

Jegyzőkönyv  
Adatkezelés XML-ben  
Féléves feladat  
Airbnb lakásközvetítő cég adatbázisa

Készítette: **Tamás Janka**  
Neptun-kód: **GIE0EJ**  
Gyakorlat: **szerda, 10:00-12:00**  
Gyakorlat vezető: **Dr. Bednarik László**  
Kelt: 2021.11.30.

## Tartalom

Feladat leírása: .....	3
Az ER modell egyedei és tulajdonságai: .....	3
Egyedek közötti kapcsolat.....	4
1. feladat .....	5
1a, ER modell .....	5
1b, XDM modell.....	6
1c, XML fájl: .....	6
1d, XML Schema.....	8
2. feladat .....	10
2a, Read fájl.....	10
Output: .....	13
2b, Query fájl.....	13
Output: .....	16
2c, Modify fájl.....	16
Output: .....	17

## Feladat leírása:

A tematika egy airbnb közvetítő cég adatbázisa, melyben egy airbnb lakás kiadásához szükséges adatok vannak nyilvántartva. Az adatbázis a lakásadatokon kívül nyilvántartja a bérlő vendégek és foglalásuk adatait, illetve a lakásokat takarító személyzetet is. Egy kiadás menete a következőképpen alakul: a vendég lefoglalja az általa kinézett lakást, majd a foglaláshoz az oldal vagy a cég egy dolgozója hozzárendel egy takarítót, aki a vendég távozása után kitakarítja a lakást.

## Az ER modell egyedei és tulajdonságai:

**Az egyedek a következők:** Lakás, Vendég, Foglalás, Takarító

### Egyedek tulajdonságai:

#### ➤ **Lakás:**

1. LakásID: elsődleges kulcs
2. Kapukód: a lakás főkapujának a belépőkódja
3. Cím: összetett tulajdonság, a lakás címét tárolja, azon belül irányítószám, település, utca, házszám, emelet és ajtó adatokat

#### ➤ **Vendég:**

1. VendégID: elsődleges kulcs
2. Név: a vendég neve
3. Telefonszám: többértékű tulajdonság, a tulajdonos telefonszáma(i), elérhetősége
4. Útlevél száma: a vendég útlevelének az azonosítója, ezzel azonosítja magát a személy (ez lehet még személyi igazolvány és jogosítvány is)

#### ➤ **Foglalás:**

1. FoglalásID: elsődleges kulcs
2. Fő: megadja, hogy hány főre szól a foglalás
3. Kezdő dátum: vendég érkezésének dátuma
4. Záró dátum: vendég távozásának dátuma
5. Éjszakák száma: származtatott tulajdonság, a „kezdő dátumnak” a „záró dátumból” való kivonásával számítható ki

#### ➤ **Takarító:**

1. TakarítóID: elsődleges kulcs
2. Név: a takarító neve
3. Személyi igazolvány szám: ez az adat azonosítja a személyt
4. Mobilszám: a takarító elérhetősége, mobilszáma

## Egyedek közötti kapcsolat

Lakás – Foglalás → 1:1 kapcsolat:

- egy lakás csak egy foglalásban szerepelhet
- egy foglaláshoz csak egy lakás tartozik

Lakás – Takarító → N:M kapcsolat:

- egy lakást több takarító takaríthat
- illetve egy takarító több lakást takaríthat

Takarító – Foglalás → 1:N kapcsolat:

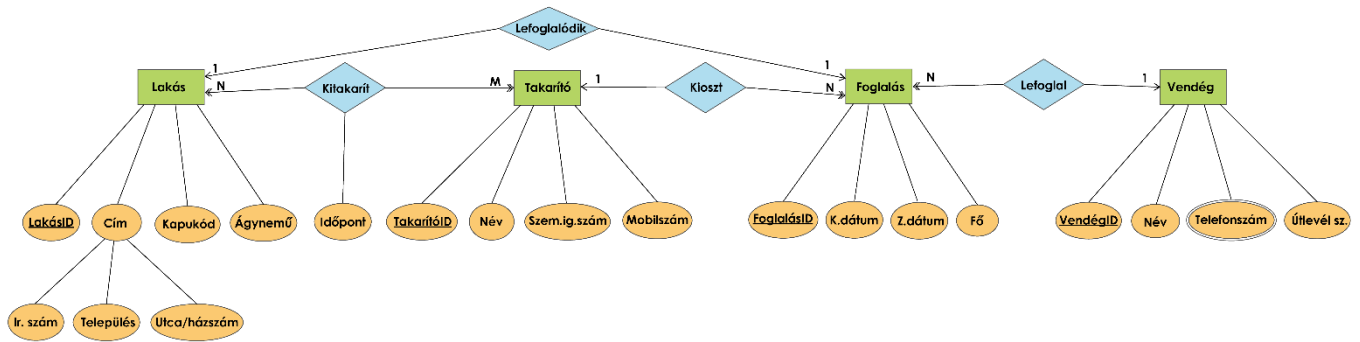
- egy takarítónak több foglalás is juthat, aminek a lakását ki kell takarítania
- viszont egy foglaláshoz csak egy takarítót oszt be a rendszer

Vendég – Foglalás → 1:N kapcsolat:

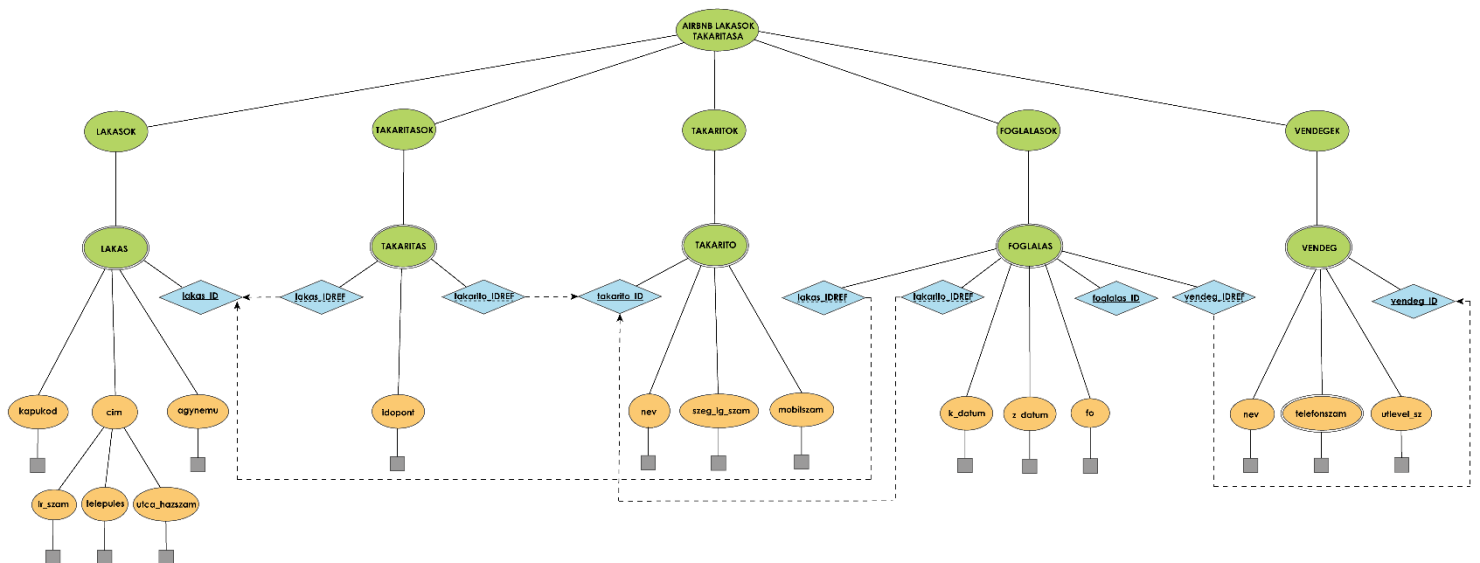
- egy vendég több foglalást is végre tud hajtani
- viszont egy foglaláshoz csak egy vendég tartozhat

## 1. feladat

### 1a, ER modell



## 1b, XDM modell



## 1c, XML fájl:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <airbnblakasoktakaritasi xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="XMLSchemaGIE0EJ.xsd">
4   <lakasok>
5     <lakas lakas_ID="1">
6       <kapukod>2345</kapukod>
7       <cim>
8         <ir_szam>1065</ir_szam>
9         <telepules>Budapest</telepules>
10        <utca_hazszam>Révay utca 8</utca_hazszam>
11      </cim>
12      <agynemu>szimpla</agynemu>
13    </lakas>
14    <lakas lakas_ID="2">
15      <kapukod>3456</kapukod>
16      <cim>
17        <ir_szam>1061</ir_szam>
18        <telepules>Budapest</telepules>
19        <utca_hazszam>Andrássy út 23</utca_hazszam>
20      </cim>
21      <agynemu>dupla</agynemu>
22    </lakas>
23    <lakas lakas_ID="3">
24      <kapukod>4585</kapukod>
25      <cim>
26        <ir_szam>1051</ir_szam>
27        <telepules>Budapest</telepules>
28        <utca_hazszam>Zrínyi utca 4</utca_hazszam>
29      </cim>
30    </lakas>
31  </lakasok>
32 </airbnblakasoktakaritasi>
  
```

```

29         </cim>
30         <agynemu>szimpla</agynemu>
31     </lakas>
32 </lakasok>
33 <takaritasok>
34     <takaritas lakas_IDREF="1" takarito_IDREF="3">
35         <idopont>2021-12-01</idopont>
36     </takaritas >
37     <takaritas lakas_IDREF="3" takarito_IDREF="2">
38         <idopont>2021-12-04</idopont>
39     </takaritas >
40     <takaritas lakas_IDREF="2" takarito_IDREF="1">
41         <idopont>2021-12-06</idopont>
42     </takaritas >
43 </takaritasok>
44 <takaritok>
45     <takarito takarito_ID="1">
46         <nev>Kiss Mária</nev>
47         <szem_ig_szam>231456PA</szem_ig_szam>
48         <mobilszam>06308573245</mobilszam>
49     </takarito>
50     <takarito takarito_ID="2">
51         <nev>Nagy Katalin</nev>
52         <szem_ig_szam>463782EE</szem_ig_szam>
53         <mobilszam>06204563998</mobilszam>
54     </takarito>
55     <takarito takarito_ID="3">
56         <nev>Szabó Péter</nev>
57         <szem_ig_szam>125478LD</szem_ig_szam>

```

```

57         <szem_ig_szam>125478LD</szem_ig_szam>
58         <mobilszam>06703456787</mobilszam>
59     </takarito>
60 </takaritok>
61 <foglalasok>
62     <foglalas foglalas_ID="1" lakas_IDREF="1" takarito_IDREF="3" vendeg_IDREF="1">
63         <k_datum>2021-11-29</k_datum>
64         <z_datum>2021-12-01</z_datum>
65         <fo>3</fo>
66     </foglalas>
67     <foglalas foglalas_ID="2" lakas_IDREF="3" takarito_IDREF="2" vendeg_IDREF="2">
68         <k_datum>2021-12-03</k_datum>
69         <z_datum>2021-12-04</z_datum>
70         <fo>2</fo>
71     </foglalas>
72     <foglalas foglalas_ID="3" lakas_IDREF="2" takarito_IDREF="1" vendeg_IDREF="3">
73         <k_datum>2021-12-04</k_datum>
74         <z_datum>2021-12-06</z_datum>
75         <fo>2</fo>
76     </foglalas>
77 </foglalasok>
78 <vendegek>
79     <vendeg vendeg_ID="1">
80         <nev>John Smith</nev>
81         <telefonszam>+223456785</telefonszam>
82         <telefonszam>06304567855</telefonszam>
83         <utlevel_sz>234567AA</utlevel_sz>
84     </vendeg>
85     <vendeg vendeg_ID="2">

```

```

85     <vendeg vendeg_ID="2">
86         <nev>Nagy Tamás</nev>
87         <telefonszam>06702874956</telefonszam>
88         <utlevel_sz>475839BD</utlevel_sz>
89     </vendeg>
90     <vendeg vendeg_ID="3">
91         <nev>Kovács Piroska</nev>
92         <telefonszam>06303455443</telefonszam>
93         <telefonszam>061223678</telefonszam>
94         <utlevel_sz>347656PL</utlevel_sz>
95     </vendeg>
96 </vendegek>
97 </airbnblakasoktakaritasa>

```

# 1d, XML Schema

```
1 <?xml version="1.0" encoding="UTF-8"?>
2<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
3  elementFormDefault="qualified"
4  attributeFormDefault="qualified">
5
6  <xs:element name="kapukod" type="xs:integer"/>
7  <xs:element name="ir_szam" type="xs:integer"/>
8  <xs:element name="telepules" type="xs:string"/>
9  <xs:element name="utca_hazszam" type="xs:string"/>
10 <xs:element name="agynemu" type="xs:string"/>
11 <xs:element name="idopont" type="xs:date"/>
12 <xs:element name="nev" type="xs:string"/>
13 <xs:element name="szem_ig_szam" type="xs:string"/>
14 <xs:element name="mobilszam" type="xs:string"/>
15 <xs:element name="k_datum" type="xs:date"/>
16 <xs:element name="z_datum" type="xs:date"/>
17 <xs:element name="fo" type="xs:integer"/>
18 <xs:element name="utlevel_sz" type="xs:string"/>
19 <xs:element name="telefonszam" type="xs:string"/>
20
21 <xs:attribute name="Lakas_ID" type="xs:integer"/>
22 <xs:attribute name="takarito_ID" type="xs:integer"/>
23 <xs:attribute name="Lakas_IDREF" type="xs:integer"/>
24 <xs:attribute name="takarito_IDREF" type="xs:integer"/>
25 <xs:attribute name="foglalas_ID" type="xs:integer"/>
26 <xs:attribute name="vendeg_IDREF" type="xs:integer"/>
27 <xs:attribute name="vendeg_ID" type="xs:integer"/>
28
29<xs:complexType name="cim_tipus">
30  <xs:sequence>
31    <xs:element ref="ir_szam"/>
32    <xs:element ref="telepules"/>
33    <xs:element ref="utca_hazszam"/>
34  </xs:sequence>
35 </xs:complexType>
36
37
38
39<xs:complexType name="lakasok_tipus">
40  <xs:sequence>
41    <xs:element name="Lakas" type="Lakas_tipus" maxOccurs="unbounded"/>
42  </xs:sequence>
43 </xs:complexType>
44
45<xs:complexType name="takaritasok_tipus">
46  <xs:sequence>
47    <xs:element name="takaritas" type="takaritas_tipus" maxOccurs="unbounded"/>
48  </xs:sequence>
49 </xs:complexType>
50
51<xs:complexType name="takaritok_tipus">
52  <xs:sequence>
53    <xs:element name="takarito" type="takarito_tipus" maxOccurs="unbounded"/>
54  </xs:sequence>
55 </xs:complexType>
56
57<xs:complexType name="foglalasok_tipus">
```



```

57<xs:complexType name="foglalasok_tipus">
58  <xs:sequence>
59    <xs:element name="foglalas" type="foglalas_tipus" maxOccurs="unbounded"/>
60  </xs:sequence>
61</xs:complexType>
62
63<xs:complexType name="vendegek_tipus">
64  <xs:sequence>
65    <xs:element name="vendeg" type="vendeg_tipus" maxOccurs="unbounded"/>
66  </xs:sequence>
67</xs:complexType>
68
69<xs:complexType name="vendeg_tipus">
70  <xs:sequence>
71    <xs:element ref="nev"/>
72    <xs:element ref="telefonszam" maxOccurs="unbounded" />
73    <xs:element ref="utlevel_sz"/>
74  </xs:sequence>
75  <xs:attribute ref="vendeg_ID" use="required"/>
76</xs:complexType>
77
78<xs:complexType name="foglalas_tipus">
79  <xs:sequence>
80    <xs:element ref="k_datum" />
81    <xs:element ref="z_datum" />
82    <xs:element ref="fo" />
83  </xs:sequence>
84  <xs:attribute ref="foglalas_ID" use="required"/>
85  <xs:attribute ref="lakas_IDREF" use="required"/>

```

```

85  <xs:attribute ref="lakas_IDREF" use="required"/>
86  <xs:attribute ref="takarito_IDREF" use="required"/>
87  <xs:attribute ref="vendeg_IDREF" use="required"/>
88</xs:complexType>
89
90<xs:complexType name="takarito_tipus">
91  <xs:sequence>
92    <xs:element ref="nev" />
93    <xs:element ref="szemig_szam" />
94    <xs:element ref="mobilszam" />
95  </xs:sequence>
96  <xs:attribute ref="takarito_ID" use="required"/>
97</xs:complexType>
98
99<xs:complexType name="takaritas_tipus">
100  <xs:sequence>
101    <xs:element ref="idopont" />
102  </xs:sequence>
103  <xs:attribute ref="takarito_IDREF" use="required"/>
104  <xs:attribute ref="lakas_IDREF" use="required"/>
105</xs:complexType>
106
107<xs:complexType name="lakas_tipus">
108  <xs:sequence>
109    <xs:element ref="kapukod" />
110    <xs:element name="cim" type="cim_tipus" />
111    <xs:element ref="agynemu" />
112  </xs:sequence>
113  <xs:attribute ref="lakas_ID" use="required"/>
114</xs:complexType>
115
116
117<xs:element name="airbnblakasoktakaritas">
118  <xs:complexType>
119    <xs:sequence>
120      <xs:element name="Lakasok" type="Lakasok_tipus"/>
121      <xs:element name="takaritasok" type="takaritasok_tipus"/>
122      <xs:element name="takaritok" type="takaritok_tipus"/>
123      <xs:element name="foglalasok" type="foglalasok_tipus"/>
124      <xs:element name="vendegek" type="vendegek_tipus"/>
125    </xs:sequence>
126  </xs:complexType>
127
128  <xs:key name="Lakas_PK">
129    <xs:selector xpath="Lakasok/Lakas"/>
130    <xs:field xpath="@Lakas_ID"/>
131  </xs:key>
132  <xs:key name="takarito_PK">
133    <xs:selector xpath="takaritok/takarito"/>
134    <xs:field xpath="@takarito_ID"/>
135  </xs:key>
136  <xs:key name="foglalas_PK">
137    <xs:selector xpath="foglalasok/foglalas"/>
138    <xs:field xpath="@foglalas_ID"/>
139  </xs:key>
140  <xs:key name="vendeg_PK">

```

```

141     <xs:selector xpath="vendegek/vendeg"/>
142     <xs:field xpath="@vendeg_ID"/>
143 </xs:key>
144
145Ⓜ <xs:keyref name="t_Lakas_FK" refer="Lakas_PK">
146     <xs:selector xpath="takaritasok/takaritas"/>
147     <xs:field xpath="@Lakas_IDREF"/>
148 </xs:keyref>
149Ⓜ <xs:keyref name="t_takarito_FK" refer="takarito_PK">
150     <xs:selector xpath="takaritasok/takaritas"/>
151     <xs:field xpath="@takarito_IDREF"/>
152 </xs:keyref>
153Ⓜ <xs:keyref name="f_Lakas_FK" refer="Lakas_PK">
154     <xs:selector xpath="foglalasok/foglalas"/>
155     <xs:field xpath="@Lakas_IDREF"/>
156 </xs:keyref>
157Ⓜ <xs:keyref name="f_takarito_FK" refer="takarito_PK">
158     <xs:selector xpath="foglalasok/foglalas"/>
159     <xs:field xpath="@takarito_IDREF"/>
160 </xs:keyref>
161Ⓜ <xs:keyref name="vendeg_FK" refer="vendeg_PK">
162     <xs:selector xpath="foglalasok/foglalas"/>
163     <xs:field xpath="@vendeg_IDREF"/>
164 </xs:keyref>
165 </xs:element>
166 </xs:schema>

```

## 2. feladat

### 2a, Read fájl

```

1 package hu.domparse.gie0ej;
2
3Ⓜ import java.io.File;
16
17
18 public class DOMReadGIE0EJ {
19
20Ⓜ     public static void main(String[] args) throws ParserConfigurationException, SAXException, IOException {
21
22
23         DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
24         DocumentBuilder dBuilder = factory.newDocumentBuilder();
25
26
27         File xmlFile = new File("src/XMLGIE0EJ.xml");
28         Document doc = dBuilder.parse(xmlFile);
29
30         doc.getDocumentElement().normalize();
31
32         System.out.println("Root elem: " + doc.getDocumentElement().getNodeName());
33
34
35         NodeList nList = doc.getElementsByTagName("lakas");
36
37         for (int i = 0; i < nList.getLength(); i++) {
38
39
40             Node nNode = nList.item(i);
41             System.out.println("\nKiválasztott elem: " + nNode.getNodeName());
42

```

```

42
43
44     if (nNode.getNodeType() == Node.ELEMENT_NODE) {
45         Element elem = (Element) nNode;
46
47         String lakas_ID = elem.getAttribute("lakas_ID");
48
49         Node n1 = elem.getElementsByTagName("ir_szam").item(0);
50         String ir_szam = n1.getTextContent();
51
52         Node n2 = elem.getElementsByTagName("telepules").item(0);
53         String telepules = n2.getTextContent();
54
55         Node n3 = elem.getElementsByTagName("utca_hazszam").item(0);
56         String hazszam = n3.getTextContent();
57
58         Node n4 = elem.getElementsByTagName("agynemu").item(0);
59         String agynemu = n4.getTextContent();
60
61         Node n5 = elem.getElementsByTagName("kapukod").item(0);
62         String kapukod = n5.getTextContent();
63
64         System.out.println("Lakás azonosító: " + lakas_ID);
65         System.out.println("Kapukód: " + kapukod);
66         System.out.println("Irányítószám: " + ir_szam);
67         System.out.println("Település: " + telepules);
68         System.out.println("Utca és Házszám: " + hazszam);
69         System.out.println("Ágynemű: " + agynemu);
70
71     }
72 }
73
74 nList = doc.getElementsByTagName("takaritas");
75 for (int i = 0; i < nList.getLength(); i++) {
76
77     Node nNode = nList.item(i);
78     System.out.println("\nKiválasztott elem: " + nNode.getNodeName());
79
80     if (nNode.getNodeType() == Node.ELEMENT_NODE) {
81         Element elem = (Element) nNode;
82
83         String lakas_IDREF = elem.getAttribute("lakas_IDREF");
84         String takarito_IDREF = elem.getAttribute("takarito_IDREF");
85
86         Node n1 = elem.getElementsByTagName("idopont").item(0);
87         String idopont = n1.getTextContent();
88
89         System.out.println("Lakás azonosító: " + lakas_IDREF);
90         System.out.println("Idopont: " + idopont);
91         System.out.println("Takarító azonosító: " + takarito_IDREF);
92
93     }
94 }
95
96 nList = doc.getElementsByTagName("takarito");
97 for (int i = 0; i < nList.getLength(); i++) {
98
99     for (int i = 0; i < nList.getLength(); i++) {
100
101         Node nNode = nList.item(i);
102         System.out.println("\nKiválasztott elem: " + nNode.getNodeName());
103
104         if (nNode.getNodeType() == Node.ELEMENT_NODE) {
105             Element elem = (Element) nNode;
106
107             String takarito_ID = elem.getAttribute("takarito_ID");
108
109             Node n1 = elem.getElementsByTagName("nev").item(0);
110             String nev = n1.getTextContent();
111             Node n2 = elem.getElementsByTagName("szem_ig_szam").item(0);
112             String szem_ig_szam = n2.getTextContent();
113             Node n3 = elem.getElementsByTagName("mobilszam").item(0);
114             String mobszszam = n3.getTextContent();
115
116             System.out.println("Takarító azonosító: " + takarito_ID);
117             System.out.println("Takarító neve: " + nev);
118             System.out.println("Személyi igazolvány szám: " + szem_ig_szam);
119             System.out.println("Móbi száma: " + mobszszam);
120
121         }
122
123     }
124
125     nList = doc.getElementsByTagName("vendeg");
126     for (int i = 0; i < nList.getLength(); i++) {
127
128         Node nNode = nList.item(i);
129         System.out.println("\nKiválasztott elem: " + nNode.getNodeName());
130
131     }
132 }

```

```

125     System.out.println("\nKiválasztott elem: " + nNode.getNodeName());
126
127     if (nNode.getNodeType() == Node.ELEMENT_NODE) {
128         Element elem = (Element) nNode;
129
130         String vendeg_ID = elem.getAttribute("vendeg_ID");
131
132         Node n1 = elem.getElementsByTagName("nev").item(0);
133         String nev = n1.getTextContent();
134
135         Node n2 = elem.getElementsByTagName("telefonszam").item(0);
136         String telefonszam = n2.getTextContent();
137
138         Node n3 = elem.getElementsByTagName("utlevel_sz").item(0);
139         String utlevel_sz = n3.getTextContent();
140
141         System.out.println("Vendég azonosító: " + vendeg_ID);
142         System.out.println("Vendég neve: " + nev);
143         System.out.println("Telefonszám: " + telefonszam);
144         System.out.println("Útlevélszám: " + utlevel_sz);
145     }
146 }
147
148 nList = doc.getElementsByTagName("foglalas");
149 for (int i = 0; i < nList.getLength(); i++) {
150
151     Node nNode = nList.item(i);
152     System.out.println("\nKiválasztott elem: " + nNode.getNodeName());
153
154
155     if (nNode.getNodeType() == Node.ELEMENT_NODE) {
156         Element elem = (Element) nNode;
157
158         String foglalas_ID = elem.getAttribute("foglalas_ID");
159         String lakas_IDREF = elem.getAttribute("lakas_IDREF");
160         String takarito_IDREF = elem.getAttribute("takarito_IDREF");
161         String vendeg_IDREF = elem.getAttribute("vendeg_IDREF");
162
163
164         Node n1 = elem.getElementsByTagName("k_datum").item(0);
165         String k_datum = n1.getTextContent();
166
167         Node n2 = elem.getElementsByTagName("z_datum").item(0);
168         String z_datum = n2.getTextContent();
169
170         Node n3 = elem.getElementsByTagName("fo").item(0);
171         String fo = n3.getTextContent();
172
173         System.out.println("Foglalás azonosító: " + foglalas_ID);
174         System.out.println("Lakás azonosító: " + lakas_IDREF);
175         System.out.println("Takarító azonosító: " + takarito_IDREF);
176         System.out.println("Kezdő dátum: " + k_datum);
177         System.out.println("Záró dátum: " + z_datum);
178         System.out.println("Létszám: " + fo);
179         System.out.println("Vendég azonosító: " + vendeg_IDREF);
180     }
181 }
182 }
183 }

```

## Output:

Root elem: airbnblakasoktakarítása

Kiválasztott elem: lakas

Lakás azonosító: 1

Kapukód: 2345

Irányítószám: 1065

Település: Budapest

Utca és Házszám: Révay utca 8

Ágynemű: szimpla

Kiválasztott elem: lakas

Lakás azonosító: 2

Kapukód: 3456

Irányítószám: 1061

Település: Budapest

Utca és Házszám: Andrássy út 23

Ágynemű: dupla

Kiválasztott elem: lakas

Lakás azonosító: 3

Kapukód: 4585

Irányítószám: 1051

Település: Budapest

Utca és Házszám: Zrínyi utca 4

Ágynemű: szimpla

## 2b, Query fájl

1. lekérdezés: lekérdezi a dupla ágyneművel rendelkező lakások adatait
2. lekérdezés: lekérdezi a 2021-12-04-én takarítandó lakás takarítójának adatait és a várost, ahol a lakás található.

```

1 package hu.dompars.gie0ej;
2
3 import java.io.File;
4 import java.io.IOException;
5 import java.text.ParseException;
6
7 import javax.xml.parsers.DocumentBuilder;
8 import javax.xml.parsers.DocumentBuilderFactory;
9 import javax.xml.parsers.ParserConfigurationException;
10
11 import org.w3c.dom.Document;
12 import org.w3c.dom.Element;
13 import org.w3c.dom.Node;
14 import org.w3c.dom.NodeList;
15 import org.xml.sax.SAXException;
16
17 public class DOMQueryGIE0EJ {
18
19     public static void main(String[] args) throws ParserConfigurationException, SAXException, IOException, ParseException {
20
21         DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
22         DocumentBuilder dBuilder = factory.newDocumentBuilder();
23
24         File xmlFile = new File("src/XMLGIE0EJ.xml");
25         Document doc = dBuilder.parse(xmlFile);
26
27         doc.getDocumentElement().normalize();
28
29         System.out.println("Root element: " + doc.getDocumentElement().getNodeName());
30
31
32         System.out.println("\nDupla ágyneművel rendelkező lakás adatai:");
33
34         NodeList lakaslist = doc.getElementsByTagName("lakas");
35
36         for (int i = 0; i < lakaslist.getLength(); i++) {
37
38             Node nNode = lakaslist.item(i);
39
40             if (nNode.getNodeType() == Node.ELEMENT_NODE) {
41                 Element elem = (Element) nNode;
42
43                 String lakas_ID = elem.getAttribute("lakas_ID");
44
45                 Node n1 = elem.getElementsByTagName("kapukod").item(0);
46                 String kapukod = n1.getTextContent();
47
48                 Node n2 = elem.getElementsByTagName("ir_szam").item(0);
49                 String ir_szam = n2.getTextContent();
50
51                 Node n3 = elem.getElementsByTagName("telepules").item(0);
52                 String telepules = n3.getTextContent();
53
54                 Node n4 = elem.getElementsByTagName("utca_hazszam").item(0);
55                 String utca_hazszam = n4.getTextContent();
56
57                 Node n5 = elem.getElementsByTagName("agynemu").item(0);
58                 String agynemu = n5.getTextContent();

```

```

58
59         if(agynemu.equals("dupla")) {
60
61             System.out.println("Lakás azonosítója: " + lakas_ID);
62             System.out.println("Kapukód: " + kapukod);
63             System.out.println("Irányítószám: " + ir_szam);
64             System.out.println("Település: " + telepules);
65             System.out.println("Utca és Házszám: " + utca_hazszam);
66             System.out.println("Ágynemű: " + agynemu);
67         }
68     }
69 }
70
71 System.out.println("\n2021-12-04-én kiírt takarítást végző dolgozó adatai és a kitakarítandó lakás települése");
72
73 NodeList takaritolist = doc.getElementsByTagName("takarito");
74 NodeList takaritaslist = doc.getElementsByTagName("takaritas");
75
76 for (int i = 0; i < takaritolist.getLength(); i++) {
77
78     Node nNode = takaritolist.item(i);
79
80     if (nNode.getNodeType() == Node.ELEMENT_NODE) {
81         Element elem = (Element) nNode;
82
83         String takarito_ID = elem.getAttribute("takarito_ID");
84
85
86         Node n1 = elem.getElementsByTagName("nev").item(0);
87
88         Node n1 = elem.getElementsByTagName("nev").item(0);
89         String nev = n1.getTextContent();
90
91         Node n2 = elem.getElementsByTagName("szem_ig_szam").item(0);
92         String szem_ig_szam = n2.getTextContent();
93
94         Node n3 = elem.getElementsByTagName("mobilszam").item(0);
95         String mobilszam = n3.getTextContent();
96
97         for (int j = 0; j < takaritaslist.getLength(); j++) {
98
99             Node nNode2 = takaritaslist.item(j);
100
101             if (nNode2.getNodeType() == Node.ELEMENT_NODE) {
102                 Element elem2 = (Element) nNode2;
103
104                 String lakas_IDREF = elem2.getAttribute("lakas_IDREF");
105                 String takarito_IDREF = elem2.getAttribute("takarito_IDREF");
106
107                 Node n4 = elem2.getElementsByTagName("idopont").item(0);
108                 String idopont = n4.getTextContent();
109
110                 if(takarito_IDREF.equals(takarito_ID) && idopont.equals("2021-12-04")) {
111
112                     for (int k = 0; k < lakaslist.getLength(); k++) {
113
114                         Node nNode3 = lakaslist.item(k);
115
116                         if (nNode2.getNodeType() == Node.ELEMENT_NODE) {
117
118                             if (nNode2.getNodeType() == Node.ELEMENT_NODE) {
119                                 Element elem3 = (Element) nNode3;
120
121                                 String lakas_ID = elem3.getAttribute("lakas_ID");
122
123                                 Node n5 = elem3.getElementsByTagName("telepules").item(0);
124                                 String telepules = n5.getTextContent();
125
126                                 if(lakas_ID.equals(lakas_IDREF) ) {
127
128                                     System.out.println("Takarító azonosító: " + takarito_ID);
129                                     System.out.println("Takarító neve: " + nev);
130                                     System.out.println("Személyigazolvány szám: " + szem_ig_szam);
131                                     System.out.println("Mobilszám: " + mobilszam);
132                                     System.out.println("Település: " + telepules);
133                                 }
134                             }
135                         }
136                     }
137                 }
138             }
139 }

```

## Output:

Root element: airbnblakasoktakaritasa

Dupla ágyneművel rendelkező lakás adatai:

Lakás azonosítója: 2

Kapukód: 3456

Irányítószám: 1061

Település: Budapest

Utca és Házszám: Andrássy út 23

Ágynemű: dupla

2021-12-04-én kiírt takarítást végző dolgozó adatai és a kitakarítandó lakás települése

Takarító azonosító: 2

Takarító neve: Nagy Katalin

Személyigazolvány szám: 463782EE

Mobilszám: 06204563998

Település: Budapest

## 2c, Modify fájl

Módosítja Nagy Katalin takarító telefonszámát.

```
1 package hu.domparse.gie0ej;
2
3 import java.io.File;
4
5 public class DOMModifyGIE0EJ {
6
7     public static void main(String[] args) {
8
9         try {
10             File inputFile = new File("src/XMLGIE0EJ.xml");
11             DocumentBuilderFactory docFactory = DocumentBuilderFactory.newInstance();
12             DocumentBuilder docBuilder = docFactory.newDocumentBuilder();
13             Document doc = docBuilder.parse(inputFile);
14
15             // Nagy Katalin telefonszolgálatot váltott és új lett a mobilszáma
16             NodeList takaritoList = doc.getElementsByTagName("takarito");
17             for (int i = 0; i < takaritoList.getLength(); i++) {
18
19                 Node nNode1 = takaritoList.item(i);
20
21                 if (nNode1.getNodeType() == Node.ELEMENT_NODE) {
22                     Element elem = (Element) nNode1;
23
24                     Node n1 = elem.getElementsByTagName("nev").item(0);
25                     String nev = n1.getTextContent();
26
27                     if (nev.equals("Nagy Katalin")) {
28                         NodeList childNodes = nNode1.getChildNodes();
```



```

45
46         NodeList childNodes = nNode1.getChildNodes();
47         for (int j = 0; j < childNodes.getLength(); j++) {
48             Node childNode = childNodes.item(j);
49             if (childNode.getNodeName().equals("mobilszam")) {
50                 childNode.setTextContent("06303333333");
51             }
52         }
53     }
54 }
55 }
56 }
57 }
58 }
59 }
60
61     modify(doc);
62
63 } catch (Exception e) {
64     e.printStackTrace();
65 }
66
67
68 }
69
70 private static void modify(Document doc) throws TransformerException {
71     TransformerFactory transformerFactory = TransformerFactory.newInstance();
72     Transformer transformer = transformerFactory.newTransformer();
73     System.out.println("-Modified File-");
74     transformer.setOutputProperty(OutputKeys.ENCODING, "UTF-8");
75
76 private static void modify(Document doc) throws TransformerException {
77     TransformerFactory transformerFactory = TransformerFactory.newInstance();
78     Transformer transformer = transformerFactory.newTransformer();
79     System.out.println("-Modified File-");
80     transformer.setOutputProperty(OutputKeys.ENCODING, "UTF-8");
81     transformer.setOutputProperty(OutputKeys.INDENT, "yes");
82     transformer.setOutputProperty("{http://xml.apache.org/xslt}indent-amunt", "2");
83
84     DOMSource source = new DOMSource(doc);
85
86     StreamResult console = new StreamResult(System.out);
87
88     transformer.transform(source, console);
89 }
90 }

```

## Output:

```

<takarito takarito_ID="2">

    <nev>Nagy Katalin</nev>

    <szem_ig_szam>463782EE</szem_ig_szam>

    <mobilszam>06303333333</mobilszam>

</takarito>

```