

ΑΝΑΠΑΡΑΣΤΑΣΗ ΓΝΩΣΗΣ ΣΤΟΝ ΠΑΓΚΟΣΜΙΟ ΙΣΤΟ

2020-2021

2^η Σειρά Ασκήσεων

Ονοματεπώνυμο: Αδαμόπουλος Κωνσταντίνος

ΑΜ: 6270 (1043750)

Ερώτημα 1

Α. Παρακάτω παρουσιάζεται ο κώδικας του .rdf αρχείου που αναπαριστά την πρόταση φυσικής γλώσσας :

Η Ιλιάδα συνθέθηκε απο ποιητή που έζησε τον 8ο αιώνα π.Χ. στην Ιωνία της Μικράς Ασίας.

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

```
<rdf:RDF
```

```
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:uni="http://www.mydomain.org/uni-ns/">
```

```
  <rdf:Description rdf:about="Ηλιάδα">
```

```
    <uni:composed_from rdf:resource="ποιητη"/>
```

```
    <uni:lived_in>8ο αιώνα</uni:lived_in>
```

```
    <uni:at>Ιωνία της Μικράς Ασίας</uni:at>
```

```
  </rdf:Description>
```

```
</rdf:RDF>
```

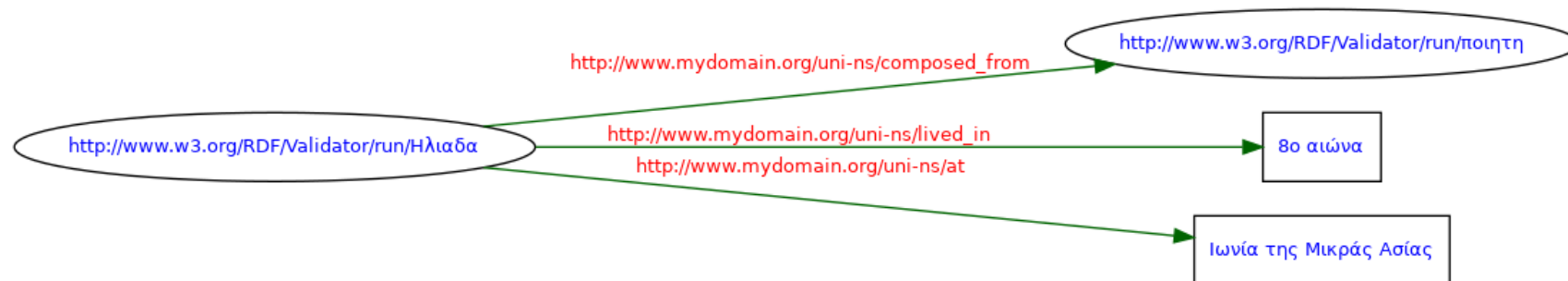
Οι τριπλέτες είναι :

Triples of the Data Model

Number	Subject	Predicate	Object
1	http://www.w3.org/RDF/Validator/run/Ηλιαδα	http://www.mydomain.org/uni-ns/composed_from	http://www.w3.org/RDF/Validator/run/ποιητη
2	http://www.w3.org/RDF/Validator/run/Ηλιαδα	http://www.mydomain.org/uni-ns/lived_in	"8ο αιώνα"
3	http://www.w3.org/RDF/Validator/run/Ηλιαδα	http://www.mydomain.org/uni-ns/at	"Ιωνία της Μικράς Ασίας"

Και το γράφημα :

Graph of the data model



B. Παρακάτω παρουσιάζεται ο κώδικας του .rdf αρχείου που αναπαριστά την παραπάνω πρόταση φυσικής γλώσσας χρησιμοποιώντας τουλάχιστον ένα κενό κόμβο:

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

<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:uni="http://www.mydomain.org/uni-ns/"

  <rdf:Description rdf:about="Ηλίας">
    <uni:poet rdf:resource="urn:uuid:fa123s"/>
  </rdf:Description>

  <rdf:Description rdf:about="urn:uuid:fa123s">
    <uni:lived_in>8ο αιώνα</uni:lived_in>
    <uni:at>Ιωνία της Μικράς Ασίας</uni:at>
  </rdf:Description>
</rdf:RDF>
```

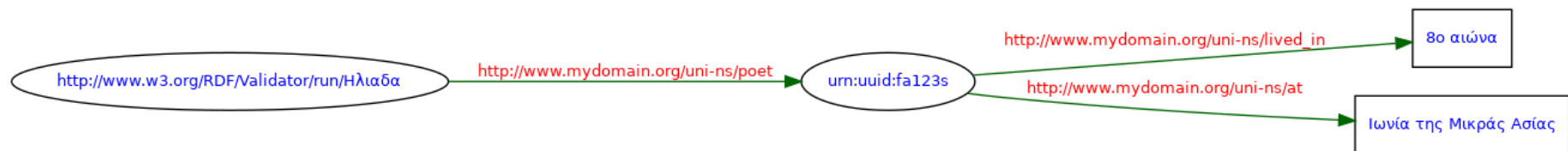
Οι τριπλέτες είναι :

Triples of the Data Model

Number	Subject	Predicate	Object
1	http://www.w3.org/RDF/Validator/run/Ηλιαδα	http://www.mydomain.org/uni-ns/poet	urn:uuid:fa123s
2	urn:uuid:fa123s	http://www.mydomain.org/uni-ns/lived_in	"8ο αιώνα"
3	urn:uuid:fa123s	http://www.mydomain.org/uni-ns/at	"Ιωνία της Μικράς Ασίας"

Και το γράφημα :

Graph of the data model



Ερώτημα 2

Παρακάτω παρουσιάζεται ο κώδικας του .rdf αρχείου που αναπαριστά την πρόταση φυσικής γλώσσας :

Η google αναφέρει ότι το Τμήμα Η/Υ & Πληροφορικής βρίσκεται στο Ρίο.

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

<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:uni="http://www.mydomain.org/uni-ns/"

  <rdf:Description rdf:about="http://mydomain/Statement1">
    <rdf:subject rdf:resource="Τμήμα Η/Υ και
Πληροφορικής"/>
    <rdf:predicate rdf:resource="βρίσκεται"/>
    <rdf:object rdf:resource="Ρίο"/>
  </rdf:Description>

  <rdf:Description rdf:about="Google">
    <uni:reports
rdf:resource="http://mydomain/Statement1"/>
  </rdf:Description>

</rdf:RDF>
```

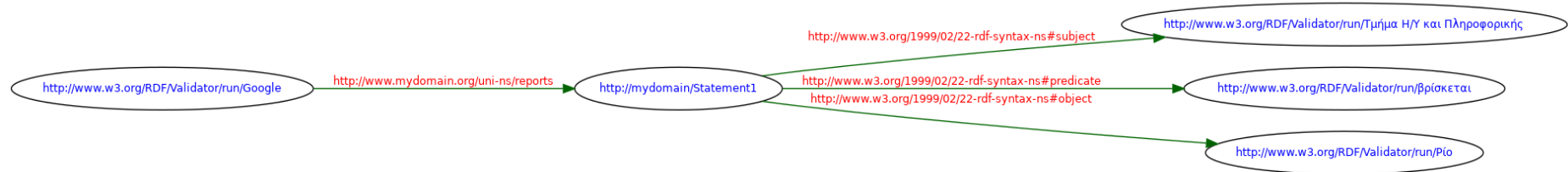
Οι τριπλέτες είναι :

Triples of the Data Model

Number	Subject	Predicate	Object
1	http://mydomain/Statement1	http://www.w3.org/1999/02/22-rdf-syntax-ns#subject	http://www.w3.org/RDF/Validator/run/Τμήμα_Η/Υ_και_Πληροφορικής
2	http://mydomain/Statement1	http://www.w3.org/1999/02/22-rdf-syntax-ns#predicate	http://www.w3.org/RDF/Validator/run/βρίσκεται
3	http://mydomain/Statement1	http://www.w3.org/1999/02/22-rdf-syntax-ns#object	http://www.w3.org/RDF/Validator/run/Pio
4	http://www.w3.org/RDF/Validator/run/Google	http://www.mydomain.org/uni-ns/reports	http://mydomain/Statement1

Και το γράφημα :

Graph of the data model



Ερώτημα 3

Ξεκινάω την δημιουργία της RDFS Οντολογίας εισάγοντας τα απαραίτητα namespaces και έπειτα δημιουργώ τις κλάσεις όπως φαίνονται παρακάτω:

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:univ="http://www.mydomain.org/univ/">

  <rdfs:Class rdf:about="http://www.mydomain.org/univ/Person">
  </rdfs:Class>

  <rdfs:Class rdf:about="http://www.mydomain.org/univ/Professor">
    <rdfs:subClassOf
  rdf:resource="http://www.mydomain.org/univ/Person"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="http://www.mydomain.org/univ/Student">
    <rdfs:subClassOf
  rdf:resource="http://www.mydomain.org/univ/Person"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="http://www.mydomain.org/univ/Department">
  </rdfs:Class>

  <rdfs:Class rdf:about="http://www.mydomain.org/univ/Classroom">
  </rdfs:Class>

  <rdfs:Class rdf:about="http://www.mydomain.org/univ/Lesson">
  </rdfs:Class>
```

Όπως φαίνεται παραπάνω δημιουργώ τις κλάσεις Person, Professor, Student, Department, Classroom, Lesson χρησιμοποιώντας το **<rdfs:Class>** tag. Έπειτα στις κλάσεις Professor και Student έχω εισάγει το tag **<rdfs:subClassOf>** και εισάγω σαν πόρο την Person το οποίο σημαίνει ότι η Professor και Student είναι υπό-κλάσεις της κλάσης Person. Αφού δήλωσα τις κλάσεις σειρά έχουν οι ιδιότητες αυτών οι οποίες παραθέτονται παρακάτω.

```
<rdfs:Property rdf:about="http://www.mydomain.org/univ/has_name">
  <rdfs:domain
  rdf:resource="http://www.mydomain.org/univ/Person"/>
  <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-
  syntax-ns#Literal"/>
</rdfs:Property>

<rdfs:Property rdf:about="http://www.mydomain.org/univ/has_phone">
  <rdfs:domain
  rdf:resource="http://www.mydomain.org/univ/Person"/>
  <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-
  syntax-ns#Literal"/>
</rdfs:Property>

<rdfs:Property rdf:about="http://www.mydomain.org/univ/has_email">
  <rdfs:domain
  rdf:resource="http://www.mydomain.org/univ/Person"/>
  <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-
  syntax-ns#Literal"/>
```

```
</rdfs:Property>

<rdfs:Property rdf:about="http://www.mydomain.org/univ/has_age">
  <rdfs:domain
rdf:resource="http://www.mydomain.org/univ/Person"/>
  <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-
syntax-ns#Integer"/>
</rdfs:Property>

<rdfs:Property rdf:about="http://www.mydomain.org/univ/member_of">
  <rdfs:domain
rdf:resource="http://www.mydomain.org/univ/Person"/>
  <rdfs:range
rdf:resource="http://www.mydomain.org/univ/Department"/>
</rdfs:Property>

<rdfs:Property rdf:about="http://www.mydomain.org/univ/teaches">
  <rdfs:domain
rdf:resource="http://www.mydomain.org/univ/Professor"/>
  <rdfs:range
rdf:resource="http://www.mydomain.org/univ/Lesson"/>
</rdfs:Property>

<rdfs:Property rdf:about="http://www.mydomain.org/univ/les_name">
  <rdfs:domain
rdf:resource="http://www.mydomain.org/univ/Lesson"/>
  <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-
syntax-ns#Literal"/>
</rdfs:Property>

<rdfs:Property rdf:about="http://www.mydomain.org/univ/taught_by">
  <rdfs:domain
rdf:resource="http://www.mydomain.org/univ/Lesson"/>
  <rdfs:range
rdf:resource="http://www.mydomain.org/univ/Professor"/>
</rdfs:Property>

<rdfs:Property rdf:about="http://www.mydomain.org/univ/dep_name">
  <rdfs:domain
rdf:resource="http://www.mydomain.org/univ/Department"/>
  <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-
syntax-ns#Literal"/>
</rdfs:Property>

<rdfs:Property rdf:about="http://www.mydomain.org/univ/dep_city">
  <rdfs:domain
rdf:resource="http://www.mydomain.org/univ/Department"/>
  <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-
syntax-ns#Literal"/>
</rdfs:Property>

<rdfs:Property rdf:about="http://www.mydomain.org/univ/room_name">
  <rdfs:domain
rdf:resource="http://www.mydomain.org/univ/Classroom"/>
  <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-
syntax-ns#Literal"/>
</rdfs:Property>

<rdfs:Property
rdf:about="http://www.mydomain.org/univ/room_capacity">
```



```

        <rdfs:domain
rdf:resource="http://www.mydomain.org/univ/Classroom"/>
        <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-
syntax-ns#Integer"/>
</rdfs:Property>

<rdfs:Property
rdf:about="http://www.mydomain.org/univ/room_department">
    <rdfs:domain
rdf:resource="http://www.mydomain.org/univ/Classroom"/>
    <rdfs:range
rdf:resource="http://www.mydomain.org/univ/Department"/>
</rdfs:Property>

```

Για να φτιάξουμε τις ιδιότητες που θα έχει μια κλάση χρησιμοποιούμε το tag **<rdfs:Property rdf:about=...>** όπου στο rdf:about θα είναι το όνομα της ιδιότητας μαζί με το URI . Έπειτα ορίζουμε τα tags **<rdfs:domain>** και **<rdfs:range>** όπου στο **<rdfs:domain>** ορίζουμε την κλάση στην οποία ανήκει η ιδιότητα και στο **<rdfs:range>** ορίζουμε το πεδίο τιμών της ιδιότητας . Συγκεκριμένα για την ιδιότητα has_name βλέπουμε ότι το domain της έχει την κλάση Person και το range της έχει την τιμή Literal που σημαίνει ότι η ιδιότητα has_name ανήκει στην κλάση Person και παίρνει αλφαριθμητικές τιμές. Το ίδιο ισχύει και για τις ιδιότητες has_phone, has_email και η has_age ανήκει και αυτή στην κλάση Person μόνο που δέχεται ακέραιους αριθμούς αντί για αλφαριθμητικές τιμές. Στην ιδιότητα member_of βλέπουμε ότι το domain της έχει την τιμή Person αλλά το range έχει την τιμή Department αυτό σημαίνει ότι η ιδιότητα αυτή ανήκει στην κλάση Person και ότι οι τιμές που μπορεί να πάρει αυτή η ιδιότητα είναι τύπου Department η οποία όπως είδαμε πιο πάνω είναι κλάση . Η ιδιότητα teaches ανήκει στην κλάση Professor και οι τιμές που μπορεί να πάρει είναι τύπου Lesson (κλάση) . Η ιδιότητα les_name ανήκει στην κλάση Lesson και οι τιμές που μπορεί να πάρει είναι αλφαριθμητικές . Η ιδιότητα taught_by ανήκει στην κλάση Lesson και οι τιμές που μπορεί να πάρει είναι τύπου Professor (κλάση). Οι ιδιότητες dep_name, dep_city ανήκουν στην κλάση Department και οι τιμές που μπορούν να πάρουν είναι αλφαριθμητικές. Η ιδιότητα room_name ανήκει στην κλάση Classroom και οι τιμές που μπορεί να πάρει είναι αλφαριθμητικές. Η ιδιότητα room_capacity ανήκει στην κλάση Classroom και οι τιμές που μπορεί να πάρει είναι ακέραιες και η ιδιότητα room_department ανήκει στην κλάση Classroom και οι τιμές που μπορεί να πάρει είναι τύπου Department (κλάση) .Έπειτα δημιουργούμε στιγμιότυπα για τις παραπάνω κλάσεις τα οποία υπάρχουν στο έγγραφο του ερωτήματος 3.

Ερώτημα 4

- i. Το SPARQL query το οποίο επιστρέφει τα τηλέφωνα των καθηγητών είναι:

```

PREFIX univ: <http://www.mydomain.org/univ/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/22-rdf-schema#>
SELECT ?name ?phone
WHERE{
    ?x rdfs:type univ:Professor.

```

```

    ?x univ:has_phone ?phone .
    ?x univ:has_name ?name.
}

```

Η μεταβλητή ?x περιέχει μόνο τα στιγμιότυπα τα οποία είναι τύπου Professor και έπειτα για το αντικείμενο που έχει η μεταβλητή ?x παίρνουμε την τιμή της ιδιότητας **univ:has_phone** και την τιμή της ιδιότητας **univ:has_name** και τις αποθηκεύουμε στις μεταβλητές **?name** και **?phone** αντίστοιχα .

Το οποίο επιστρέφει σαν έξοδο:

name	phone
"P8"	"6983231758"
"P10"	"6803039464"
"P3"	"6993639508"
"P2"	"6953609764"
"P9"	"6553630762"
"P6"	"6173030368"
"P4"	"6393290768"
"P5"	"6977679728"
"P7"	"6473332798"
"P1"	"6973639768"

Όπου η στήλη name δείχνει το όνομα των καθηγητών και η στήλη phone δείχνει τα τηλεφωνά τους .

- ii. Το SPARQL query το οποίο επιστρέφει τα τηλέφωνα των μαθητών που έχουν ηλικία πάνω από 23 έτη είναι:

```

PREFIX univ: <http://www.mydomain.org/univ/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/22-rdf-schema#>
SELECT ?name ?phone
WHERE{
    ?x rdf:type univ:Student .
    ?x univ:has_phone ?phone .
    ?x univ:has_name ?name.
    ?x univ:has_age ?age .
    FILTER(xsd:int(?age) > 23)
}

```

Η μεταβλητή ?x περιέχει μόνο τα στιγμιότυπα τα οποία είναι τύπου Student και έπειτα για το αντικείμενο που έχει η μεταβλητή ?x παίρνω την τιμή της ιδιότητας **univ:has_phone** ,την τιμή της ιδιότητας **univ:has_name** και την τιμή της ιδιότητας **univ:has_age** και τις αποθηκεύουμε στις μεταβλητές **?name** **?phone** και **?age** αντίστοιχα και έπειτα λόγω του **FILTER** φιλτράρουμε το αποτέλεσμα ανάλογα με το τι τιμή έχει η

μεταβλητή ?age. Αν είναι μεγαλύτερη από 23 κρατάω το αποτέλεσμα αλλιώς συνεχίζω με το επόμενο entry.

Το οποίο επιστρέφει σαν έξοδο:

name	phone
"S1"	"6800039364"
"S9"	"6403039064"

Όπου η στήλη name δείχνει το όνομα των μαθητών και η στήλη phone δείχνει τα τηλεφωνά τους .

- iii. Το SPARQL query το οποίο επιστρέφει το όνομα όλων των ατόμων που είναι μέλη σε τμήμα που είναι στην Πάτρα είναι:

```
PREFIX univ: <http://www.mydomain.org/univ/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/22-rdf-schema#>
SELECT ?name ?members_of
WHERE {
    ?x univ:has_name ?name;
    univ:member_of ?members_of.
    ?members_of univ:dep_city ?city
    FILTER(?city="Patras")
}
```

Η μεταβλητή ?x «δέχεται» τα στιγμιότυπα τα οποία έχουν την ιδιότητα univ:has_name που ουσιαστικά είναι οι καθηγητές(Professor class) και οι μαθητές (Student class) και αποθηκεύει την τιμή στην μεταβλητή ?name. Έπειτα παίρνω την τιμή της ιδιότητας univ:member_of την αποθηκεύω στην μεταβλητή ?members_of και στην συνέχεια χρησιμοποιώ την μεταβλητή ?members_of η οποία περιέχει το Τμήμα στο οποίο ανήκει ο μαθητής/καθηγητής και παίρνω την τιμή της ιδιότητας univ:dep_city και την αποθηκεύω στην μεταβλητή ?city. Έπειτα μέσω του FILTER κοιτάω για το αν η μεταβλητή ?city έχει την τιμή Patras αν ναι τότε το αποτέλεσμα καταγράφεται αλλιώς συνεχίζω στα επόμενα entries .

Το οποίο επιστρέφει σαν έξοδο:

name	members_of
"P4"	univ:D2
"P5"	univ:D2
"S3"	univ:D2
"P3"	univ:D1
"P2"	univ:D1
"S1"	univ:D1
"P1"	univ:D1
"S2"	univ:D1
"S5"	univ:D3
"S4"	univ:D3
"P6"	univ:D3

Όπου η στήλη name δείχνει το όνομα των ατόμων και η στήλη members_of δείχνει τα τμήματα στα οποία ανήκουν τα άτομα αυτά .

- iv. Το SPARQL query το οποίο επιστρέφει το όνομα όλων των τάξεων των οποίων το τμήμα είναι στην Πάτρα και έχουν χωρητικότητα μεγαλύτερη από 150 είναι:

```
PREFIX univ: <http://www.mydomain.org/univ/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/22-rdf-schema#>
SELECT ?classroom_name
WHERE {
  ?x rdf:type univ:Classroom.
  ?x univ:room_name ?classroom_name;
  univ:room_capacity ?cap
  filter(xsd:int(?cap)>150)
  ?x univ:room_department ?dep.
  ?dep univ:dep_city ?city
  FILTER(?city="Patras") }
```

Το οποίο επιστρέφει σαν έξοδο:

classroom_name
"C12"
"C22"
"C32"

Όπου η στήλη `classroom_name` δείχνει το όνομα των τάξεων που είναι σε τμήμα το οποίο είναι στην Πάτρα και έχουν χωρητικότητα μεγαλύτερη του 150 .

Ερώτημα 5

A. Για το **A.** ερώτημα δημιουργώ για αρχή έναν file chooser frame για να μπορώ να διαλέξω το αρχείο. Έπειτα διαβάζω το αρχείο και δημιουργείται το μοντέλο που αναπαριστά το .rdf αρχείο που διαβάστηκε, στην συνέχεια δημιουργώ ένα SPARQL query σε μια String μεταβλητή και έπειτα μέσω της **Query query = QueryFactory.create(queryString);** Δημιουργώ ένα **Query object** το οποίο εκτελώ με βάση το model που έχει δημιουργηθεί διαβάζοντας το .rdf αρχείο προηγουμένως . Μετά για όλα τα entries που επιστρέφει το query(for-loop) παίρνω τον rdf node που αντιστοιχεί στην τιμή που έχει η μεταβλητή ?x του SPARQL ερωτήματος και γράφω αυτή την τιμή στον πίνακα dataTable του GUI και ενεργοποιώ την λειτουργία δημιουργίας νέων εγγραφών καθώς και την λειτουργία εμφάνισης των Statements για ένα συγκεκριμένο URI . Η συνάρτηση που επιτελεί τα παραπάνω είναι η:

```
private void open_btnActionPerformed(java.awt.event.ActionEvent evt)
{
    FileFilter filter = new FileNameExtensionFilter("RDF file",
    "rdf");
    jFileChooser.addChoosableFileFilter(filter);
    jFileChooser.setFileFilter(filter);
    DefaultTableModel tableModel =(DefaultTableModel)
    dataTable.getModel();
    int returnVal = jFileChooser.showOpenDialog(this);
    if (returnVal == JFileChooser.APPROVE_OPTION) {
        file = jFileChooser.getSelectedFile();
        String path = file.getAbsolutePath();
        path_label.setText(file.getName());
        InputStream in = FileManager.get().open(path);
        InputStreamReader rin = new
        InputStreamReader(in,Charset.forName("UTF-8").newDecoder());
        model = ModelFactory.createDefaultModel();
        model.read(rin,"");
        // model.write(System.out);
        String queryString = "PREFIX univ:
        <http://www.mydomain.org/univ/> PREFIX xsd:
        <http://www.w3.org/2001/XMLSchema#> PREFIX rdf:
        <http://www.w3.org/1999/02/22-rdf-syntax-ns#> PREFIX rdfs:
        <http://www.w3.org/2000/01/22-rdf-schema#> SELECT ?x {?x rdf:type
        univ:Department . }";
        Query query = QueryFactory.create(queryString);
        query.serialize(new IndentedWriter(System.out,true));
        QueryExecution qexec =
        QueryExecutionFactory.create(query,model);
        ResultSet rs = qexec.execSelect();
        for (; rs.hasNext());{
            QuerySolution rb = rs.nextSolution();
            RDFNode x = rb.get("x");
            System.out.println(x.toString());
            tableModel.addRow(new Object[]{x});
        }
    }
```

```

    }
    URI = "http://www.mydomain.org/univ/";
}
dataCategory.setEnabled(true);
stmt_btn.setEnabled(true);
}

```

Τώρα στην συνέχεια του **A.** ερωτήματος, όταν ο χρήστης επιλέξει ένα από τα τμήματα που αναγράφονται τότε θα εκτελεστεί η μέθοδος **dataTableMouseClicked** η οποία θα τρέξει μόνο όταν γίνει ένα click event στον πίνακα dataTable. Όταν ξεκινήσει η εκτέλεση αυτής της μεθόδου ξανά διαβάζει το αρχείο ώστε να πάρει τις τελευταίες αλλαγές, αν έχουν γίνει στο έγγραφο και αμέσως μετά παίρνει τον αριθμό της γραμμής του πίνακα που επέλεξε ο χρήστης και με βάση τον αριθμό της γραμμής παίρνει την τιμή που έχει ο πίνακας σε εκείνη την γραμμή και την αποθηκεύει στην μεταβλητή department. Έπειτα δημιουργώ και ένα instance της κλάσης AllData που είναι η κλάση του παραθύρου που θα εμφανιστούν τα τελικά δεδομένα, στην συνέχεια εκτελούνται τα απαραίτητα SPARQL ερωτήματα από τα οποία παίρνουμε το όνομα του τμήματος που επέλεξε ο χρήστης καθώς και όλους τους Καθηγητές, Μαθητές, Μαθήματα και Τάξεις που έχει το συγκεκριμένο τμήμα και τα εισάγει στους πίνακες professorsTable, studentsTable, lessonsTable και classesTable αντίστοιχα μέσω της μεθόδου writeToTable η οποία ανήκει στην κλάση AllData που είναι η κλάση του παραθύρου που περιέχει τους παραπάνω πίνακες. Η μέθοδος writeToTable δέχεται 4 παραμέτρους τύπου String από τις οποίες η πρώτη αφορά το σε ποιον πίνακα θα γράψει τις υπόλοιπες τρεις παραμέτρους που είναι τα δεδομένα που ανακτώνται από τα SPARQL ερωτήματα.

Η συνάρτηση **dataTableMouseClicked** βρίσκεται στην κλάση **Ask2Frame** και ο κώδικας της φαίνεται παρακάτω :

```

private void dataTableMouseClicked(java.awt.event.MouseEvent evt) {

    InputStream in =
    FileManager.get().open(file.getAbsolutePath());
    InputStreamReader rin = new
    InputStreamReader(in, Charset.forName("UTF-8").newDecoder());
    model.read(rin, "");

    int row = dataTable.getSelectedRow();
    int col = 0;
    Object department = dataTable.getValueAt(row, col);
    String str_dep = department.toString();
    System.out.printf("Dep! "+str_dep);
    System.out.printf("Clicked! %d
\n", dataTable.getSelectedRow());

    // dataWindow.setVisible(true);
    Literal title=null,
           name = null,
           age=null,
           phone=null,

```

```

        cap=null,
        taught_by=null;
        AllData dataWindow = new AllData();

        String queryString = "PREFIX univ:
<http://www.mydomain.org/univ/>" +
        "PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>" +
        "PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-
syntax-ns#>" +
        "PREFIX rdfs: <http://www.w3.org/2000/01/22-rdf-
schema#>" +
        "SELECT ?name WHERE {?x rdf:type univ:Department ;
univ:dep_name ?name . }" +
        "FILTER(?x=<"+str_dep+">)" +
        "}" ;
        Query query = QueryFactory.create(queryString);
        query.serialize(new IndentedWriter(System.out, true));
        QueryExecution qexec = QueryExecutionFactory.create(query,
model);
        ResultSet rs = qexec.execSelect();
        if(rs.hasNext()){
            for (; rs.hasNext();){
                QuerySolution rb = rs.nextSolution();
                title = rb.getLiteral("name");
                System.out.printf("Dep! " + title.toString());
            }
            qexec.close();
            dataWindow.setTitle("Τμήμα " + title.toString());
        }else{
            JOptionPane.showMessageDialog(this, "Το Τμήμα που
επέλεξες δεν έχει δεδομένα.");
            return;
        }

        String ProfQuerystr = "PREFIX univ:
<http://www.mydomain.org/univ/>"
        + "PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>"
        + "PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-
syntax-ns#>"
        + "PREFIX rdfs: <http://www.w3.org/2000/01/22-rdf-
schema#>"
        + "SELECT ?name ?age ?phone WHERE { ?x rdf:type
univ:Professor . ?x univ:has_name ?name . ?x univ:has_age ?age . ?x
univ:has_phone ?phone . }"
        + "?x univ:member_of ?dep ."
        + "FILTER(?dep=<"+str_dep+">)}";
        Query prof_query = QueryFactory.create(ProfQuerystr);
        prof_query.serialize(new IndentedWriter(System.out, true));
        QueryExecution p_qexec =
QueryExecutionFactory.create(prof_query, model);
        ResultSet p_rs = p_qexec.execSelect();
        for (; p_rs.hasNext();){
            QuerySolution p_rb = p_rs.nextSolution();
            name = p_rb.getLiteral("name");
            age = p_rb.getLiteral("age");
            phone = p_rb.getLiteral("phone");
            System.out.printf("Dep! " + name.toString());

            dataWindow.writeToTable("professor", name.toString(), age.toString(), ph
one.toString());

```

```

    }
    p_gexec.close();

    String StudQuerystr = "PREFIX univ:
<http://www.mydomain.org/univ/>"
        + "PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>"
        + "PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-
syntax-ns#>"
        + "PREFIX rdfs: <http://www.w3.org/2000/01/22-rdf-
schema#>"
        + "SELECT ?name ?age ?phone WHERE { ?x rdf:type
univ:Student . ?x univ:has_name ?name . ?x univ:has_age ?age . ?x
univ:has_phone ?phone ."
        + "?x univ:member_of ?dep ."
        + "FILTER(?dep=<" + str_dep + ">)}";
    Query stud_query = QueryFactory.create(StudQuerystr);
    stud_query.serialize(new IndentedWriter(System.out, true));
    QueryExecution s_gexec =
    QueryExecutionFactory.create(stud_query, model);
    ResultSet s_rs = s_gexec.execSelect();
    for (; s_rs.hasNext();) {
        QuerySolution s_rb = s_rs.nextSolution();
        name = s_rb.getLiteral("name");
        age = s_rb.getLiteral("age");
        phone = s_rb.getLiteral("phone");
        System.out.printf("Dep! " + name.toString());
        dataWindow.writeToTable("student", name.toString(),
age.toString(), phone.toString());
    }
    s_gexec.close();

    String ClassQuerystr = "PREFIX univ:
<http://www.mydomain.org/univ/>"
        + "PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>"
        + "PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-
syntax-ns#>"
        + "PREFIX rdfs: <http://www.w3.org/2000/01/22-rdf-
schema#>"
        + "SELECT ?name ?capacity WHERE { ?x rdf:type
univ:Classroom . ?x univ:room_name ?name . ?x univ:room_capacity
?capacity ."
        + "?x univ:room_department ?dep ."
        + "FILTER(?dep=<" + str_dep + ">)}";
    Query class_query = QueryFactory.create(ClassQuerystr);
    class_query.serialize(new IndentedWriter(System.out, true));
    QueryExecution c_gexec =
    QueryExecutionFactory.create(class_query, model);
    ResultSet c_rs = c_gexec.execSelect();
    for (; c_rs.hasNext();) {
        QuerySolution c_rb = c_rs.nextSolution();
        name = c_rb.getLiteral("name");
        cap = c_rb.getLiteral("capacity");
        System.out.printf("Dep! " + name.toString());
        dataWindow.writeToTable("classroom", name.toString(),
cap.toString(), "");
    }
    c_gexec.close();

    String LessonQuerystr = "PREFIX univ:
<http://www.mydomain.org/univ/>"
        + "PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>"

```



```

        + "PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-
syntax-ns#>"
        + "PREFIX rdfs: <http://www.w3.org/2000/01/22-rdf-
schema#>"
        + "SELECT ?name ?teacher WHERE { ?prof rdf:type
univ:Professor . ?prof univ:teaches ?y . ?y univ:les_name ?name .
?prof univ:has_name ?teacher .}"
        + "?prof univ:member_of ?dep ."
        + "FILTER(?dep=<" + str_dep + ">)}";
Query lesson_query = QueryFactory.create(LessonQuerystr);
lesson_query.serialize(new IndentedWriter(System.out, true));
QueryExecution l_gexec =
QueryExecutionFactory.create(lesson_query, model);
ResultSet l_rs = l_gexec.execSelect();
for (; l_rs.hasNext(); ) {
    QuerySolution l_rb = l_rs.nextSolution();
    name = l_rb.getLiteral("name");
    taught_by = l_rb.getLiteral("teacher");
    System.out.printf("Dep! " + name.toString());
    dataWindow.writeToTable("lessons", name.toString(),
taught_by.toString(), "");
}
l_gexec.close();

dataWindow.setVisible(true);
}

```

Η συνάρτηση **writeToTable** βρίσκεται στην κλάση **AllData** και ο κώδικας της φαίνεται παρακάτω :

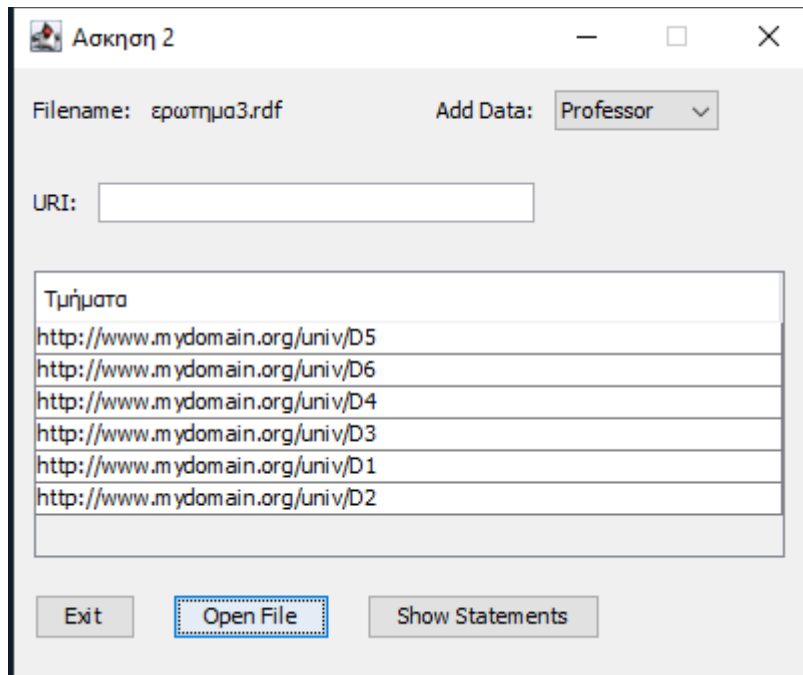
```

public void writeToTable(String table,String v1,String v2,String v3) {
    if(table=="professor"){
        DefaultTableModel tableModel =(DefaultTableModel)
professorsTable.getModel();
        tableModel.addRow(new Object[]{v1,v2,v3});
    }else if(table=="student"){
        DefaultTableModel tableModel =(DefaultTableModel)
studentsTable.getModel();
        tableModel.addRow(new Object[]{v1,v2,v3});
    }else if(table=="classroom"){
        DefaultTableModel tableModel =(DefaultTableModel)
classesTable.getModel();
        tableModel.addRow(new Object[]{v1,v2});
    }else if(table=="lessons"){
        DefaultTableModel tableModel =(DefaultTableModel)
lessonsTable.getModel();
        tableModel.addRow(new Object[]{v1,v2});
    }
}

```

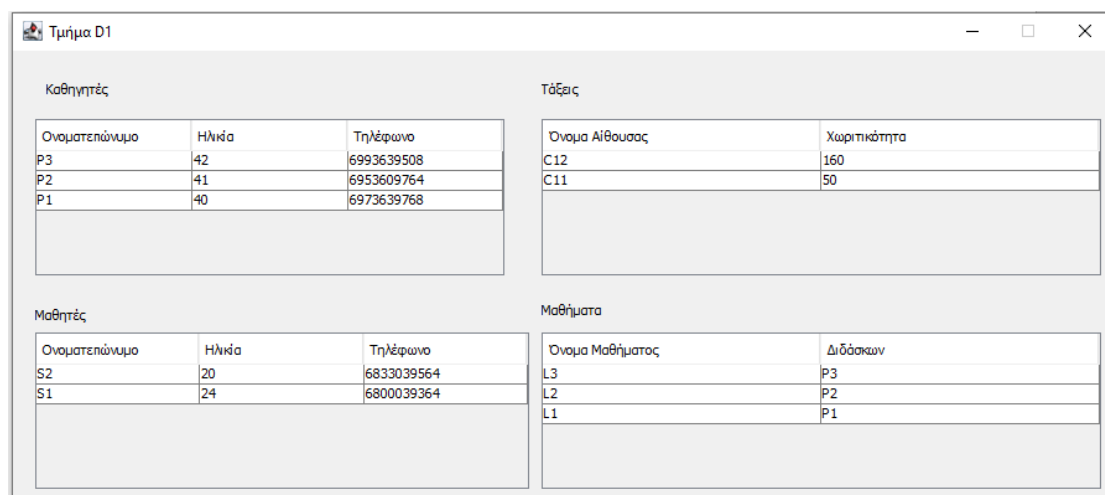
Παρακάτω παρουσιάζονται screenshots από την εκτέλεση του Α. ερωτήματος της εφαρμογής .

Όταν έχω διαλέξει το αρχείο το οποίο είναι το ερώτημα3.rdf σύμφωνα με την εκφώνηση.



The screenshot shows a window titled "Ασκηση 2". It has a "Filename:" field with the value "ερωτημα3.rdf" and an "Add Data:" dropdown menu currently set to "Professor". Below these is a "URI:" text input field. A list box titled "Τμήματα" contains several URIs: "http://www.mydomain.org/univ/D5", "http://www.mydomain.org/univ/D6", "http://www.mydomain.org/univ/D4", "http://www.mydomain.org/univ/D3", "http://www.mydomain.org/univ/D1", and "http://www.mydomain.org/univ/D2". At the bottom are three buttons: "Exit", "Open File" (which is highlighted with a blue dashed border), and "Show Statements".

Όταν διαλέξω το τμήμα D1 :



The screenshot shows a window titled "Τμήμα D1" with four data tables arranged in a 2x2 grid:

- Καθηγητές (Teachers):**

Όνοματεπώνυμο	Ηλικία	Τηλέφωνο
P3	42	6993639508
P2	41	6953609764
P1	40	6973639768

- Τάξεις (Classes):**

Όνομα Αίθουσας	Χωρητικότητα
C12	160
C11	50

- Μαθητές (Students):**

Όνοματεπώνυμο	Ηλικία	Τηλέφωνο
S2	20	6833039564
S1	24	6800039364

- Μαθήματα (Lessons):**

Όνομα Μαθήματος	Διδάσκων
L3	P3
L2	P2
L1	P1

Β. Για το Β. ερώτημα δημιουργώ ένα combo Box στο παράθυρο της κλάσης Ask2Frame το οποίο περιέχει τις κατηγορίες Professor, Student , Department, Lesson, Classroom. Όταν ο χρήστης επιλέξει ένα από αυτά τότε ανοίγει το αντίστοιχο παράθυρο, ο χρήστης γράφει τα δεδομένα στα πεδία των ιδιοτήτων της

κατηγορίας που διάλεξε και έπειτα πατώντας το κουμπί Add προσθέτει το instance στο .rdf αρχείο . Η λειτουργία για το πιο παράθυρο θα ανοίξει με βάση το τι επέλεξε ο χρήστης υλοποιεί η μέθοδος ***dataCategoryActionPerformed*** η οποία όταν ο χρήστης πατήσει click σε κάποια επιλογή η μέθοδος παίρνει την επιλογή του χρήστη και δημιουργεί ένα instance του παραθύρου που αντιστοιχεί σε μία από τις παραπάνω κατηγορίες . Έπειτα αφού δημιουργηθεί το αντικείμενο καλείται η μέθοδος ***getModel*** ή ***getModelTable*** στην περίπτωση που ο χρήστης θέλει να εισάγει νέο τμήμα στο .rdf αρχείο όπου αυτές η μέθοδοι παίρνουν το μονοπάτι του αρχείου, το URI και το model που δημιουργείται από το διάβασμα του .rdf αρχείου (και το model του πίνακα dataTable στην περίπτωση του Department) και τα αποθηκεύουν στις μεταβλητές των αντίστοιχων κλάσεων . Οι μέθοδοι που κάνουν τα παραπάνω είναι οι:

dataCategoryActionPerformed η οποία είναι στην κλάση ***Ask2Frame*** και οι μέθοδοι ***getModel*** ή ***getModelTable*** βρίσκονται στις κλάσεις ***ProfessorData, StudentData, LessonData, ClassroomData και DepartmentData*** αντίστοιχα .

Ο κώδικας τους παρουσιάζεται παρακάτω :

```
private void dataCategoryActionPerformed(java.awt.event.ActionEvent
evt) {

    String ch = dataCategory.getSelectedItem().toString();
    System.out.println("DataChoses:"+ch);
    if(ch.equals("Professor")){
        ProfessorData pd = new ProfessorData();
        pd.getModel(model,URI,file.getAbsolutePath());
        pd.setVisible(true);
    }else if(ch.equals("Lesson")){
        LessonData ld = new LessonData();
        ld.getModel(model,URI,file.getAbsolutePath());
        ld.setVisible(true);
    }else if(ch.equals("Student")){
        StudentData sd = new StudentData();
        sd.getModel(model,URI,file.getAbsolutePath());
        sd.setVisible(true);
    }else if(ch.equals("Department")){
        DepartmentData dd = new DepartmentData();
        DefaultTableModel tableModel =(DefaultTableModel)
dataTable.getModel();

        dd.getModelTable(model,URI,file.getAbsolutePath(),tableModel);
        dd.setVisible(true);
    }else if(ch.equals("Classroom")){
        ClassroomData cd = new ClassroomData();
        cd.getModel(model,URI,file.getAbsolutePath());
        cd.setVisible(true);
    }
}

public void getModel(Model m,String uri_var,String f){
    model = m;
    uri = uri_var;
    file = f;
}
```

```

public void getModelTable(Model m,String uri_var,String
f,DefaultTableModel tmodel){
    model = m;
    uri = uri_var;
    file = f;
    tableModel = tmodel;
}

```

Όταν ο χρήστης πατήσει το κουμπί Add σε ένα από τα παράθυρα που αναφέρθηκαν πιο πάνω τότε εκτελείται η μέθοδος ***addToRdfActionPerformed*** η οποία παίρνει τα δεδομένα που εισήγαγε ο χρήστης και έπειτα ελέγχει για το αν κάποιο πεδίο είναι κενό. Αν ναι τότε εμφανίζεται κατάλληλο μήνυμα, αλλιώς δημιουργώ ένα Resource object το οποίο περιέχει την τιμή URI και το όνομα του αντικειμένου της κατηγορίας που θέλει να εισάγει στο αρχείο. Έπειτα δημιουργώ τόσα Property objects όσα και οι ιδιότητες που έχει η κάθε κατηγορία και τα «συνδέω» με το Resource object που δημιούργησα πιο πάνω και σε αυτές τις ιδιότητες βάζω τις τιμές που έχει εισάγει ο χρήστης και έπειτα αφού το Resource έχει προστεθεί στο model κατά την δημιουργία του , γράφω το model στο αρχείο που επέλεξε αρχικά ο χρήστης . Στην περίπτωση που ο χρήστης εισάγει ένα νέο τμήμα η μέθοδος αφού γράψει τα νέα δεδομένα στο αρχείο, θα το ξανά διαβάσει και θα εκτελέσει ένα SPARQL ερώτημα για να πάρει μόνο το τμήμα που προστέθηκε και να το εισάγει στον πίνακα dataTable με αυτό τον τρόπο ενημερώνεται ο πίνακας των τμημάτων μετά από κάθε προσθήκη τμήματος .

Η μέθοδος ***addToRdfActionPerformed*** βρίσκεται στις κλάσεις ***ProfessorData,StudentData,LessonData,ClassroomData και DepartmentData*** αντίστοιχα .

Οι κώδικες παρουσιάζονται παρακάτω :

Στην κλάση ***ProfessorData***:

```

private void addToRdfActionPerformed(java.awt.event.ActionEvent evt)
{

    String name  = this.name_field.getText();
    String age   = this.age_field.getText();
    String email = this.email_field.getText();
    String phone = this.phone_field.getText();
    String lesson_str = this.less_field.getText();
    String department_str = this.dep_field.getText();

    if (!"".equals(name) && !"".equals(age) && !"".equals(email)
&& !"".equals(phone) && !"".equals(lesson_str) &&
!"".equals(department_str)) {
        Resource professor = model.createResource(uri+name);
        Property p1 = model.createProperty(uri, "has_name");
        Property p2 = model.createProperty(uri, "has_phone");
        Property p3 = model.createProperty(uri, "has_email");
        Property p4 = model.createProperty(uri, "has_age");
        Property p5 = model.createProperty(uri, "teaches");
        Property p6 = model.createProperty(uri, "member_of");
    }
}

```

```

professor.addProperty(RDF.type,model.createResource(uri+"Professor"))
;

    professor.addProperty(p1,name);
    professor.addProperty(p2,phone);
    professor.addProperty(p3,email);
    professor.addProperty(p4,age);

professor.addProperty(p5,model.createResource(uri+lesson_str));

professor.addProperty(p6,model.createResource(uri+department_str));

    FileOutputStream out = null;
    OutputStreamWriter sw = null;
    try {
        out = new FileOutputStream(file,false);
        sw = new OutputStreamWriter(out,Charset.forName("UTF-
8").newEncoder());
    } catch (FileNotFoundException ex) {

Logger.getLogger(ProfessorData.class.getName()).log(Level.SEVERE,
null, ex);
    }
    RDFDataMgr.write(sw, model, Lang.RDFXML);

} else {
    JOptionPane.showMessageDialog(this,"Κάποιο απο τα πεδία
δέν έχει τιμή.");
}
}

```

Στην κλάση *StudentData*:

```

private void addToRdfActionPerformed(java.awt.event.ActionEvent evt)
{

    String name = this.stud_name.getText();
    String age = this.stud_age.getText();
    String email = this.stud_email.getText();
    String phone = this.stud_phone.getText();
    String department_str = this.stud_dep.getText();

    if (!"".equals(name) && !"".equals(age) && !"".equals(email)
&& !"".equals(phone) && !"".equals(department_str)) {
        Resource student = model.createResource(uri + name);
        Property p1 = model.createProperty(uri, "has_name");
        Property p2 = model.createProperty(uri, "has_phone");
        Property p3 = model.createProperty(uri, "has_email");
        Property p4 = model.createProperty(uri, "has_age");
        Property p5 = model.createProperty(uri, "member_of");

        student.addProperty(RDF.type, model.createResource(uri +
"Student"));
        student.addProperty(p1, name);
        student.addProperty(p2, phone);
        student.addProperty(p3, email);
        student.addProperty(p4, age);
        student.addProperty(p5, model.createResource(uri +
department_str));
    }
}

```

```

        FileOutputStream out = null;
        OutputStreamWriter sw = null;
        try {
            out = new FileOutputStream(file, false);
            sw = new OutputStreamWriter(out, Charset.forName("UTF-
8").newEncoder());
        } catch (FileNotFoundException ex) {

Logger.getLogger(StudentData.class.getName()).log(Level.SEVERE, null,
ex);

        }
        RDFDataMgr.write(sw, model, Lang.RDFXML);

    } else {
        JOptionPane.showMessageDialog(this, "Κάποιο απο τα πεδία
δέν έχει τιμή.");
    }
}

```

Στην κλάση *LessonData*:

```

private void addToRdfActionPerformed(java.awt.event.ActionEvent evt)
{
    String name = this.less_name.getText();
    String teacher = this.taught_by.getText();

    if(!"".equals(name) && !"".equals(teacher)) {
        Resource lesson = model.createResource(uri+name);
        Property p1 = model.createProperty(uri, "les_name");
        Property p2 = model.createProperty(uri, "taught_by");

        lesson.addProperty(RDF.type, model.createResource(uri+"Lesson"));
        lesson.addProperty(p1, name);
        lesson.addProperty(p2, teacher);

        FileOutputStream out = null;
        OutputStreamWriter sw = null;
        try {
            out = new FileOutputStream(file, false);
            sw = new OutputStreamWriter(out, Charset.forName("UTF-
8").newEncoder());
        } catch (FileNotFoundException ex) {

Logger.getLogger(LessonData.class.getName()).log(Level.SEVERE, null,
ex);

        }
        RDFDataMgr.write(sw, model, Lang.RDFXML);

    } else {
        JOptionPane.showMessageDialog(this, "Κάποιο απο τα πεδία
δέν έχει τιμή.");
    }
}

```

Στην κλάση *ClassroomData*:

```
private void addToRdfActionPerformed(java.awt.event.ActionEvent evt)
{

    String name = this.class_name.getText();
    String cap = this.class_cap.getText();
    String department_str = this.class_dep.getText();

    if (!"".equals(name) && !"".equals(cap) &&
    !"".equals(department_str)) {
        Resource classroom = model.createResource(uri + name);
        Property p1 = model.createProperty(uri, "room_name");
        Property p2 = model.createProperty(uri, "room_capacity");
        Property p3 = model.createProperty(uri,
"room_department");

        classroom .addProperty(RDF.type, model.createResource(uri
+ "Classroom"));
        classroom .addProperty(p1, name);
        classroom .addProperty(p2, cap);
        classroom .addProperty(p3, model.createResource(uri +
department_str));

        FileOutputStream out = null;
        OutputStreamWriter sw = null;
        try {
            out = new FileOutputStream(file, false);
            sw = new OutputStreamWriter(out, Charset.forName("UTF-
8")).newEncoder();
        } catch (FileNotFoundException ex) {

Logger.getLogger(ClassroomData.class.getName()).log(Level.SEVERE,
null, ex);
        }
        RDFDataMgr.write(sw, model, Lang.RDFXML);

    } else {
        JOptionPane.showMessageDialog(this, "Κάποιο απο τα πεδία
δέν έχει τιμή.");
    }
}
```

Στην κλάση *DepartmentData*:

```
private void addToRdfActionPerformed(java.awt.event.ActionEvent evt)
{

    String name = this.dep_name.getText();
    String city = this.dep_city.getText();

    if (!"".equals(name) && !"".equals(city)) {
        Resource department = model.createResource(uri + name);
        Property p1 = model.createProperty(uri, "dep_name");
        Property p2 = model.createProperty(uri, "dep_city");

        department.addProperty(RDF.type, model.createResource(uri
+ "Department"));
        department.addProperty(p1, name);
    }
```

```

        department.addProperty(p2, city);

        FileOutputStream out = null;
        OutputStreamWriter sw = null;
        try {
            out = new FileOutputStream(file, false);
            sw = new OutputStreamWriter(out, Charset.forName("UTF-
8").newEncoder());
        } catch (FileNotFoundException ex) {

            Logger.getLogger(DepartmentData.class.getName()).log(Level.SEVERE,
            null, ex);
        }
        RDFDataMgr.write(sw, model, Lang.RDFXML);

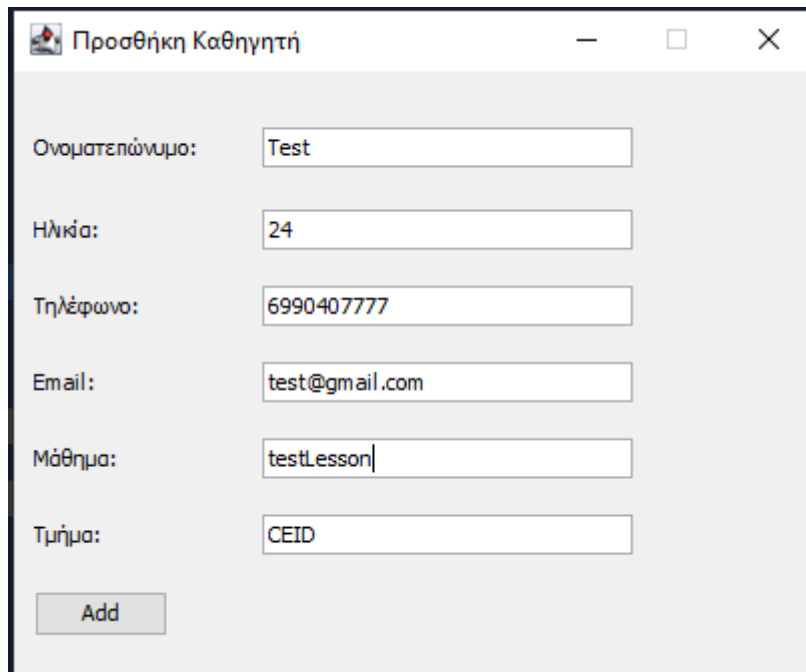
    } else {
        JOptionPane.showMessageDialog(this, "Κάποιο απο τα πεδία
δέν έχει τιμή.");
    }
    InputStream in = FileManager.get().open(file);
    InputStreamReader rin = new
InputStreamReader(in, Charset.forName("UTF-8").newDecoder());
    model.read(rin, "");

    String queryString = "PREFIX univ:
<http://www.mydomain.org/univ/> PREFIX xsd:
<http://www.w3.org/2001/XMLSchema#> PREFIX rdf:
<http://www.w3.org/1999/02/22-rdf-syntax-ns#> PREFIX rdfs:
<http://www.w3.org/2000/01/22-rdf-schema#> SELECT ?x {?x rdf:type
univ:Department .FILTER(?x=<" + uri + name + ">)}";
    Query query = QueryFactory.create(queryString);
    query.serialize(new IndentedWriter(System.out, true));
    QueryExecution qexec =
QueryExecutionFactory.create(query, model);
    ResultSet rs = qexec.execSelect();
    rs.hasNext();
    QuerySolution rb = rs.nextSolution();
    RDFNode x = rb.get("x");
    System.out.println(x.toString());
    tableModel.addRow(new Object[]{x});
}

```


Παρακάτω παρουσιάζονται screenshots από την εκτέλεση του Β. ερωτήματος της εφαρμογής .

Προσθήκη Καθηγητή:

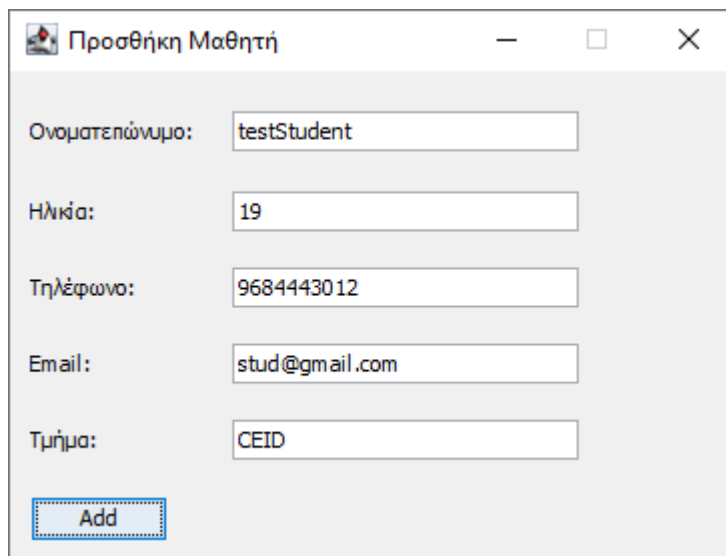


Όνοματεπώνυμο:	<input type="text" value="Test"/>
Ηλικία:	<input type="text" value="24"/>
Τηλέφωνο:	<input type="text" value="6990407777"/>
Email:	<input type="text" value="test@gmail.com"/>
Μάθημα:	<input type="text" value="testLesson"/>
Τμήμα:	<input type="text" value="CEID"/>
<input type="button" value="Add"/>	

Το entry στο .rdf αρχείο :

```
<univ:Professor rdf:about="http://www.mydomain.org/univ/Test">  
  <univ:member_of rdf:resource="http://www.mydomain.org/univ/CEID"/>  
  <univ:teaches rdf:resource="http://www.mydomain.org/univ/testLesson"/>  
  <univ:has_age>24</univ:has_age>  
  <univ:has_email>test@gmail.com</univ:has_email>  
  <univ:has_phone>6990407777</univ:has_phone>  
  <univ:has_name>Test</univ:has_name>  
</univ:Professor>
```

Προσθήκη Μαθητή:



Προσθήκη Μαθητή

Όνοματεπώνυμο: testStudent

Ηλικία: 19

Τηλέφωνο: 9684443012

Email: stud@gmail.com

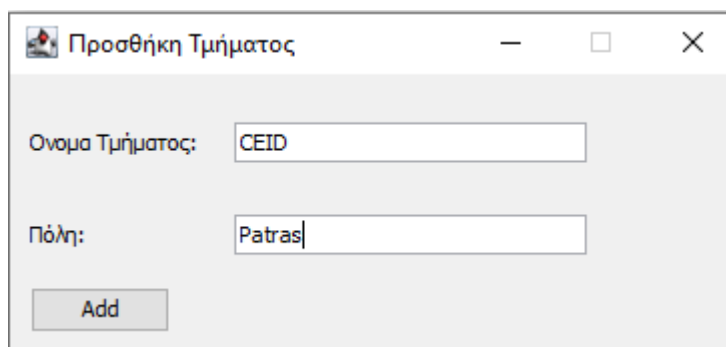
Τμήμα: CEID

Add

Το entry στο .rdf αρχείο :

```
<univ:Student rdf:about="http://www.mydomain.org/univ/testStudent">  
  <univ:member_of rdf:resource="http://www.mydomain.org/univ/CEID"/>  
  <univ:has_age>19</univ:has_age>  
  <univ:has_email>stud@gmail.com</univ:has_email>  
  <univ:has_phone>9684443012</univ:has_phone>  
  <univ:has_name>testStudent</univ:has_name>  
</univ:Student>
```

Προσθήκη Τμήματος:



Προσθήκη Τμήματος

Όνομα Τμήματος: CEID

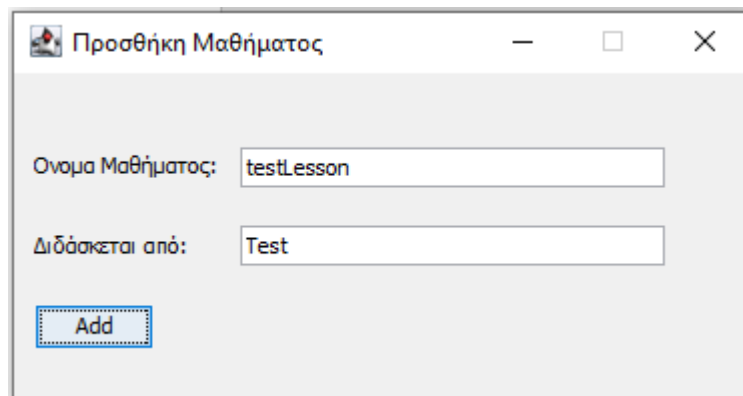
Πόλη: Patras

Add

Το entry στο .rdf αρχείο :

```
<univ:Department rdf:about="http://www.mydomain.org/univ/CEID">  
  <univ:dep_city>Patras</univ:dep_city>  
  <univ:dep_name>CEID</univ:dep_name>  
</univ:Department>
```

Προσθήκη Μαθήματος:



Προσθήκη Μαθήματος

Όνομα Μαθήματος: testLesson

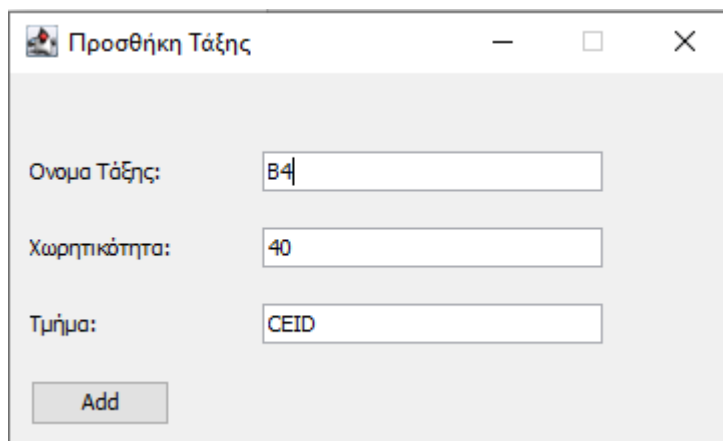
Διδάσκεται από: Test

Add

Το entry στο .rdf αρχείο :

```
<univ:Lesson rdf:about="http://www.mydomain.org/univ/testLesson">  
  <univ:taught_by>Test</univ:taught_by>  
  <univ:les_name>testLesson</univ:les_name>  
</univ:Lesson>
```

Προσθήκη Τάξης:



Προσθήκη Τάξης

Όνομα Τάξης: B4

Χωρητικότητα: 40

Τμήμα: CEID

Add

Το entry στο .rdf αρχείο :

```
<univ:Classroom rdf:about="http://www.mydomain.org/univ/B4">  
  <univ:room_department rdf:resource="http://www.mydomain.org/univ/CEID"/>  
  <univ:room_capacity>40</univ:room_capacity>  
  <univ:room_name>B4</univ:room_name>  
</univ:Classroom>
```

Αν επιλέξουμε έπειτα το τμήμα CEID από τον πίνακα των τμημάτων θα δούμε ότι :

The screenshot shows a window titled 'Τμήμα CEID' with four tables arranged in a 2x2 grid:

- Καθηγητές**:

Όνοματεπώνυμο	Ηλικία	Τηλέφωνο
Test	24	6990407777
- Τάξεις**:

Όνομα Αίθουσας	Χωριτικότητα
B4	40
- Μαθητές**:

Όνοματεπώνυμο	Ηλικία	Τηλέφωνο
testStudent	19	9684443012
- Μαθήματα**:

Όνομα Μαθήματος	Διδάσκων
testLesson	Test

Από το οποίο βλέπουμε ότι όλα τα δεδομένα που εισήγαγε ο χρήστης πιο πάνω εγγράφηκαν σωστά και όλα όσα υπάρχουν στους πίνακες αφορούν το τμήμα CEID .

Γ. Για το Γ. ερώτημα δημιουργήσα ένα `textField` στο οποίο ο χρήστης θα μπορεί να εισάγει το URI του resource που επιθυμεί και πατώντας το κουμπί `Show Statements` θα μπορεί να δει όλες τις τριπλέτες που υπάρχουν για αυτό στον γράφο . Πατώντας το παραπάνω κουμπί θα εκτελεστεί η μέθοδος **`stmt_btnActionPerformed`** η οποία παίρνει το κείμενο που έχει εισάγει ο χρήστης στο παραπάνω `textField` και έπειτα κοιτάει για το αν υπάρχει το URI μέσα στο κείμενο που έχει εισάγει ο χρήστης. Αν το κείμενο είναι κενό ή δεν περιέχει το URI τότε εμφανίζεται κατάλληλο προειδοποιητικό μήνυμα αλλιώς δημιουργώ ένα μοντέλο συμπερασμού με βάση το μοντέλο που δημιουργείται από το διάβασμα του `.rdf` αρχείου και μετά παίρνω όλους τους πόρους που περιέχει ο πόρος τον οποίον δήλωσα στο `textField` και για κάθε statement που περιέχει ο πόρος παίρνω τα ***Subject, Predicate και Object*** και τα γράφω στον πίνακα που υπάρχει στο παράθυρο τύπου `StatementData` μέσω της μεθόδου **`addToTable`** το οποίο παράθυρο το δημιουργώ αφού έχω εισάγει σωστά δεδομένα στο `textField` .

Ο κώδικας της μεθόδου **`stmt_btnActionPerformed`** βρίσκεται στην κλάση **`Ask2Frame`** και παρουσιάζεται παρακάτω :

```
private void stmt_btnActionPerformed(java.awt.event.ActionEvent evt)
{
    String text = uri_value.getText();
    Boolean contains = text.contains(URI);
    if(text.equals("") || !contains ){
        JOptionPane.showMessageDialog(this, "Λάθος URI.");
    }else{
        StatementData stmts = new StatementData();
        InfModel inf_model = ModelFactory.createRDFSModel(model);
```

```

Resource infr = inf_model.getResource(text);
StmtIterator iter = infr.listProperties();
while(iter.hasNext()) {
    Statement stmt = iter.nextStatement();
    System.out.print(" " + stmt.getSubject().toString());
    System.out.print(" " +
stmt.getPredicate().toString());
    System.out.print(" " + stmt.getObject().toString());
    System.out.print("\n");
    stmts.addToTable(stmt.getSubject().toString(),
stmt.getPredicate().toString(), stmt.getObject().toString());
}

stmts.setVisible(true);
}
}

```

Ο κώδικας της μεθόδου **addToTable** βρίσκεται στην κλάση **StatementData** και παρουσιάζεται παρακάτω :

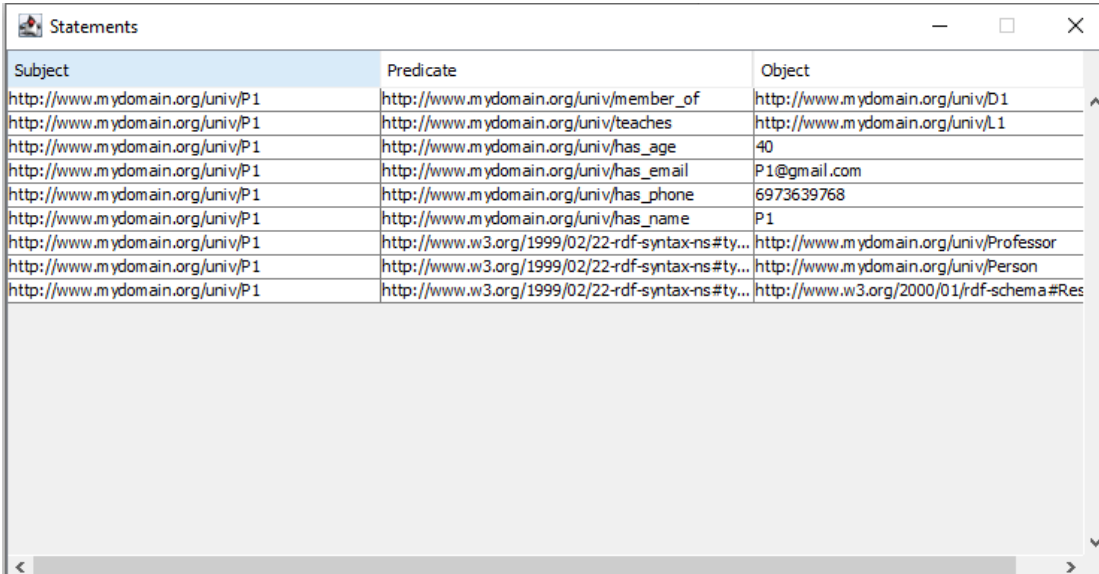
```

public void addToTable(String subject, String predicate,String
object) {
    DefaultTableModel tableModel =(DefaultTableModel)
stmtTable.getModel();
    tableModel.addRow(new Object[] {subject,predicate,object});
}

```

Παρακάτω παρουσιάζονται screenshots από την εκτέλεση του Γ. ερωτήματος της εφαρμογής .

Για URI: <http://www.mydomain.org/univ/P1>



Subject	Predicate	Object
http://www.mydomain.org/univ/P1	http://www.mydomain.org/univ/member_of	http://www.mydomain.org/univ/D1
http://www.mydomain.org/univ/P1	http://www.mydomain.org/univ/teaches	http://www.mydomain.org/univ/L1
http://www.mydomain.org/univ/P1	http://www.mydomain.org/univ/has_age	40
http://www.mydomain.org/univ/P1	http://www.mydomain.org/univ/has_email	P1@gmail.com
http://www.mydomain.org/univ/P1	http://www.mydomain.org/univ/has_phone	6973639768
http://www.mydomain.org/univ/P1	http://www.mydomain.org/univ/has_name	P1
http://www.mydomain.org/univ/P1	http://www.w3.org/1999/02/22-rdf-syntax-ns#ty...	http://www.mydomain.org/univ/Professor
http://www.mydomain.org/univ/P1	http://www.w3.org/1999/02/22-rdf-syntax-ns#ty...	http://www.mydomain.org/univ/Person
http://www.mydomain.org/univ/P1	http://www.w3.org/1999/02/22-rdf-syntax-ns#ty...	http://www.w3.org/2000/01/rdf-schema#Res

Οπού βλέπουμε ότι ο καθηγητής P1 εκτός από Professor είναι και Person .

Και για ένα μάθημα για παράδειγμα το <http://www.mydomain.org/univ/L1> έχουμε :

Statements		
Subject	Predicate	Object
http://www.mydomain.org/univ/L1	http://www.mydomain.org/univ/taught_by	http://www.mydomain.org/univ/P1
http://www.mydomain.org/univ/L1	http://www.mydomain.org/univ/les_name	L1
http://www.mydomain.org/univ/L1	http://www.w3.org/1999/02/22-rdf-syntax-ns#ty...	http://www.mydomain.org/univ/Lesson
http://www.mydomain.org/univ/L1	http://www.w3.org/1999/02/22-rdf-syntax-ns#ty...	http://www.w3.org/2000/01/rdf-schema#Res

Όπου και εδώ βλέπουμε ότι βλέπει ότι το <http://www.mydomain.org/univ/L1> είναι τύπου Lesson παρόλο που κάτι τέτοιο δεν έχει δηλωθεί στο .rdf αρχείο .