ЕДомашно бр. 1
Jena RDF API

А) Прашања

1. Како се изразува еден запис (еден факт) во RDF моделот?

Еден запис односно една тројка се изразува со субјект-предикат- објект структура.

2. Кои различни синтакси за RDF моделот постојат? Изразете го следниот факт во неколку различни RDF синтакси: „ВБС се предава на ФИНКИ“. Користете го префиксот @prefix finki: <http://finki.ukim.mk/resource#> за URI вредностите на ентитетите и релацијата.

Turtle:

```
@prefix finki: <http://finki.ukim.mk/resource#> .
finki:VNP finki:sePredavaNa finki:FINKI.
```

RDF/XML:

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:finki="http://finki.ukim.mk/resource#">

<rdf:Description rdf:about="http://finki.ukim.mk/resource#VBS">
    <finki: sePredavaNa rdf:resource="http://finki.ukim.mk/resource#FINKI"/>
</rdf:Description>

</rdf:RDF>
```

RDFa:

```
<html xmlns:finki="http://finki.ukim.mk/resource#">
  <div finki:about="finki:VBS">
    VBS is studied at
    <span finki:rel="finki:isStudiedAt" finki:resource="finki:FINKI">
      FINKI
    </span>
  </div>
</html>
```

JSON-LD:

```
{
  "@context": {
    "finki": "http://finki.ukim.mk/resource#"
  },
  "@id": "finki:VBS",
  "finki:isStudiedAt": "finki:FINKI"
}
```

3. За што се користи RDF Schema?

Јазик за опис на вокабулар. Се користи за опишување на групи од поврзани ресурси и односите помеѓу нив.

4. Дефинирајте RDFS класи за „факултет“ и „предмет“, како и една релација која ги поврзува нив, „е предмет на“. Користете го префиксот од 2. за нивните URI вредности. Користете Turtle синтакса.

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix finki: <http://finki.ukim.mk/resource#> .

finki:Faculty a rdfs:Class ;
   rdfs:label "Faculty" ;
   rdfs:comment "The class of faculties" .

finki:Subject a rdfs:Class ;
   rdfs:label "Subject" ;
   rdfs:comment "The class of subjects" ;
   rdfs:subClassOf finki:Faculty .

finki:isSubjectAt a rdf:Property ;
   rdfs:label "is a subject at" ;
   rdfs:comment "Relates a subject to a faculty" ;
   rdfs:domain finki:Faculty ;
   rdfs:range finki:Subject .
```

Б) Практична задача

I. Креирање едноставен RDF граф

1. Креирајте нов Java проект во IDE по ваш избор. Вклучете ги во проектот сите .jar библиотеки од lib фолдерот од Jena. Jena преземете ја директно од Jena сајтот. ✓

2. Во main() методот на главната класа од проектот, креирајте основен Jena model, кој ќе го содржи RDF графот кој треба да го изградите во текот на вежбата. ✓

3. Во моделот додадете нов ресурс, кој ќе ве репрезентира вас како личност. Како URI на ресурсот искористете URL адреса од некој ваш социјален профил (Facebook, Twitter, Instagram, TikTok, …), кој уникатно ве идентификува. ✓

4. Додадете својство на вашиот ресурс, кое ќе го репрезентира вашето целосно име. Искористете го својството 'vcard:fn'. ✓

5. Додадете уште неколку својства по избор, кои ќе бидат од истата 'vcard' или пак од 'foaf' RDF шемата. Во моделот треба да имате минимум 10 RDF тројки. Притоа, внимавајте на тоа дали range вредноста на својството кое го додавате треба да биде литерал или друг објект. ✓

II. Печатење на RDF граф

6. Со користење на model.listStatements() методот на моделот, изминете ги сите RDF записи (тројки) од графот и отпечатете ги во формат: "subject – predicate – object". При печатењето, литералите отпечатете ги во наводници (""). Печатењето нека биде во конзола, т.е. преку System.out.

   Напомена: Пред да ги отпечатите RDF тројките, напишете на конзола "Printing with model.listStatements():".✓

7. Извршете ја програмата. Може да го извршите целиот проект или само класата во која го дефинирвте кодот до сега. Проверете дали излезот на конзола соодветствува со она што очекувате да се прикаже. Дали сите RDF тројки кои ги дефинирвте во кодот, ги гледате отпечатени? ✓

8. Без да го бришете претходното печатење, додадете ново печатење на RDF тројките од моделот, со користење на model.write() методот. Притоа направете повеќе печатења, во следните RDF формати: RDF/XML, Pretty RDF/XML, N-Triples и Turtle.

   Напомена: Пред секое од печатењата, напишете на конзола "Printing with model.print(), in *Turtle*.", во зависност од конкретниот формат. ✓

9. Извршете ја програмата. Може да го извршите целиот проект или само класата во која го дефинирате кодот до сега. Проверете дали излезот на конзола соодветствува со она што очекувате да се прикаже.
   Кој од RDF форматите има најкратка (најкомпактна) содржина?
   - N-Triples е најкомпактниот формат,

   Кој најлесно се „чита“ на прв поглед?
   - Pretty RDF/XML и Turtle се најлесни за читање

   Кој, пак, сметате дека најлесно би го испроцесирале во код, доколку го прочитате програмски од некаде?
   - Turtle

```
import org.apache.jena.rdf.model.*;
import org.apache.jena.vocabulary.VCARD;

public class RDFGraph {
    public static void main(String[] args) {
        String personURI = "http://www.linkedin.com/in/monikajovevska";
        String givenName = "Monika";
        String familyName = "Jovevska";
        String fullName = givenName+" "+familyName;
        String City = "Pehcevo";
        String Faculty = "FINKI";
        String Subject = "VBS";
        String Telephone = "070 807 917";

        Model model = ModelFactory.createDefaultModel();

        Resource MonikaJovevska =
                model.createResource(personURI)
                        .addProperty(VCARD.FN, fullName)
                        .addProperty(VCARD.N, model.createResource()
                                .addProperty(VCARD.Given, givenName)
                                .addProperty(VCARD.Family, familyName))
                        .addProperty(VCARD.Locality, City)
                        .addProperty(VCARD.Pcode,
model.createResource()
                                .addProperty(VCARD.Pcode, "2326"))
                        .addProperty(VCARD.ORG, Faculty)
                        .addProperty(VCARD.ROLE, Subject)
                        .addProperty(VCARD.TEL, Telephone);

        StmtIterator iter = model.listStatements();
        System.out.println("Printing with model.listStatements():");
        while (iter.hasNext()){
            Statement stmt = iter.nextStatement();
            Resource subject = stmt.getSubject();
            Property predicate = stmt.getPredicate();
            RDFNode object = stmt.getObject();
            //subject - predicate - object
            if(object instanceof Resource) {
                System.out.print(subject.toString()+" -
"+object.toString()+" - "+predicate.toString());
            } else {
                //object is a literal
                System.out.print(" \"" + subject.toString()+" -
"+object.toString()+" - "+predicate.toString() + " \"");
            }
            System.out.println(" .");
        }
        System.out.println("\nPrinting with model.write(), in
```

```
RDF/XML:");
        model.write(System.out);

        System.out.println("\nPrinting with model.write(), in Pretty
RDF/XML:");
        model.write(System.out, "RDF/XML-ABBREV");

        System.out.println("\nPrinting with model.write(), in N-
Triples:");
        model.write(System.out, "N-TRIPLES");

        System.out.println("\nPrinting with model.write(), in
Turtle:");
        model.write(System.out, "TURTLE");
    }
}
```

III. Читање на RDF граф

10. Креирајте нова Java класа во проектот, во која ќе додадете и main() метод. ✓

11. Ископирајте еден од излезите од претходните задачи (вашиот RDF граф во некоја од RDF синтаксите) и ставете го во текстуален фајл, кој ќе го снимите локално, под произволно име и соодветна наставка: .xml за RDF/XML и Pretty RDF/XML, .ttl за Turtle, .nt за N-Triples и n3 за N3.

12. Во main() методот на новата класа креирајте нов модел и со користење на model.read() вчитајте го RDF графот од датотеката креирана во претходниот чекор.

    Напомена: Искористете го третиот параметар на model.read() кој го означува RDF форматот на датотеката која ја читате – има исти вредности како model.write() при запишување, односно “RDF/XML”, “RDF/XML-ABBREV”, “TTL”, “N-TRIPLES”, итн. ✓

13. Напишете код за печатење на моделот (графот), за да видите дали успешно е прочитан. ✓

14. Извршете ја програмата. Може да го извршите целиот проект или само класата во која го дефинирвте кодот до сега. Проверете дали излезот на конзола соодветствува со она што очекувате да се прикаже. ✓

IV. Навигација низ RDF граф

15. Откако ќе го вчитате графот од датотека во претходниот дел од вежбата, додадете код кој ќе го селектира ресурсот од графот кој ве репрезентира вас.

16. Преку селектираниот ресурс, прочитајте ја вредноста на дел од релациите (целосно име, име, презиме, итн.), во зависност од тоа што сте креирале како RDF тројки на почетокот од вежбата.

    Напомена: Внимавајте како пристапувате до вредностите кои се ресурси, а како до вредностите кои се литерали. Постои ли разлика во начинот на пристап?

17. Извршете ја програмата. Може да го извршите целиот проект или само класата во која го дефинирате кодот до сега. Проверете дали излезот на конзола соодветствува со она што очекувате да се прикаже.

```
import org.apache.jena.rdf.model.Model;
import org.apache.jena.rdf.model.ModelFactory;
import org.apache.jena.rdf.model.Resource;
import org.apache.jena.util.FileManager;
import org.apache.jena.vocabulary.VCARD;

import java.io.InputStream;

public class RDFGraphReader {
    public static void main(String[] args) {

        Model model = ModelFactory.createDefaultModel();

        String filename = "C:\\Users\\Korisnik\\OneDrive - UKIM,
FINKI\\Desktop\\VBS domasno 1\\RDF_XML_Graph.xml"; // Патот до вашиот фајл
        InputStream in = FileManager.get().open(filename);
        if(in == null) {
            throw new IllegalArgumentException("File: " + filename + " not
found");
        }
        model.read(in, "");

        model.write(System.out, "TTL");

        String personURI = "http://www.linkedin.com/in/monikajovevska";
        Resource MonikaVcard = model.getResource(personURI);

        Resource given = MonikaVcard.getProperty(VCARD.N).getResource();
        System.out.println("Given Name: " + given);

        String fullName = MonikaVcard.getProperty(VCARD.FN).getString();
        System.out.println("Full Name: " + fullName);

        Resource PostCode =
MonikaVcard.getProperty(VCARD.Pcode).getResource();
        System.out.println("Post Code: " + PostCode);

        String city = MonikaVcard.getProperty(VCARD.Locality).getString();
        System.out.println("City: " + city);

        String faculty = MonikaVcard.getProperty(VCARD.ORG).getString();
        System.out.println("Faculty: " + faculty);

        String subject = MonikaVcard.getProperty(VCARD.ROLE).getString();
        System.out.println("Subject: " + subject);

        String telephone = MonikaVcard.getProperty(VCARD.TEL).getString();
        System.out.println("Telephone: " + telephone);
    }
}
```

V. Извлекување податоци од RDF граф

18. Креирајте нова Java класа во проектот, во која ќе додадете и main() метод.

19. Преземете ја датотеката “hifm-dataset.ttl” од Courses и снимете ја локално.

20. Во main() методот на новата класа напишете код со кој ќе ја прочитате содржината на оваа датотека. Внимавајте третиот параметар на model.read() да го поставите за вчитување на Turtle содржина.

```
Model model = ModelFactory.createDefaultModel();

String filename = "C:\\Users\\Korisnik\\OneDrive - UKIM, FINKI\\Desktop\\VBS
domasno 1\\hifm-dataset.ttl";
model.read(filename, "TURTLE");
```

21. Проучете ја содржината на “hifm-dataset.ttl” датотеката. Станува збор за податочно множество кое содржи лекови од Фондот за здравство на РМ. За секој од лековите имаме тип (hifm-ont:Drug и drugbank:drugs), име (rdfs:label, drugbank:brandName и drugbank:genericName), цена (hifm-ont:refPriceWithVAT), релации кон други локални (hifm-ont:similarTo) и светски лекови (rdfs:seeAlso), итн.

22. Врз база на научената од досегашниот тек на вежбата, излистајте ги имињата на сите лекови кои се наоѓаат во графот (моделот) (една од трите релации за име е доволна), по азбучен редослед.

```
System.out.println("List of drug names in alphabetical order:");
ResIterator drugIterator = model.listSubjectsWithProperty(RDFS.label);
drugIterator.toList().stream()
        .map(drug -> drug.getProperty(RDFS.label).getString())
        .sorted()
        .forEach(System.out::println);
```

23. Одберете еден лек од графот (моделот) и за него излистајте ги сите релации и вредности.

```
String selectedDrugURI = "http://purl.org/net/hifm/data#36897";
Resource selectedDrug = model.getResource(selectedDrugURI);
System.out.println("\nSelected Drug: " +
selectedDrug.getRequiredProperty(RDFS.label).getString());
selectedDrug.listProperties().toList().forEach(System.out::println);
```

24. Одберете еден лек од графот (моделот) и за него излистајте ги имињата на сите лекови кои имаат иста функција како и тој, т.е. лекови со кои тој е во релација ‘hifm-ont:similarTo’. Форматирајте го печатењето за да е јасно видливо за што станува збор.

```
System.out.println("\nSimilar drugs to the selected drug:");
selectedDrug.listProperties(model.getProperty("http://purl.org/net/hifm/ontol
ogy#similarTo")).toList().stream()
        .map(stmt ->
stmt.getObject().asResource().getRequiredProperty(RDFS.label).getString())
        .forEach(System.out::println);
```

25. Одберете еден лек од графот (моделот) и за него најпрвин излистајте ја неговата цена (hifm-ont:refPriceWithVAT), а потоа излистајте ги и имињата и цените на лековите кои

ја имаат истата функција како и тој (hifm-ont:similarTo). Форматирајте го печатењето за да е јасно видливо за што станува збор.

```
System.out.println("\nPrices of the selected drug and similar drugs:");
System.out.println("Selected Drug: " +
selectedDrug.getProperty(model.getProperty("http://purl.org/net/hifm/ontology
#refPriceNoVAT")).getFloat());
selectedDrug.listProperties(model.getProperty("http://purl.org/net/hifm/ontol
ogy#similarTo")).toList().forEach(stmt -> {
    Resource similarDrug = stmt.getObject().asResource();
    System.out.print(similarDrug.getRequiredProperty(RDFS.label).getString()
+ ": ");

System.out.println(similarDrug.getProperty(model.getProperty("http://purl.org
/net/hifm/ontology#refPriceNoVAT")).getFloat());
});
```

26. Извршете ја програмата. Може да го извршите целиот проект или само класата во која го дефинирате кодот до сега. Проверете дали излезот на конзола соодветствува со она што очекувате да се прикаже.

Напомена: Доколку успеавте да ги завршите задачите под точка 22, 23, 24 и 25, практично напишавте код кој може да биде основа за една мобилна, веб или десктоп апликација за лекови: на корисникот му се претставуваат сите лекови (22), може да одбере некој од нив и да му се отвори приказ со сите детали за лекот (23), да ги види алтернативните лекови со иста функција кои може да ги купи наместо селектираниот (24) и да ги спореди нивните цени (25) со цел да го избере најевтиниот од таа група лекови со исто дејство.