

JEGYZŐKÖNYV

Adatkezelés XML környezetben

2022. ősz féléves feladat

Készítette: **Tózsér Zétény**

Neptunkód: **QGNLD2**

Dátum: **2022.11.27**

Tartalomjegyzék

Tartalom

Feladat leírása.....	2
1. feladat	2
1a).....	3
1b).....	3
1c).....	3
1d).....	7
2. feladat	11
2a).....	11
2b).....	16
2c).....	20

Feladat leírása

A feladat futball csapatokat követ. Tagjaikat, az edzőket és közös meccseiket.

Minden játékosnak van egyedi kódja: `jatekosID`. Ezen felül a játékosban ott van a neve, mezszáma és hajszíne. A játékos tagja egy csapatnak, ezért még a csapat kódja is ott van, idegen kulcsként.

Az edzoben az edzo kódja: `edzoID`, neve, hajszíne és életkora található. Minden csapatnak egy edzője van, a csapat kulcsa van az edzoben, mint idegen kulcs.

Már említettem, hogy a csapatoknak is van azonosítója, ami pedig a `csapatID`. A csapatoknak is van neve, emellett a városuk és a mezük színe van feljegyezve.

A meccseknek is van ID-je, a `meccsID`. A meccseken lehetnek gollövők, többen is. A gólok száma is fel van jegyezve. A meccsekre eladott jegyek száma is tárolt.

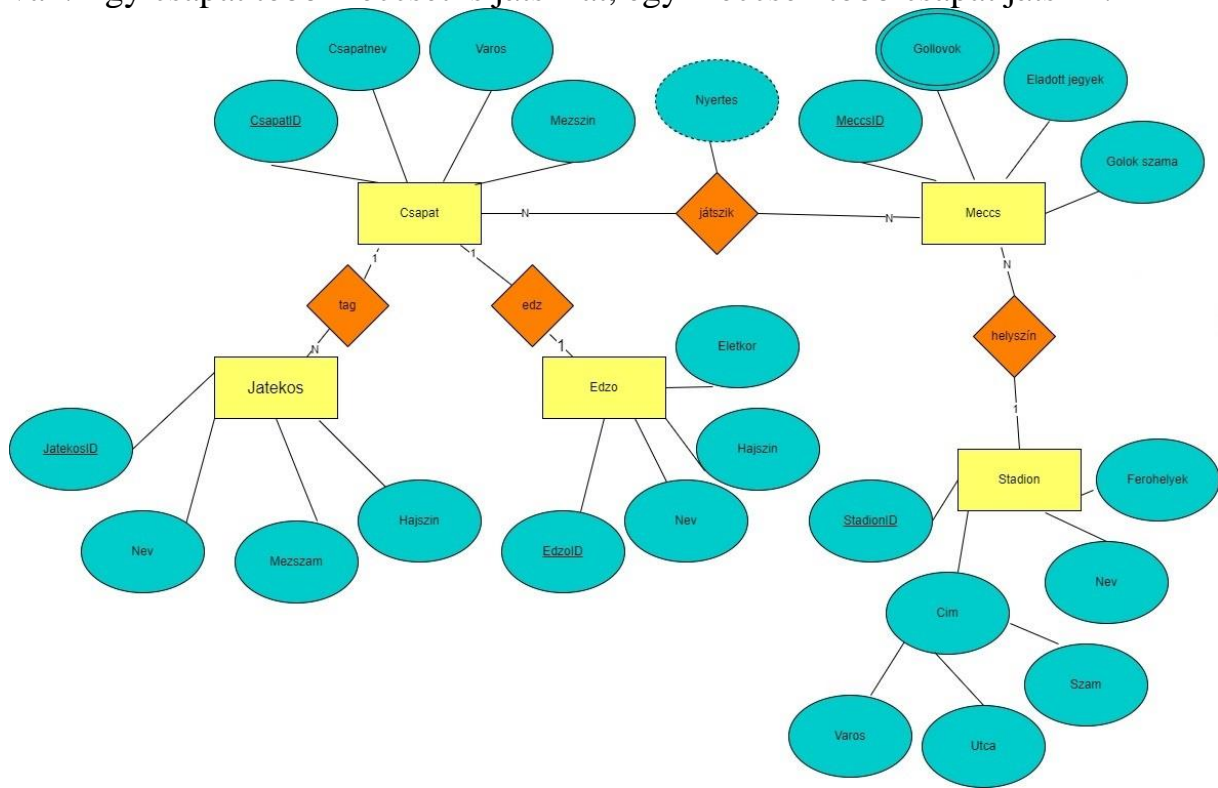
A csapat–meccs kapcsolat több-több kapcsolat, az adott meccs nyertese kapcsolódik hozzá.

A stadionoknak már van neve, ID-je is. A stadionok címe több elemű (város, utca, szám). Még a stadion kapacitása is tárolt (férőhelyek).

1. feladat

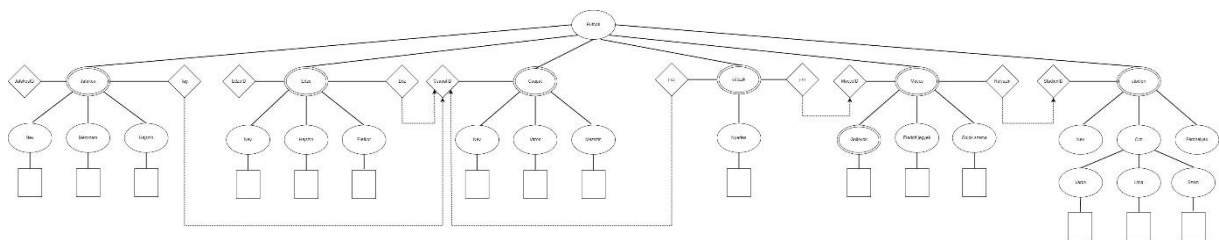
1a)

A modell labdarúgó csapatokat (tagjaikat és edzőiket), a meccseiket követi nyomon (helyszínnel együtt). Minden csapatnak több tagja, de csak egy edzője van. Egy csapat több meccset is játszhat, egy meccsen több csapat játszik.



1b)

Az modell a futball összefogó nevet kapta. A csapat-meccs kapcsolatból kapcsoló"tábla" lett.



1c)

Az xml dokumentumban először a csapatok szerepelnek, mivel a jatekos és az edzo is használja a kulcsait. A stadion pedig a meccs előtt, hasonló módon.

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

<futball noNamespaceSchemaLocation="XMLSchemaQGNLD2.xsd">
```

```
<!-- csapatok-->
<csapat csapatID="c1">
  <nev>ABCFC</nev>
  <varos>Budapest</varos>
  <mezzsin>piros</mezzsin>
</csapat>

<csapat csapatID="c2">
  <nev>MTK</nev>
  <varos>Miskolc</varos>
  <mezzsin>zold</mezzsin>
</csapat>

<csapat csapatID="c3">
  <nev>HalmaJSC</nev>
  <varos>HalmaJ</varos>
  <mezzsin>kek</mezzsin>
</csapat>

<!-- edzok-->
<edzo edzoID="e1" csapat-idegen-kulcs="c1">
  <nev>Soma</nev>
  <hajszin>barna</hajszin>
  <eletkor>28</eletkor>
</edzo>

<edzo edzoID="e2" csapat-idegen-kulcs="c2">
  <nev>Samu</nev>
  <hajszin>fekete</hajszin>
  <eletkor>30</eletkor>
</edzo>

<edzo edzoID="e3" csapat-idegen-kulcs="c3">
  <nev>Pista</nev>
  <hajszin>voros</hajszin>
  <eletkor>29</eletkor>
</edzo>

<!-- jatekosok-->
<jatekos jatekosID="j1" csapat-idegen-kulcs="c1">
  <nev>Jose</nev>
  <mezzsam>12</mezzsam>
  <hajszin>szoke</hajszin>
</jatekos>

<jatekos jatekosID="j2" csapat-idegen-kulcs="c1">
  <nev>Armando</nev>
  <mezzsam>11</mezzsam>
  <hajszin>barna</hajszin>
```

```
</jatekos>

<jatekos jatekosID="j3" csapat-idegen-kulcs="c2">
  <nev>Abel</nev>
  <mezzsam>9</mezzsam>
  <hajszin>barna</hajszin>
</jatekos>

<jatekos jatekosID="j4" csapat-idegen-kulcs="c2">
  <nev>Rokus</nev>
  <mezzsam>11</mezzsam>
  <hajszin>szoke</hajszin>
</jatekos>

<jatekos jatekosID="j5" csapat-idegen-kulcs="c3">
  <nev>Abel</nev>
  <mezzsam>8</mezzsam>
  <hajszin>voros</hajszin>
</jatekos>

<jatekos jatekosID="j6" csapat-idegen-kulcs="c3">
  <nev>Alfonz</nev>
  <mezzsam>12</mezzsam>
  <hajszin>szoke</hajszin>
</jatekos>

<!-- stadionok-->
<stadion stadionID="s1">
  <nev>ABC-Arena</nev>
  <cim>
    <varos>Budapest</varos>
    <utca>Vamhaz</utca>
    <szam>9</szam>
  </cim>
  <ferohelyek>20000</ferohelyek>
</stadion>

<stadion stadionID="s2">
  <nev>Duhongo</nev>
  <cim>
    <varos>Miskolc</varos>
    <utca>Kiraly</utca>
    <szam>22</szam>
  </cim>
  <ferohelyek>18000</ferohelyek>
</stadion>

<stadion stadionID="s3">
  <nev>Halmaj-Stadion</nev>
```

```
<cim>
  <varos>Halmaj</varos>
  <utca>Danko-Pista</utca>
  <szam>37</szam>
</cim>
<ferohelyek>25000</ferohelyek>
</stadion>

<!-- meccsek-->
<meccs meccsID="m1" stadion-idegen-kulcs="s1">
  <gollovok>j1</gollovok>
  <gollovok>j3</gollovok>
  <eladott-jegyek>12000</eladott-jegyek>
  <golok-szama>2</golok-szama>
</meccs>

<meccs meccsID="m2" stadion-idegen-kulcs="s2">
  <gollovok>j5</gollovok>
  <eladott-jegyek>18000</eladott-jegyek>
  <golok-szama>3</golok-szama>
</meccs>

<meccs meccsID="m3" stadion-idegen-kulcs="s3">
  <gollovok>j2</gollovok>
  <eladott-jegyek>15000</eladott-jegyek>
  <golok-szama>1</golok-szama>
</meccs>

<!-- jatszik-->
<jatszik jID="01" meccs-idegen-kulcs="m1" csapat-idegen-kulcs="c1">
  <nyertes></nyertes>
</jatszik>

<jatszik jID="02" meccs-idegen-kulcs="m1" csapat-idegen-kulcs="c2">
  <nyertes></nyertes>
</jatszik>

<jatszik jID="03" meccs-idegen-kulcs="m2" csapat-idegen-kulcs="c3">
  <nyertes>c3</nyertes>
</jatszik>

<jatszik jID="04" meccs-idegen-kulcs="m2" csapat-idegen-kulcs="c2">
  <nyertes>c3</nyertes>
</jatszik>

<jatszik jID="05" meccs-idegen-kulcs="m3" csapat-idegen-kulcs="c1">
  <nyertes>c1</nyertes>
</jatszik>
```

```
<jatszik jID="06" meccs-idegen-kulcs="m3" csapat-idegen-kulcs="c3">
  <nyertes>c1</nyertes>
</jatszik>
```

1d)

A sémában először az egyszerű típusok szerepelnek, vagyis a példányok gyerekelemi. Utána kezdődik a „futball”, benne a komplex típusokkal. A játszik után, még a futballon belül szerepelnek a kulcsok és idegen kulcsok.

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

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

  <!-- egyszerű típusok-->

  <xs:simpleType name="nev-type">
    <xs:restriction base="xs:string"/>
  </xs:simpleType>

  <xs:simpleType name="hajszin-type">
    <xs:restriction base="xs:string"/>
  </xs:simpleType>

  <xs:simpleType name="mezzsam-type">
    <xs:restriction base="xs:int">
      <xs:minExclusive value="1"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="eletkor-type">
    <xs:restriction base="xs:int">
      <xs:minExclusive value="0"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="varos-type">
    <xs:restriction base="xs:string"/>
  </xs:simpleType>

  <xs:simpleType name="mezzsin-type">
    <xs:restriction base="xs:string"/>
  </xs:simpleType>

  <xs:simpleType name="eladott-jegyek-type">
    <xs:restriction base="xs:int">
```

```

        <xs:minInclusive value="0"/>
    </xs:restriction>
</xs:simpleType>

<xs:simpleType name="gollovok-type">
    <xs:restriction base="xs:string"/>
</xs:simpleType>

<xs:simpleType name="golok-szama-type">
    <xs:restriction base="xs:int">
        <xs:minExclusive value="0"/>
    </xs:restriction>
</xs:simpleType>

<xs:simpleType name="utca-type">
    <xs:restriction base="xs:string"/>
</xs:simpleType>

<xs:simpleType name="szam-type">
    <xs:restriction base="xs:int">
        <xs:minExclusive value="1"/>
    </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ferohelyek-type">
    <xs:restriction base="xs:int">
        <xs:minExclusive value="0"/>
    </xs:restriction>
</xs:simpleType>

<!-- -->

<xs:element name="futball">
    <xs:complexType>
        <xs:sequence>

            <xs:element name="csapat" maxOccurs="unbounded">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="nev"/>
                        <xs:element name="varos"/>
                        <xs:element name="mezzsin"/>
                    </xs:sequence>
                    <xs:attribute name="csapatID" type="xs:string"
use="required"/>
                </xs:complexType>
            </xs:element>

            <xs:element name="edzo" maxOccurs="unbounded">

```



```

        <xs:complexType>
            <xs:sequence>
                <xs:element name="nev"/>
                <xs:element name="hajszin"/>
                <xs:element name="eletkor"/>
            </xs:sequence>
            <xs:attribute name="edzoID" type="xs:string"
use="required"/>
        </xs:complexType>
    </xs:element>

    <xs:element name="jatekos" maxOccurs="unbounded">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="nev"/>
                <xs:element name="mezszám"/>
                <xs:element name="hajszin"/>
            </xs:sequence>
            <xs:attribute name="jatekosID" type="xs:string"
use="required"/>
        </xs:complexType>
    </xs:element>

    <xs:element name="stadion" maxOccurs="unbounded">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="nev"/>
                <xs:element name="cim">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="varos"/>
                            <xs:element name="utca"/>
                            <xs:element name="szám"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="ferhelyek"/>
            </xs:sequence>
            <xs:attribute name="stadionID" type="xs:string"
use="required"/>
        </xs:complexType>
    </xs:element>

    <xs:element name="meccs" maxOccurs="unbounded">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="gollovok"/>
                <xs:element name="eladott-jegyek"/>
                <xs:element name="golok-szama"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

```

```

        </xs:sequence>
        <xs:attribute name="meccsID" type="xs:string"
use="required"/>
    </xs:complexType>
</xs:element>

    <xs:element name="jatszok" maxOccurs="unbounded">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="nyertes"/>
            </xs:sequence>
            <xs:attribute name="j-cs" type="xs:string"
use="required"/>
            <xs:attribute name="j-m" type="xs:string"
use="required"/>
        </xs:complexType>
    </xs:element>

</xs:sequence>
</xs:complexType>

<!-- kulcsok-->

<xs:key name="csapat-kulcs">
    <xs:selector xpath="csapat"/>
    <xs:field xpath="@csapatID"/>
</xs:key>

<xs:key name="edzo-kulcs">
    <xs:selector xpath="edzo"/>
    <xs:field xpath="@edzoID"/>
</xs:key>

<xs:key name="jatekos-kulcs">
    <xs:selector xpath="jatekos"/>
    <xs:field xpath="@jatekosID"/>
</xs:key>

<xs:key name="stadion-kulcs">
    <xs:selector xpath="stadion"/>
    <xs:field xpath="@stadionID"/>
</xs:key>

<xs:key name="meccs-kulcs">
    <xs:selector xpath="meccs"/>
    <xs:field xpath="@meccsID"/>
</xs:key>

<!-- idegen kulcsok-->

```

```

<xs:keyref refer="csapat-kulcs" name="csapat-idegen-kulcs">
  <xs:selector xpath="csapat"/>
  <xs:field xpath="@csapatID"/>
</xs:keyref>

<xs:keyref refer="meccs-kulcs" name="meccs-idegen-kulcs">
  <xs:selector xpath="meccs"/>
  <xs:field xpath="@meccsID"/>
</xs:keyref>

<xs:keyref refer="stadion-kulcs" name="stadion-idegen-kulcs">
  <xs:selector xpath="stadion"/>
  <xs:field xpath="@stadionID"/>
</xs:keyref>

</xs:element>

</xs:schema>

```

2. feladat

2a)

A DomRead belovassa és struktúráltan írja ki a konzolra az xml tartalmát. Közben létrehoz egy .txt fájlt és oda is írja.

```
package hu.domparse.QGNLD2;
```

```
import java.io.*;
```

```
import javax.xml.parsers.DocumentBuilderFactory;
```

```
import javax.xml.parsers.DocumentBuilder;
```

```
import javax.xml.parsers.ParserConfigurationException;
```

```
import org.xml.sax.SAXException;
```

```
import org.w3c.dom.Document;
```

```
import org.w3c.dom.NodeList;
```

```
import org.w3c.dom.Node;
import org.w3c.dom.Element;

public class DomReadQGNLD2 {

    public static void main(String argv[]) throws SAXException,
        IOException, ParserConfigurationException {

        //Elemek es gyerekelemeik.

        String[] csapatFields = {

            "nev",

            "varos",

            "mezzsin"

        };

        String[] edzoFields = {

            "nev",

            "hajszin",

            "eletkor"

        };

        String[] jatekosFields = {

            "nev",

            "mezzsam",

            "hajszin"

        };

        String[] stadionFields = {
```

```
        "nev",  
        "cim",  
        "ferohelyek"  
    };
```

```
String[] meccsFields = {  
    "gollovok",  
    "eladott-jegyek",  
    "golok-szama"  
};
```

```
String[] jatszFields = {  
    "nyertes"  
};
```

```
String[][] fields = {  
    csapatFields,  
    edzoFields,  
    jatekosFields,  
    stadionFields,  
    meccsFields,  
    jatszFields  
};
```

```
String [] subRoots = {  
    "csapat",
```

```
        "edzo",  
        "jatekos",  
        "stadion",  
        "meccs",  
        "jatszok"  
    };
```

```
String [] idList = {  
    "csapatID",  
    "edzoID",  
    "jatekosID",  
    "stadionID",  
    "meccsID",  
    "jID"  
};
```

```
//XML file megnyitasa.
```

```
File xmlFile = new File("XMLQGNLD2.xml");
```

```
//TXT file letrehozasa, iras elokeszites.
```

```
File txt = new File("DomRead.txt");
```

```
FileWriter fw = new FileWriter(txt);
```

```
PrintWriter pw = new PrintWriter(fw);
```

```
DocumentBuilderFactory factory =  
DocumentBuilderFactory.newInstance();
```

```
DocumentBuilder dBuilder = factory.newDocumentBuilder();
```

```
Document doc = dBuilder.parse(xmlFile);
```

```
doc.getDocumentElement().normalize();
```

```

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

        //Txt-be is iras.

        pw.write( "Root element: " +
doc.getDocumentElement().getNodeName() + "\n" );


int index = 0;

//Az xml-en vegigmenni.
for(String element : subRoots) {
    NodeList nList = doc.getElementsByTagName(element);

    for (int i = 0; i < nList.getLength(); i++){
        Node nNode = nList.item(i);
        //Uj elemnel irni konzolra es fajlba.
        System.out.println("\nCurrent Element: " +
nNode.getNodeName());
        pw.write( "\nCurrent Element: " +
nNode.getNodeName() + "\n");
        //A fieldek segitsegevel gyerekelemek kiirasa.
        if (nNode.getNodeType() ==
Node.ELEMENT_NODE) {
            Element elem = (Element) nNode;
            String uid = elem.getAttribute(idList[index]);
            System.out.println(idList[index] + ": " + uid);
            pw.write( idList[index] + ": " + uid + "\n");
            for(String field : fields[index]){
                Node node =
elem.getElementsByTagName(field).item(0);
                String data = node.getTextContent();

```

```

        System.out.println(field + ": " + data);
        pw.write( field + ": " + data + "\n");
    }
}
}
index++;
}
//Fajl iro bezarasa.
pw.close();
}
}

```

2b)

A DomModify nem menti el a változtatásokat fájlba, csupán a konzolra írja ki a megváltoztatott xml dokumentumot.

```

package hu.domparse.QGNLD2;

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

import javax.xml.transform.*;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import javax.xml.parsers.*;

import org.w3c.dom.Document;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;

```



```

import org.w3c.dom.Element;
import org.w3c.dom.NamedNodeMap;

public class DomModifyQGNLD2 {
    public static void main(String argv[]) throws
ParserConfigurationException,
    TransformerException, IOException, TransformerConfigurationException
    {
        try {
            File inputFile = new File("XMLQGNLD2.xml");

            DocumentBuilderFactory docFactory =
DocumentBuilderFactory.newInstance();

            DocumentBuilder docBuilder = docFactory.newDocumentBuilder();

            Document doc = docBuilder.parse(inputFile);
            NodeList edzoLista = doc.getElementsByTagName("edzo");
            //Edzolistan vegigmenni.
            for(int i = 0; i < edzoLista.getLength(); i++) {
                Node edzo = doc.getElementsByTagName("edzo").item(i);
                NamedNodeMap attr = edzo.getAttributes();
                Node nodeAttr = attr.getNamedItem("edzoID");
                //ID Változtatás
                nodeAttr.setTextContent("e" + (i));
                //Az edzo gyerekelemein vegigmenni
                NodeList lista = edzo.getChildNodes();
                for(int t = 0; t < lista.getLength(); t++) {
                    Node node = lista.item(t);

```

```

if (node.getNodeType() == Node.ELEMENT_NODE) {
    Element element = (Element) node;
    //Ha a hajszi elemre er...
    if ("hajszi".equals(element.getNodeName())) {
        //es a hajszi erteke barna...
        if ("barna".equals(element.getTextContent())){
            //akkor voros lesz.
            element.setTextContent("voros");
        }
    }
}

```

```

}
}

```

//Elso meccs torlese.

```

NodeList meccsLista = doc.getElementsByTagName("meccs");

```

```

for(int i = 0; i < meccsLista.getLength(); i++) {
    Node meccs = meccsLista.item(i);
    Element element = (Element) meccs;

```

```

if(element.getAttribute("meccsID").equals("m1")) {

```

```

    element.getParentNode().removeChild(element);

```

```

        break;

```

```

    }

```

```

    }

    //Meccsekhez szabalytalansagok szama szama.

    for (int i = 0; i < meccsLista.getLength(); i++)
    {
        Node meccs = meccsLista.item(i);
        //szabalytalansagok elem letrehozasa
        Element szabalytalansagok =
doc.createElement("szabalytalansagok");
        meccs.appendChild(szabalytalansagok);
        //Minden meccsnek 4et allit be.
        szabalytalansagok.appendChild(doc.createTextNode("4"));
    }

    TransformerFactory transformerFactory =
TransformerFactory.newInstance();
    Transformer transformer = transformerFactory.newTransformer();

    DOMSource source = new DOMSource(doc);

    System.out.println("--Results--");
    StreamResult consoleResult = new StreamResult(System.out);
    transformer.transform(source, consoleResult);
    }catch (Exception e){
e.printStackTrace();
}

```

```
    }  
}
```

2c)

A DomQuery csak a lekérdezések eredményét írja ki a konzolra.

```
package hu.domparse.QGNLD2;
```

```
import java.io.*;
```

```
import javax.xml.parsers.DocumentBuilderFactory;
```

```
import javax.xml.parsers.DocumentBuilder;
```

```
import javax.xml.parsers.ParserConfigurationException;
```

```
import org.xml.sax.SAXException;
```

```
import org.w3c.dom.Document;
```

```
import org.w3c.dom.NodeList;
```

```
import org.w3c.dom.Node;
```

```
import org.w3c.dom.Element;
```

```
public class DomQueryQGNLD2 {
```

```
    public static void main(String argv[]) throws SAXException,  
        IOException, ParserConfigurationException {
```

```
        File xmlFile = new File("XMLQGNLD2.xml");
```

```

        DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();

        DocumentBuilder dBuilder = factory.newDocumentBuilder();

String[] jatekosFields = {
            "nev",
            "mezzsam",
            "hajszin"
};

//A file parse-olasa.
Document doc = dBuilder.parse(xmlFile);
doc.getDocumentElement().normalize();

//Edzok adatai (minusz az ID).
System.out.println("Edzok adatai (minusz az ID)\n");

NodeList edzoLista = doc.getElementsByTagName("edzo");
//Forral vegig az edzokon.
for(int i = 0; i < edzoLista.getLength(); i++) {
    Node e = edzoLista.item(i);
    if(e.getNodeType() == Node.ELEMENT_NODE) {
        Element edzo = (Element) e;
        NodeList nevList = edzo.getChildNodes();
        System.out.println("Edzo: ");
        //Az edzo gyerekelemein vegigmenni.
        for(int j = 0; j < nevList.getLength(); j++) {
            Node n = nevList.item(j);

```

```

        if
(n.getNodeType()==Node.ELEMENT_NODE) {
            Element nev = (Element) n;
            System.out.println(nev.getTagName() +
"= " + nev.getTextContent());
        }
    }
    System.out.println("");
}
}

```

//12-es mezszamu jatekosok neveii.

```
System.out.println("12-es mezszamu jatekosok nevei.\n");
```

```
NodeList jatekosLista = doc.getElementsByTagName("jatekos");
```

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

```
    Node jatekos = jatekosLista.item(i);
```

```
    if(jatekos.getNodeType() == Node.ELEMENT_NODE) {
```

```
        Element element = (Element) jatekos;
```

```
        Node n =
element.getElementsByTagName("mezszam").item(0);
```

```
        String mezszam = n.getTextContent();
```

```
        if("12".equals(mezszam)) {
```

```
            Node node =
element.getElementsByTagName("nev").item(0);
```

```
            String nev = node.getTextContent();
```

```

        System.out.println("Nev: " + nev);
    }
}

System.out.println("");

//Egyes csapatban játszo jatekosok.
System.out.println("Egyes csapatban játszo jatekosok.\n");

//Vegigmenni a jatekosokon.
for (int i = 0; i < jatekosLista.getLength(); i++){
    Node jatekos = jatekosLista.item(i);
    if (jatekos.getNodeType() == Node.ELEMENT_NODE) {
        Element element = (Element) jatekos;
        String jatekosID = element.getAttribute("jatekosID");
        //A csapat kulcs ellenorzese.
        String csapata = element.getAttribute("csapat-idegen-
kulcs");

        if (!csapata.contains("c1")) {
            continue;
        }

        System.out.println("jatekosID" + ": " + jatekosID);
        //A jatekos mezovel a jatekos adatainak kiirasa.
        for(String field : jatekosFields){
            Node node =
element.getElementsByTagName(field).item(0);
            String data = node.getTextContent();
            System.out.println(field + ": " + data);
        }
    }
}

```

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