MANDATORY EXERCISE no. 3

Delivery files: `query.java, exercise_1.rq, ..., exercise_7.rq, and an output.txt` (for each query you run).

It is wise to read through the whole exercise text first.

1 Query engine

In this exercise you are asked to make a SPARQL query engine.

1.1 Exercise

Write a java program which reads an RDF graph and a SPARQL query from file, queries the graph and outputs the query results as a table. Your program should accept `SELECT` queries, `CONSTRUCT` queries and `ASK` queries. Messages should be given if the query is of a different type.

1.1.1 Tip

If I query the Simpsons RDF graph (`simpsons.ttl`) we wrote in a previous exercise with my SPARQL query engine and the `SELECT` query

```
1 PREFIX sim: <http://www.ifi.uio.no/INF3580/simpsons#>
2 PREFIX fam: <http://www.ifi.uio.no/INF3580/family#>
```

```
3 PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
4 PREFIX foaf: <http://xmlns.com/foaf/0.1/>
5 SELECT ?s ?o
6 WHERE{ ?s foaf:age ?o } 7 LIMIT 1
```

I get[1] the following: (To get the nicely formatted output I use the class ResultSetFormatter.)

```
-------------------------------------------------------------------
| s                                                 | o           |
===================================================================
| <http://www.ifi.uio.no/INF3580/simpsons#Maggie> | "1"^^xsd:int |
-------------------------------------------------------------------
```

Executing with the `ASK` query

```
1 ASK{ ?s ?p ?o }
```

gives me

```
true
```

Executing with the `CONSTRUCT` query

```
1 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
2 PREFIX fam: <http://www.ifi.uio.no/INF3580/family#>
3 PREFIX sim: <http://www.ifi.uio.no/INF3580/simpsons#>
4 PREFIX foaf: <http://xmlns.com/foaf/0.1/>
5 CONSTRUCT{ sim:Bart rdfs:label ?name }
6 WHERE{ sim:Bart foaf:name ?name }
```

gives me

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix sim: <http://www.ifi.uio.no/INF3580/simpsons#> .
@prefix fam: <http://www.ifi.uio.no/INF3580/family#> .
sim:Bart rdfs:label "Bart Simpsons" .
```

1.2 Exercise

In the following exercises, we will use the program from the previous question to write SPARQL queries to retrieve data from the Simpsons RDF graph (`simpsons.ttl`). For each of the exercises below write a SPARQL query

[1] Note that your results may be different according to how your Simpsons RDF file looks like.

whose results answer the question in the exercise.

Each exercise which requires a `SELECT` query specifies the result variable names you must use, i.e., the variable names that shall follow the `SELECT` word. We require this in order to easier be able to test your delivery.

See W3C's SPARQL 1.1 Query Language[2] for definitions and examples. The SPARQLer Validator[3] might also come in handy.

Exercise 1

Query: Find all Persons and order them by identifier, list also optionally their name. Use the result variable names `?person, ?name`.

Tip: This is a simple query, where you will need to use `SELECT`, `WHERE`, `OPTIONAL` and `ORDER BY`. It is also important to set namespaces correctly in order for the query to work. If you do not get any results, try the query

```
SELECT ?s ?p ?o
WHERE {?s ?p ?o}
```

and see if your get the expected results and namespaces. This query lists all triples in the graph.

Mr. Oblig should also be helpful in weeding out typos and similar simple mistakes.

Exercise 2

Query: Find everyone who has a mother or a father and list both the person and the mother or father. Order by mother/father. Use the result variable name `?person, ?parent`.

Exercise 3

Query: Find everyone with a name with 'M' as first letter. Result variable name: `?person`.

---

[2] https://www.w3.org/TR/rdf-sparql-query/

[3] http://sparql.org/query-validator.html

Exercise 4

Query: Find all of Maggie's grandmothers. Result variable name: `?grandmother`.

Exercise 5

Query: Find everyone older than 10. Order by age, oldest first. Output name and age. Result variable names: `?person, ?age`.

Exercise 6

Query: Is Herb the brother of Homer?

Tip: Use `ASK`.

Exercise 7

Write a `CONSTRUCT` SPARQL query that produces a FOAF file for Homer, adding his name, a `foaf:knows` relationship to his spouse, and the name of his spouse.

Ending notes

Delivery

You shall deliver one file per exercise. Name the files `exercise_X.rq`, where `X` is the exercise number.

Mr. Oblig

You can, and should, use Mr. Oblig to test your delivery before handing it in. Mr. Oblig is located at `http://sws.ifi.uio.no/mroblig/`. Note that Mr. Oblig comes without any warranty. A non-functioning Mr. Oblig will not have any effect on the due date of this mandatory exercise, and a flawless test report from Mr. Oblig does not guarantee that your delivery will be graded with passed. Also, since there may be many correct answers to an exercise, it

is possible that Mr. Oblig does not agree with your solution even though it is correct. However, a perfect score from Mr. Oblig is an indication that your delivery is good and that it does not contain any "stupid" errors.

Good Luck!