

	<xs:simpleType name="selejtekSzamaTipus">
		<xs:restriction base="xs:string">
			<xs:pattern value="(\d*%)"></xs:pattern>
		</xs:restriction>
	</xs:simpleType>

	<!-- Komplex típusok -->




	<xs:complexType name="alapitasIdejeTipus">
		<xs:sequence>
			<xs:element name="ev" type="evTipus"/>
			<xs:element name="honap" type="honapTipus"/>
			<xs:element name="nap" type="napTipus"/>
		</xs:sequence>
	</xs:complexType>
```

```
	<xs:complexType name="tulajdonosokTipus">
		<xs:sequence>
			<xs:element name="tulajdonos" type="nevTipus" maxOccurs="unbounded"/>
		</xs:sequence>
	</xs:complexType>

	<xs:complexType name="alkatreszekTipus">
		<xs:sequence>
			<xs:element ref="alkatresz"  maxOccurs="unbounded"/>
		</xs:sequence>
	</xs:complexType>

	<xs:complexType name="vevokTipus">
		<xs:sequence>
			<xs:element ref="vevo"  maxOccurs="unbounded"/>
		</xs:sequence>
	</xs:complexType>


	<xs:complexType name="cegTipus">
		<xs:sequence>
			<xs:element name="nev" type="nevTipus"/>
			<xs:element ref="tipus"/>
			<xs:element name="alapitasIdeje" type="alapitasIdejeTipus"/>
			<xs:element name="tulajdonosok" type="tulajdonosokTipus"/>
		</xs:sequence>
		<xs:attribute ref="cegID" use="required"/>
	</xs:complexType>

	<xs:complexType name="telepulesTipus">
		<xs:sequence>
			<xs:element name="iranyitoszam" type="iranyitoszamTipus"/>
			<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="termekTipus">
		<xs:sequence>
			<xs:element ref="termekNeve"/>
			<xs:element ref="eladasiAr"/>
			<xs:element name="alkatreszek" type="alkatreszekTipus"/>
			<xs:element name="vevok" type="vevokTipus"/>
		</xs:sequence>
		<xs:attribute ref="termekID" use="required"/>
```

```
	</xs:complexType>

	<xs:complexType name="gyartasiInformacioTipus">
		<xs:sequence>
			<xs:element ref="gyartasiKoltseg"/>
			<xs:element ref="gyartasiIdo"/>
			<xs:element name="selejtekSzama" type="selejtekSzamaTipus"/>
			<xs:element ref="dolgozokSzama"/>
		</xs:sequence>
		<xs:attribute ref="gyartasiInformacioID" use="required"/>
		<xs:attribute ref="termekREF" use="required"/>
	</xs:complexType>

	<xs:complexType name="dolgozoTipus">
		<xs:sequence>
			<xs:element name="dolgozoNeve" type="nevTipus"/>
			<xs:element name="belepesiDatum" type="datumTipus"/>
			<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_termekTipus">
		<xs:attribute ref="cegREF" use="required"/>
		<xs:attribute ref="termekREF" use="required"/>
	</xs:complexType>

	<xs:complexType name="ceg_telepulesTipus">
		<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="cegTipus"
maxOccurs="unbounded"/>
				<xs:element name="telepules" type="telepulesTipus"
maxOccurs="unbounded"/>
				<xs:element name="ceg_telepules" type="ceg_telepulesTipus"
maxOccurs="unbounded"/>
```

```
                        <xs:element name="termek" type="termekTipus" maxOccurs="unbounded"/>
                        <xs:element name="ceg_termek" type="ceg_termekTipus" maxOccurs="unbounded"/>
                        <xs:element name="gyartasiInformacio" type="gyartasiInformacioTipus" maxOccurs="unbounded"/>
                        <xs:element name="dolgozo" type="dolgozoTipus" 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

A DOM programokat JAVA nyelven készítettem el, ahogyan azt a feladatkiírás is előírta. Az alábbi programok részletes bemutatását a következő részletekben fogom elvégezni.

2a) Adatolvasás:

- A main metódus meghívja az introduceFile metódust, amely megnyitja az „XMLBC6X4X.xml” filet és beolvassa annak adaitait. Ha sikeresen beolvasta az adatokat, akkor azokat megjeleníti/kiírja a console-ra a listData metódus segítségével. A kiíratás után a beolvasott adatokat kiírja egy új xml file-ba „XMLBC6X4Xreadoutput.xml” néven.

```
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 javax.xml.transform.OutputKeys;

import javax.xml.transform.Transformer;

import javax.xml.transform.TransformerFactory;

import javax.xml.transform.dom.DOMSource;

import javax.xml.transform.stream.StreamResult;

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("<?xml version=\"1.0\" encoding=\"utf-8\"?>");
               System.out.println("<" + doc.getDocumentElement().getNodeName()+"
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:noNamespaceSchemaLocation=\"XMLSchemaBC6X4X.xsd\">");
            }


            NodeList nodeList = doc.getDocumentElement().getChildNodes();
            String indent = "";
            listData(nodeList, indent);
            System.out.println("</" + doc.getDocumentElement().getNodeName()+">");
            try {
               TransformerFactory transformerFactory = TransformerFactory.newInstance();
               Transformer transformer = transformerFactory.newTransformer();
               transformer.setOutputProperty(OutputKeys.INDENT, "no");
```

```
            transformer.transform(new DOMSource(doc), new StreamResult("XMLBC6X4Xreadoutput.xml"));

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

    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.print(indent + "<" + node.getNodeName());

                if (node.hasAttributes()) {

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

                        Node attribute = node.getAttributes().item(k);

                        System.out.print("
"+attribute.getNodeName()+"=\""+attribute.getNodeValue()+"\"");

                    }

                    System.out.println(">");

                }else {

                    System.out.println(">");

                }


                NodeList nodeList_new = node.getChildNodes();

                listData(nodeList_new, indent);

                System.out.println(indent + "</" + node.getNodeName() + ">");

            } else if (node instanceof Text){

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

                if (value.isEmpty()){

                    continue;

                }

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

            }
```

```
            }
        }
    }
}
```

2b) Adatlekérdezés:

- A main metódus meghívja az introduceFile metódust, amely megnyitja az „XMLBC6X4X.xml” filet és beolvassa annak adaitait. Ha sikeresen beolvasta az adatokat, akkor azokat megjeleníti/kiírja a console-ra a listData metódus segítségével azokat az adatokat, amelyek megegyeznek az általunk megadott szűrőfeltételeknek. Például az első lekérdezésnél megkapjuk azoknak a cégeknek az adatait, amelyeket 2017 előtt alapították.

```
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 javax.xml.transform.OutputKeys;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;

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();
      }

      //KiĂ-rja azokat a cĂ©geket, amelyeket 2017 elĹ‘tt alapĂ-tottak
       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("------------------------------------------------------------");


    //TermÃ©kek kiÃ-rsa, ahol az eladÃˇsi Ăˇr tÃ¶bb, 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("------------------------------------------------------------");


    //ProgramozÃł dolgozÃłk adatainak kiÃ-rÃˇsa
    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("ProgramozÃł")){
                listData(programozok.item(i).getChildNodes(), "");
            }
        }
    }

    System.out.println("------------------------------------------------------------");
    //KiÃ-rja azoknak a termÃ©keknek az adatait, amik pontosan 3 alkatrÃ©sszel
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("----------------------------------------------------------");
    //KiĂ-rja azokat a dolgozĂłkat, akiknek a fizetĹ‘se oszthatĂł 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.print(indent + "<" + node.getNodeName());
                if (node.hasAttributes()) {
                    for (int k = 0; k < node.getAttributes().getLength(); k++) {
                        Node attribute = node.getAttributes().item(k);
                        System.out.print("
"+attribute.getNodeName()+"=\""+attribute.getNodeValue()+"\"");
                    }
```

```
                System.out.println(">");
              }else {
                System.out.println(">");
              }

              NodeList nodeList_new = node.getChildNodes();
              listData(nodeList_new, indent);
              System.out.println(indent + "</" + node.getNodeName() + ">");
            } else if (node instanceof Text){
              String value = node.getNodeValue().trim();
              if (value.isEmpty()){
                continue;
              }
              System.out.println(indent + node.getTextContent());
            }
          }
        }
      }

}
```

2c) Adatmódosítás:

- A main metódus meghívja az introduceFile metódust, amely megnyitja az „XMLBC6X4X.xml” filet és beolvassa annak adaitait. Ha sikeresen beolvasta az adatokat, akkor azokat megjeleníti/kiírja a console-ra a listData metódus segítségével azokat az adatokat, amelyeket sikeresen módosítottunk a feladatban. Például az első módosításnál kiíratjuk az összes cégnek az adatait, de előtte minden cég alapításának évét csökkentjük 10 évvel.

```
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 javax.xml.transform.OutputKeys;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;

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();
         }

      //CĂ©g alapĂ-tĂ˙s Ă©vĂ©nek csĂ¶kkentĂ©se 10 Ă©vvel
      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("------------------------------------------------------------");

        //DolgozÄłk fizetÄ©sÄ©nek nÄ¶velÄ©se a duplÄˇjÄˇra
        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).getTextC
ontent())*2));
        }

        dolgozok = doc.getDocumentElement().getElementsByTagName("dolgozo");
        for (int i = 0; i < dolgozok.getLength(); i++) {
            listData(dolgozok.item(i).getChildNodes(), "");
        }

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

        //TermÄ©kek eladÄˇsi ÄˇrÄˇnak nÄ¶velÄ©se 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).getTextC
ontent())+200));
        }

        termekek = doc.getDocumentElement().getElementsByTagName("termek");
        for (int i = 0; i < termekek.getLength(); i++) {
            listData(termekek.item(i).getChildNodes(), "");
        }

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

        //DolgozÄłk munkakÄ¶rÄ©nek mÄłdosÄ-tÄˇsa Ä¶sszeszerelĹ‘re
        NodeList                                    munkakorok                                    =
doc.getDocumentElement().getElementsByTagName("munkakor");
        for (int i = 0; i < munkakorok.getLength(); i++) {
            munkakorok.item(i).setTextContent("Ä–sszeszerelĹ‘");
        }

        munkakorok = doc.getDocumentElement().getElementsByTagName("dolgozo");
        for (int i = 0; i < munkakorok.getLength(); i++) {
             listData(munkakorok.item(i).getChildNodes(), "");
        }

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

```
      //Minden cĂ©g tulajdonosĂˇhoz hozzĂˇadom a sajĂˇt 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("SzĂ¶llĹ‘si JĂˇnos");
      }

      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.print(indent + "<" + node.getNodeName());
               if (node.hasAttributes()) {
                  for (int k = 0; k < node.getAttributes().getLength(); k++) {
                     Node attribute = node.getAttributes().item(k);
                     System.out.print("
"+attribute.getNodeName()+"=\""+attribute.getNodeValue()+"\"");
                  }
                  System.out.println(">");
               }else {
                  System.out.println(">");
               }

               NodeList nodeList_new = node.getChildNodes();
               listData(nodeList_new, indent);
```

```
                System.out.println(indent + "</" + node.getNodeName() + ">");
            } else if (node instanceof Text){
                String value = node.getNodeValue().trim();
                if (value.isEmpty()){
                    continue;
                }
                System.out.println(indent + node.getTextContent());
            }
        }
    }
  }

}
```

2d) Adatírás:

- A main metódus létrehozza az új xml filet, ahol a gyökérelem a „gyartocegek”. Minden elemhez van 1-1 metódus, amely létrehozza a megadott paraméterek alapján az egyedet és hozzácsatolja az xml file-unkhoz. Miután minden egyedet létrehoztunk a metódusokkal a saját elemeikkel és attribútumaikkal együtt kiírjuk őket a console-ra és elmentjük „XMLBC6X4X1.xml” néven a printDocument metódus segítségével.

```
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 javax.xml.transform.OutputKeys;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.Arrays;
import java.util.List;
import java.util.StringJoiner;
import org.w3c.dom.*;
public class DomWriteBC6X4X {

        public static void main(String[] args) {
                try {
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        DocumentBuilder builder = factory.newDocumentBuilder();
        Document document = builder.newDocument();
        Element rootElement = document.createElement("gyartocegek");
        rootElement.setAttribute("xmlns:xsi",          "http://www.w3.org/2001/XMLSchema-
instance");
        rootElement.setAttribute("xsi:noNamespaceSchemaLocation",
"XMLSchemaBC6X4X.xsd");
        document.appendChild(rootElement);
        cegek(document,rootElement,"1","ABC
Zrt","Zrt","2010","02","15",Arrays.asList("Kovács István","Nagy Mária","Szabó János"));
```

```
        cegek(document,rootElement,"2","DEF
Kft","Kft","2015","06","10",Arrays.asList("Kiss Andrea","Nagy Gábor","Kovács Anikó"));
        cegek(document,rootElement,"3","GHI
Bt","Bt","2018","11","25",Arrays.asList("Nagy Péter","Kiss Judit","Kovács Bence"));

        telepules(document,rootElement,"1","1098","Pécs","Király utca","25");
        telepules(document,rootElement,"2","1024","Budapest","Alkotás utca","8");
        telepules(document,rootElement,"3" ,"3300","Eger","Széchenyi utca","12");

        ceg_telepules(document,rootElement,"1" ,"1","5000" );
        ceg_telepules (document,rootElement,"2" ,"2" ,"4000");
        ceg_telepules (document,rootElement,"3" ,"3","3000");

        termek(document,rootElement,"1","Laptop
XYZ","1500",Arrays.asList("Memória","Processzor","Tárhely"),Arrays.asList("Andrásné","
Kiss Géza","Nagy Balázs"));
        termek(document,rootElement,"2","Smartphone
Plus","800",Arrays.asList("Kijelző","Akku","Kamera"),Arrays.asList("Nagy    Katalin","Kiss
József","Szabó Anna"));
        termek(document,rootElement,"3","Asztali                    Számítógép
Pro","2500",Arrays.asList("Processzor","RAM","GPU"),Arrays.asList("Kiss    Péter","Nagy
Zoltán","Szabó Eszter"));

        ceg_termek(document,rootElement,"1","1");
        ceg_termek(document,rootElement,"2","2");
        ceg_termek(document,rootElement,"3","3");

        gyartasi(document,rootElement,"1","1","800","24","3%","25");
        gyartasi(document,rootElement,"2","2","400","25","2%","15");
        gyartasi(document,rootElement,"3","3","1200","26","1%","30");

        dolgozo(document,rootElement,"1","1","Kis                    Anikó","2021-08-
05","Rendszergazda","3500");
        dolgozo(document,rootElement,"2","2","Nagy                    Gábor","2022-01-
15","Programozó","4200");
        dolgozo(document,rootElement,"3","3","Szabó          Mónika","2023-05-20","HR
Menedzser","3800");

        TransformerFactory transformerFactory = TransformerFactory.newInstance();
        Transformer transformer = transformerFactory.newTransformer();
        transformer.setOutputProperty(OutputKeys.ENCODING, "UTF-8");
        transformer.setOutputProperty(OutputKeys.INDENT, "yes");

        printDocument(document);
                } catch (Exception e) {
        e.printStackTrace();
    }
        }
```

```
        private static void cegek(Document document, Element rootElement, String cegID,
String nev, String tipus, String ev, String honap, String nap, List<String> tulajdonosok) {
                Element ceg = document.createElement("ceg");
                ceg.setAttribute("cegID", cegID);

                Element nevE = createElement(document, "nev", nev);
                ceg.appendChild(nevE);
                Element tipusE = createElement(document, "tipus", tipus);
                ceg.appendChild(tipusE);
                Element alapitasIdejeE = document.createElement("alapitasIdeje");
                Element eve = createElement(document, "ev", ev);
                Element honape = createElement(document, "honap", honap);
                Element nape = createElement(document, "nap", nap);
                alapitasIdejeE.appendChild(eve);
                alapitasIdejeE.appendChild(honape);
                alapitasIdejeE.appendChild(nape);
                ceg.appendChild(alapitasIdejeE);
                Element tulajdonosokE = document.createElement("tulajdonosok");
                for (String s : tulajdonosok) {
            Element temp = createElement(document, "tulajdonos", s);
            tulajdonosokE.appendChild(temp);
        }
                ceg.appendChild(tulajdonosokE);
                rootElement.appendChild(ceg);
            }

        private static void telepules(Document document, Element rootElement, String
telepulesid, String iranyitoszam, String nev, String utca, String szam) {
                Element telepules = document.createElement("telepules");
                telepules.setAttribute("telepulesID", telepulesid);

                Element irE = createElement(document, "iranyitoszam", iranyitoszam);
                telepules.appendChild(irE);
                Element nevE = createElement(document, "telepulesNeve", nev);
                telepules.appendChild(nevE);
                Element utcaE = createElement(document, "utca", utca);
                telepules.appendChild(utcaE);
                Element szamE = createElement(document, "hazszam", szam);
                telepules.appendChild(szamE);

                rootElement.appendChild(telepules);
            }

        private static void ceg_telepules(Document document, Element rootElement, String
cegref, String telepulesref, String alkalmazottak) {
                Element ceg_telepules = document.createElement("ceg_telepules");
                ceg_telepules.setAttribute("cegREF", cegref);
                ceg_telepules.setAttribute("telepulesREF", telepulesref);
                Element alkalmazottakE = createElement(document, "alkalmazottak",
alkalmazottak);
```

```
                ceg_telepules.appendChild(alkalmazottakE);

                rootElement.appendChild(ceg_telepules);
        }

        private static void termek(Document document, Element rootElement, String termekid, String nev, String ar, List<String> alkatreszek, List<String> vevok) {
                Element termek = document.createElement("termek");
                termek.setAttribute("termekID", termekid);

                Element nevE = createElement(document, "termekNeve", nev);
                termek.appendChild(nevE);
                Element arE = createElement(document, "eladasiAr", ar);
                termek.appendChild(arE);
                Element alkE = document.createElement("alkatreszek");
                for (String s : alkatreszek) {
            Element temp = createElement(document, "alkatresz", s);
            alkE.appendChild(temp);
        }
                termek.appendChild(alkE);

                Element vevokE = document.createElement("vevok");
                for (String s : vevok) {
            Element temp = createElement(document, "vevo", s);
            vevokE.appendChild(temp);
        }
                termek.appendChild(vevokE);
                rootElement.appendChild(termek);
        }


        private static void ceg_termek(Document document, Element rootElement, String cegref, String termekref) {
                Element ceg_termek = document.createElement("ceg_termek");
                ceg_termek.setAttribute("cegREF", cegref);
                ceg_termek.setAttribute("termekREF", termekref);

                rootElement.appendChild(ceg_termek);
        }


        private static void gyartasi(Document document, Element rootElement, String id, String termekREF, String koltseg, String ido, String selejtszam,String dolgozoszam) {
                Element gyartasi = document.createElement("gyartasiInformacio");
                gyartasi.setAttribute("gyartasiInformacioID", id);
                gyartasi.setAttribute("termekREF", termekREF);

                Element koltsegE = createElement(document, "gyartasiKoltseg", koltseg);
                gyartasi.appendChild(koltsegE);
                Element gyartasiIdoE = createElement(document, "gyartasiIdo", ido);
```

```
                gyartasi.appendChild(gyartasiIdoE);
                Element selejtekSzamaE = createElement(document, "selejtekSzama",
selejtszam);
                gyartasi.appendChild(selejtekSzamaE);
                Element dolgozokSzamaE = createElement(document, "dolgozokSzama",
dolgozoszam);
                gyartasi.appendChild(dolgozokSzamaE);

                rootElement.appendChild(gyartasi);
        }

        private static void dolgozo(Document document, Element rootElement, String id,
String cegref, String nev, String datum, String munkakor,String fizetes) {
                Element dolgozo = document.createElement("dolgozo");
                dolgozo.setAttribute("dolgozoID", id);
                dolgozo.setAttribute("cegREF", cegref);

                Element nevE = createElement(document, "dolgozoNeve", nev);
                dolgozo.appendChild(nevE);
                Element belepesiDatumE = createElement(document, "belepesiDatum",
datum);
                dolgozo.appendChild(belepesiDatumE);
                Element munkakorE = createElement(document, "munkakor", munkakor);
                dolgozo.appendChild(munkakorE);
                Element fizetesE = createElement(document, "fizetes", fizetes);
                dolgozo.appendChild(fizetesE);

                rootElement.appendChild(dolgozo);
        }


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


        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.print(indent + "<" + node.getNodeName());
                        if (node.hasAttributes()) {
                            for (int k = 0; k < node.getAttributes().getLength(); k++) {
```

```
                        Node attribute = node.getAttributes().item(k);
                        System.out.print("
"+attribute.getNodeName()+"=\""+attribute.getNodeValue()+"\"");
                    }
                    System.out.println(">");
                }else {
                   System.out.println(">");
                }

                NodeList nodeList_new = node.getChildNodes();
                listData(nodeList_new, indent);
                System.out.println(indent + "</" + node.getNodeName() + ">");
            } else if (node instanceof Text){
                String value = node.getNodeValue().trim();
                if (value.isEmpty()){
                    continue;
                }
                System.out.println(indent + node.getTextContent());
            }
        }
    }
}

    private static void printDocument(Document document) {
        try {
            File xmlFile = new File("XMLBC6X4X1.xml");

            FileWriter writer =new FileWriter(xmlFile, false);

            System.out.print("<?xml version=\"1.0\" encoding=\"utf-8\"
standalone=\"no\"?>\n");
            writer.write("<?xml version=\"1.0\" encoding=\"utf-8\"
standalone=\"no\"?>\n");

            System.out.print("<gyartocegek
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:noNamespaceSchemaLocation=\"XMLSchemaBC6X4X.xsd\">\n");
            writer.write("<gyartocegek
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:noNamespaceSchemaLocation=\"XMLSchemaBC6X4X.xsd\">\n");

            NodeList nodeList = document.getDocumentElement().getChildNodes();
            listData(nodeList, "");

            try {
                TransformerFactory transformerFactory =
TransformerFactory.newInstance();
                Transformer transformer = transformerFactory.newTransformer();
                transformer.setOutputProperty(OutputKeys.INDENT, "yes");
```

```
                    transformer.transform(new DOMSource(document), new
StreamResult("XMLBC6X4X1.xml"));

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

                System.out.print("</gyartocegek>");
                writer.close();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }

}
```