E03 The Resource Description Framework

Quiz

Decide whether the following statements are true or false.

Assume the following prefix declarations:

```
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix : <#> .
```

Q3.1 The W3C RDF specification defines the triple data model and a vocabulary.

TRUE

Q3.2 Blank node labels are globally unique identifiers.

FALSE: Blank nodes are only unique within one RDF graph.

Q3.3 Web Blank nodes can be directly referenced from outside the document in which they are introduced.

FALSE: Blank nodes can only be referenced within one document.

Q3.4 On the subject position of an RDF triple, any RDF term is allowed.

FALSE: Literals are not allowed on the subject position of an RDF triple.

Q3.5 One must declare a `:rel a rdf:Property .` triple before using `:rel` as predicate in an RDF triple.

FALSE: One can use :rel as predicate in any RDF triple.

Q3.6 The following is a valid RDF triple:

```
:Alice [ a :FamilyRelationship ] :Bob .
```

FALSE: Blank nodes are not allowed on the predicate position.

Q3.7 Suitable RDF parsers can transform RDF documents from one serialisation to another.

TRUE

Q3.8 Every valid N-Triples document is also a valid Turtle document.

TRUE

Q3.9 The following Turtle document:

```
<#s> <#p> ( "a" "b" ) .
```

can be also expressed as:

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
<#s> <#p> [ rdf:first "a" ; rdf:rest [ rdf:first "b" ; rdf:rest rdf:nil ] ] .
```

TRUE

Q3.10 A subgraph of an RDF graph is a subset of the RDF terms in the graph.

FALSE: It is a subset of the RDF triples in the graph.

Q3.11 We employ isomorphism to check whether two RDF graphs are equivalent.

TRUE

Q3.12 The merge of two RDF graphs is obtained by taking the union of the triples of the two graphs.

FALSE: When taking the union, shared blank nodes need to be made distinct.

Q3.13 Multiple RDF graphs can go into the default graph of an RDF dataset.

TRUE

Exercises

E3.1 Mark the syntactically correct RDF triples:

- `"s"  "p"  "o"  .` ✗
- `<#s>   <#p>  <#o>  .` √
- `_:s    _:p   _:o   .` ✗
- `"s"    <#p>  <#o>  .` ✗
- `_:s    <#p>  <#o>  .` √
- `_:s    <#p>  "o"   .` √

E3.2 Given is the following Turtle document containing an rdf:Seq representing a set of persons' names mentioned in Chapter 1 at http://example.org/bag.ttl:

```
@prefix : <#> .

:C01        :mentions       _:bn1 .
_:bn1       rdf:type        rdf:Seq ;
            rdf:_1  "Paul Otlet" ;
            rdf:_2  "Vannevar Bush" ;
            rdf:_3  "Doug Engelbart";
            rdf:_4  "Ted Nelson"  .
```

Create a new RDF graph representing the same information using an RDF List. Represent your graph both in Turtle serialization with as many abbreviations as possible (at http://example.org/list.ttl) and in N-Triples serialization (at http://example.org/list.nt).

Solution:(Turtle):

:C01 :mentions
("Paul Otlet" "Vannevar Bush" "Doug Engelbart" "Ted Nelson") .

Solution: (N-Triples):

```
<http://example.org/list.nt#C01> <http://example.org/list.nt#mentions> _:bn1 .
_:bn1 <http://www.w3.org/1999/02/22-rdf-syntax-ns#first> "Paul Otlet"^^<http://www.w3.org/2001/XMLSchema#string> .
_:bn1 <http://www.w3.org/1999/02/22-rdf-syntax-ns#rest> _:bn2 .
_:bn2 <http://www.w3.org/1999/02/22-rdf-syntax-ns#first> "Vannevar Bush"^^<http://www.w3.org/2001/XMLSchema#string> .
_:bn2 <http://www.w3.org/1999/02/22-rdf-syntax-ns#rest> _:bn3 .
_:bn3 <http://www.w3.org/1999/02/22-rdf-syntax-ns#first> "Doug Engelbart"^^<http://www.w3.org/2001/XMLSchema#string> .
_:bn3 <http://www.w3.org/1999/02/22-rdf-syntax-ns#rest> _:bn4 .
_:bn4 <http://www.w3.org/1999/02/22-rdf-syntax-ns#first> "Ted Nelson"^^<http://www.w3.org/2001/XMLSchema#string> .
_:bn4 <http://www.w3.org/1999/02/22-rdf-syntax-ns#rest> <http://www.w3.org/1999/02/22-rdf-syntax-ns#nil> .
```

E3.3 Model the following statements in RDF using reification:

- Alice thinks Pizza is healthy.
- Bob says Alice is of type Person.
- The graph at http://example.org/bag.ttl contains the statement <#C01> <#mentions> _:bn1 .

Solution:

```
:Alice :thinks [ a rdf:Statement ;
rdf:subject :Pizza ;
rdf:predicate :is ;
rdf:object "healthy" ] .

:Bob :says [ a rdf:Statement ;
rdf:subject :Alice ;
rdf:predicate rdf:type ;
rdf:object :Person ] .

@prefix bag : <http://example.org/bag.ttl#> .
<http://example.org/bag.ttl> :contains
[ a rdf:Statement ;
rdf:subject bag:C01 ;
rdf:predicate bag:mentions ;
rdf:object _:bn1 ] .
```

E3.4 Proof that the following two RDF graphs are isomorphic.

Graph A represented in document **a.ttl**:

```
@prefix : <http://example.org/doc.ttl#> .

:s        :p        _:bn1 .
_:bn1     :p2       "A", "B" .
```

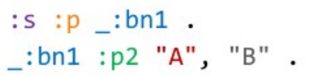
Graph B represented in document **b.ttl**:

```
@prefix ex: <http://example.org/doc.ttl#> .

_:bnode        ex:p2   "B" .

ex:s           ex:p           _:bnode .

_:bnode        ex:p2   "A" .
```

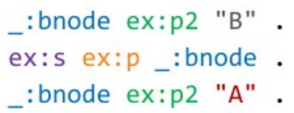
Solution:

Recap from lecture C03:

- Two RDF graphs are isomorphic if there is a bijection M between the two sets of nodes in the graphs G and G' such that:
  - M maps blank nodes to blank nodes.
  - $M(lit) = lit$ for all RDF literals lit which are nodes of G.
  - $M(iri) = iri$ for all IRIs iri which are nodes of G.
  - The triple (s, p, o) is in G if and only if the triple $(M(s), p, M(o))$ is in G'

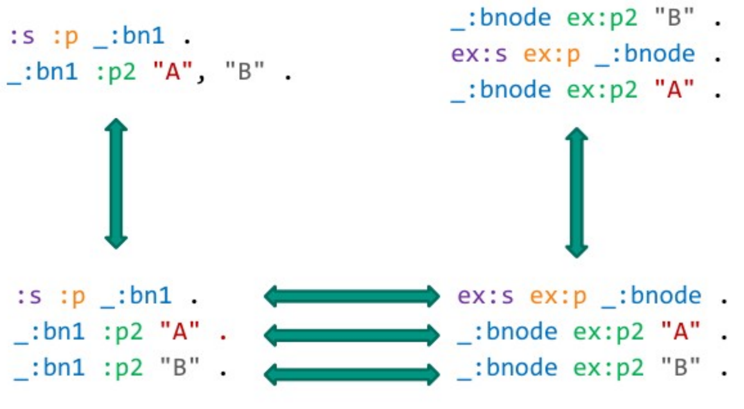


```
:s :p _:bn1 .
_:bn1 :p2 "A", "B" .
```



```
_:bnode ex:p2 "B" .
ex:s ex:p _:bnode .
_:bnode ex:p2 "A" .
```

$$
\begin{aligned}
M(\_{:}bn1) &= \_{:}bnode\\
M(:s) &= ex{:}s\\
M(:p) &= ex{:}p\\
M(:p2) &= ex{:}p2\\
M(\text{“}A\text{”}) &= \text{“}A\text{”}\\
M(\text{“}B\text{”}) &= \text{“}B\text{”}
\end{aligned}
$$



```
:s :p _:bn1 .
_:bn1 :p2 "A", "B" .
```

```
_:bnode ex:p2 "B" .
ex:s ex:p _:bnode .
_:bnode ex:p2 "A" .
```

```
:s :p _:bn1 .
_:bn1 :p2 "A" .
_:bn1 :p2 "B" .
```

```
ex:s ex:p _:bnode .
_:bnode ex:p2 "A" .
_:bnode ex:p2 "B" .
```

E3.5 Let G be the following graph `_:s <#p> <#o> , <#o2> .` Enumerate all subgraphs of G.

Solution:

<the empty graph>
_:s <#p> <#o> .
_:s <#p> <#o2> .
_:s <#p> <#o> , <#o2> .

E3.6 Let G be the following graph `<#s> <#p> <#o> , <#o2> .` Enumerate all instances of G.

Solution:

Since there are no blank nodes in the graph, there is only one instance:
<#s> <#p> <#o> , <#o2> .

E3.7 Let G be the following graph `_:s <#p> _:o , _:o2 .` Enumerate all instances of G.

Solution:

There is an infinite amount of instances:
Each blank node can be replaced with a new blank node.
Each blank node can be replaced with any URI.
Example: _:s <#p> _:s , <#foo> .

E3.8 Given is the following Turtle document at http://example.org/foo.ttl:

```
@prefix : <#> .
:Alice      :a      :Person ;
            :knows [ :a Person ; :name "Bob" ] .
[ :name "Bob" ] .
```

Parse the document and provide the resulting graph in N-Triples serialization.

Solution:

<http://example.org/foo.ttl#Alice> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://example.org/foo.ttl#Person> .
<http://example.org/foo.ttl#Alice> <http://example.org/foo.ttl#knows> _:genid1 .
_:genid1 <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://example.org/foo.ttl#Person> .
_:genid1 <http://example.org/foo.ttl#name> "Bob" .
_:genid2 <http://example.org/foo.ttl#name> "Bob" .

E3.9 Parse the following documents and merge the resulting graphs. Provide the resulting graph in N-Triples serialisation.

Turtle document at **http://example.org/people.ttl**:

```
@prefix : <people#> .
@prefix lib: <lib#> .

<>       lib:title "Some organisations mentioned in chapter 1" .

_:bn1    :name "European Organisation for Nuclear Research"@en .
         :location [ :name "Geneva"@en , "Genf"@de ] .

_:bn2    :name "Institut international de documentation"@fr ;
         :location [ :name "Brussels" ] .
```

Turtle document at http://example.org/lib.ttl:

```
@prefix : <lib#> .

<>      :title "Some publications mentioned in chapter 1" .

_:bn1   :title "Classification décimale universelle"@fr ; :year
1932 .

_:bn2   :title "As we may think"@en ; :year 1945 .
```

Solution:

<http://example.org/people.ttl> <http://example.org/lib#title> "Some organisations mentioned in chapter 1" .
<http://example.org/lib.ttl> <http://example.org/lib#title> "Some publications mentioned in chapter 1" .
_:genid1 <http://example.org/people#name> "European Organisation for Nuclear Research"@en .
_:genid1 <http://example.org/people#location> _:genid2 .
_:genid2 <http://example.org/people#name> "Geneva"@en .
_:genid2 <http://example.org/people#name> "Genf"@de .
_:genid3 <http://example.org/people#name> "Institut international de documentation"@fr .
_:genid3 <http://example.org/people#location> _:genid4 .
_:genid4 <http://example.org/people#name> "Brussels" .
_:genid5 <http://example.org/lib#title> "Classification décimale universelle"@fr .
_:genid5 <http://example.org/lib#year> "1932"^^<http://www.w3.org/2001/XMLSchema#integer> .
_:genid6 <http://example.org/lib#title> "As we may think"@en .
_:genid6 <http://example.org/lib#year> "1945"^^<http://www.w3.org/2001/XMLSchema#integer> .

E3.10 Out of the two documents in E 3.9, construct an RDF dataset with two named graphs and an empty default graph. Provide the graphs in the RDF dataset in Turtle format. Ensure the correct handling of blank nodes.

Solution:

@prefix : <people#> .
@prefix lib: <lib#> .

Named Graph 1: <http://example.org/people.ttl>
<http://example.org/people.ttl>   lib:title

```
"Some organisations mentioned in chapter 1" .
_:bn1   :name "European Organisation for Nuclear Research"@en ;
:location        [ :name "Geneva"@en , "Genf"@de ] .
_:bn2   :name "Institut international de documentation"@fr ;
:location        [ :name "Brussels" ] .
```

Named Graph 2: <http://example.org/lib.ttl>

```
<http://example.org/lib.ttl> lib:title
"Some publications mentioned in chapter 1" .
_:bn3   lib:title "Classification décimale universelle"@fr ;
lib:year          1932 .
_:bn4   lib:title "As we may think"@en ;
lib:year          1945 .
```

E3.11 We now consider a larger example with several RDF documents accessible as Linked Data. For brevity, we assume the following prefix declarations in the subsequent documents:

```
@prefix a: <http://example.org/a> .
@prefix b: <http://example.org/b> .
@prefix c: <http://example.org/c> .
@prefix d: <http://example.org/d> .
@prefix p: <http://example.org/p> .
```

Document at **http://example.org/a**:

```
a:i p:i a:j .
a:j p:i a:i .
a:i p:i b:i .
```

Document at **http://example.org/b**:

```
b:i p:i d:i .
```

Document at **http://example.org/c**:

```
b:i p:i c:i .
```

Document at **http://example.org/d**:

```
d:i p:i a:j .
a:i p:i d:i .
```

Document at **http://example.org/e**:

```
d:i p:i c:i .
```

Explain how to construct a local RDF dataset from the different documents. What options do you have for assigning graphs to the RDF dataset?

`Solution:`

- You can use the default graph or create named graphs for each graph to be merged.
- The names of the named graphs can represent the URI of the source file of the graph (or any other name, but that would not really make sense).

Practices

P3.1 Supply a custom Accept header in one of the RDF content types in a HTTP request with a user agent of your choice on a DBpedia request URI.

Learning Goals

G 3.1 Explain the benefits of a graph-structured data model and outline different serialization syntaxes for RDF graphs.

G 3.2 Correctly use RDF lists in both the Turtle syntax shortcut and the triple representation; correctly use reification in modelling.

G 3.3 Decide whether two RDF graphs are subgraphs of each other.

G 3.4 Check whether one graph is an instance of another graph; provide instance mappings between a graph and its instance.

G 3.5 Construct an RDF dataset from multiple RDF graphs