KNOWLEGDE BASED SYSTEMS

(CSC 758)

**TERM PROJECT PAPER ON**

# AN ONTOLOGICAL-BASED APPROACH TO KNOWLEDGE REPRESENTATION AND REASONING IN AI FOR UNIVERSITY DOMAIN

(DOMAIN: UNIVERSITY)

BY

**ADEYEMI ADEDOYIN SIMEON**

**(209188)**

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF

MASTER OF SCIENCE (MSC.) IN COMPUTER SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

FACULTY OF SCIENCES

UNIVERSITY OF IBADAN, NIGERIA

AUGUST, 2018

## *ABSTRACT*

*Recent works in Artificial Intelligence (AI) is exploring the use of formal ontologies as a way of specifying content-specific agreements for the sharing and reuse of knowledge among software entities and experts in a given domain (Thomas, 1995). Genesereth and Nilsson (1987) described Ontology as a body of formally represented knowledge based on conceptualization of objects, entities or concepts that are assumed to exist in some area of interest and the relationships that hold among them.*

*In this paper, a university Ontology was developed that describes the various concepts within a university educational system domain, using Knowledge-Engineering Process. The axioms were represented in First-Order Predicate Logic and the ontology was developed using Protégé 5.2.0 and implemented using Prolog Programming Language (SWI-Prolog IDE). And the result obtained from querying the Knowledge-base was able to answer appropriate competency questions such as who teaches what course? which degree is a student enrolled for? who is the head of what department? in which faculty is a particular department? what are the list of available degree programs offered by a department? etc.*

***Keywords:*** *Ontology, Knowledge representation, domain, axioms, university.*

## I. INTRODUCTION

The connection between Artificial Intelligence (AI) and Knowledge has become too tightly coupled that one can hardly be mentioned without the other. The application of previous knowledge in the display of intelligent behavior cannot be over-emphasized. Although this connection has received a lot of debate from experts from different schools of thought. I agree however that there is a major interface between knowledge and Intelligence. Since Knowledge plays a major role in reasoning, how that knowledge is represented is of importance, especially to make it acceptable and usable to a machine in an unambiguous manner. A useful knowledge must not only be represented but in an explicit and generally agreed form among software agents and experts who would use it. Hence, the concept of Ontology. Ontology is a term borrowed from Philosophy meaning a systematic account of Existence. In Artificial Intelligence, the term "exists" implies that which can be represented.

"Ontology is a body of formally represented knowledge based on a conceptualization: the objects, concepts, and other entities that are assumed to exist in some area of interest and the relationships that hold among" (Thomas, 1995). It is an explicit specification of conceptualization of real world.

An Ontology encompasses a representation, formal naming and definition of the categories, properties, and relations between the concepts, data, and entities that substantiate one, many, or all domains (Ifnord, 2018).

A more specific definition of Ontology is one by Natalya and Deborah (2001) as a formal explicit description of concepts in a domain of discourse (**classes** (sometimes called **concepts))**, properties of each concept describing various features and attributes of the concept (**slots** (sometimes called **roles** or **properties**)), and restrictions on slots (**facets** (sometimes called **role restrictions**)).

When the knowledge of a domain is represented in a declarative formalism, the set of objects that can be represented is called the universe of discourse. This set of objects, and the describable relationships among them, are reflected in the representational vocabulary with which a knowledge-based program represents knowledge.

This paper presents the development and implementation of a University ontology to allow formalization and sharing of common concepts and relationships (among concepts) in the university domain among staff, students, applicants, researchers, software agents and other users of the university knowledge. And allow several new information to be inferred from available knowledge by querying the knowledge base. The ontology represented knowled

## II. LITERATURE REVIEW

Natalya and Deborah (2001) postulated three fundamental ontology design rules to help designers make better design decisions as follow: There is no correct way to model a domain, Ontology development is necessarily an iterative process, and Concepts in the ontology should be close to objects (physical or logical) and relationships in your domain of interest. These are most likely to be nouns (objects) or verbs (relationships) in sentences that describe your domain. They also proposed a Simple Knowledge Engineering Methodology for developing an ontology.

Thomas (1995) also supports of engineering perspective to ontology development. In his work, he described the roles of ontology in supporting knowledge sharing activities and then presented a set of criteria to guide the development of ontologies for this purpose. He designed an ontology for physical quantities in the domain of Engineering Mathematics as a case study.

Chimaera (McGuinness et al. 2000) provides diagnostic tools for analyzing ontologies. The analysis that Chimaera performs includes both a check for logical correctness of an ontology and diagnostics of common ontology-design errors. An ontology designer may want to run Chimaera diagnostics over the evolving ontology to determine the conformance to common ontology-modeling practices.

Several researchers have also created diverse ontology for different domains. Among these is An ontology for User Profile developed (Maria, *et al.*, 2007). Ayesha, Khaleel and Padmaja (2012) also created a course ontology for education domain which contains complete details of various courses to guide students in selecting a future course on the basis of their existing qualifications. Akande, Akinkunmi and Ayorinde (2015) also developed an Ontoloy to represent the production activities of drug tablet using both Allen interval temporal logic and McDermott's logic of time points. The represented the axioms in FOL and implemented it using Prolog Programming Language.

Nicola (1998) also corroborated the idea of Natalya and Deborah (2001) on the fact that there is no definite and generally acceptable methodology for the development of Ontology, as such each ontology present its own methodological and architectural peculiarities. He emphasized the adoption of interdisciplinary approach as the peculiarity of the Methodological side and the centrality of the role played in an information system as the peculiarity of the Architectural side.

## III. METHODOLOGY (KNOWLEDGE ENGINEERING PROCESS)

A University is an organized formal institution of learning that offers various courses to students on the basis of their qualification, after which a certificate is awarded to successful students and such institution allows research to be conducted in an organized and standard form. The University ontology was developed using Natalya and Deborah (2018) Simple Knowledge-Engineering Methodological approach. This is an iterative process of ontology development whose approach is based on starting with a rough pass and then iteratively revising and refining the evolving ontology. This method follows a logical sequence of iterative activities as highlighted below in the consequent sub-sections.

The development of this ontology followed the Knowledge Engineering process. This comprises of a number of stages made up of a number of deliverables. These stages as shown in Figure 1:

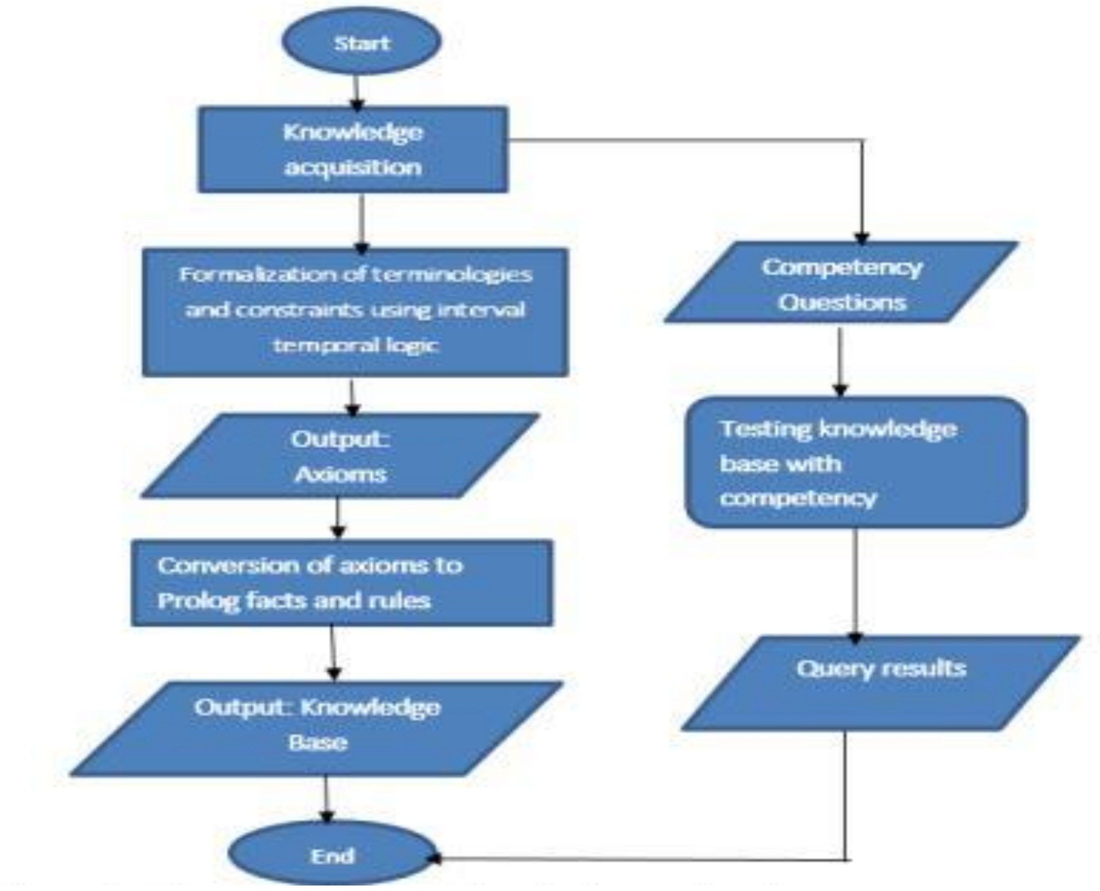

Start
Knowledge acquisition
Formalization of terminologies and constraints using interval temporal logic
Output: Axioms
Conversion of axioms to Prolog facts and rules
Output: Knowledge Base
End
Competency Questions
Testing knowledge base with competency
Query results

Figure 1: Knowledge Engineering Process *(Akande, Akinkunmi & Ayorinde, 2015).*

The process involves the following major stages:

- Knowledge acquisition: This involves gathering relevant concepts and arranging these into hierarchies. The deliverables are the hierarchy of concepts and competency questions.
- Formalization: This entails the representation of the knowledge about the all aspects of the domain as the scope allows in a formal language that machine can understand or in a form that can be easily converted into machine understandable form. Since the whole essence is to make machine use available knowledge about a given domain to carry-out intelligent behavior and infer new information. The deliverable is collection of axioms.
- Implementation: This stage involves the conversion of axioms into actual machine-understandable instructions. Prolog Programming Language is used for implementation of the Knowledge base because of its simplicity, expressive capability, scalability, availability and wide acceptance.

## IV. DEVELOPMENT OF THE UNIVERSITY ONTOLOGY

The university ontology was developed using Protégé 5.2.0 using the Knowledge Engineering Process Methodology.

### KNOWLEDGE ACQUISITION

This stage involves all activities of gathering relevant knowledge of the domain of discourse. It involves identifying concepts also called “objects” or “classes” of the real world under study and relationships existing among these concepts. Of major importance in this stage is the classification of objects (**concepts** or **classes**) into hierarchy, (separating concepts into parent and child) and the description of features or properties of each concepts (**slots / roles / properties**) as well as constraints on them (**facets**, also called **role restrictions**).

1. **Domain Description**

   The university ontology is described in terms of its academic activities as it relates students to lecturers. As it is obviously almost impossible to fully and completely conceptualize and represent all the complex activities of a standard university in a single project as this. It will be impractical for the researcher to attempt such complex conceptualization. As such, the scope of this ontology is defined by the following questions, who are the intended users? (staff and students excluding administrative staff without direct influence on academic), for what purpose? (academics), what questions should it provide answers to (Competency questions)? (academic and performance related questions). In view of the above, the scope of the concepts covered in this ontology is a university described in view of its academic activities and relationship among constituent parts (that have direct influence on academics).

The table below describes what we see a university to be.

TABLE 1: Description of a university Domain

| S/No | Description |
|---|---|
| 1. | University is composed of faculties |
| 2. | Faculty is made up of departments |
| 3. | Department have staff |
| 4. | Department have students |
| 5. | Department offers degree |
| 6. | Department offers courses |
| 7. | Staff can be an academic, non-academic or contract staff. |
| 8. | A student can be classified as either undergraduate or postgraduate |
| 9. | Degree can be classified as BSc, PGD, MSc, MPhil and Phd |
| 10. | Academic staff and contract staff each teaches some course(s) |
| 11. | Undergraduate student registers for at least one course and Postgraduate students may register for a course. |
| 12. | A Student must be at least 18 years of age. |
| 13. | A Student can only enroll for one type of degree program at a time. |

## Competency Questions

Competency questions are common sense questions that the knowledge base should be able to answer to prove its competency. Below are some competency questions that the ontology is expected to provide answers to.

- What department belong to which faculty?
- What are the available faculties in a university?
- What degree is a student enrolled for?
- What course(s) is a particular academic staff taking?
- Does a particular student register for a particular course?
- What are the courses offered in a department?
- What are the available departments in a university or faculty?
- What is the score of a student in a particular course?
- What are the courses that a student registered for?
- In which department does a student or staff belong to?
- Who are the staff in a particular department, faculty or university?
- What degree programs are available in a university, faculty or department?
- Etc.

## 2. Identified Concepts and Creation of Class Hierarchy

From the description of a university as displayed in Table 1, the following hierarchy of concepts have been created using Protégé 5.2.0 as shown in Figure 2 below.

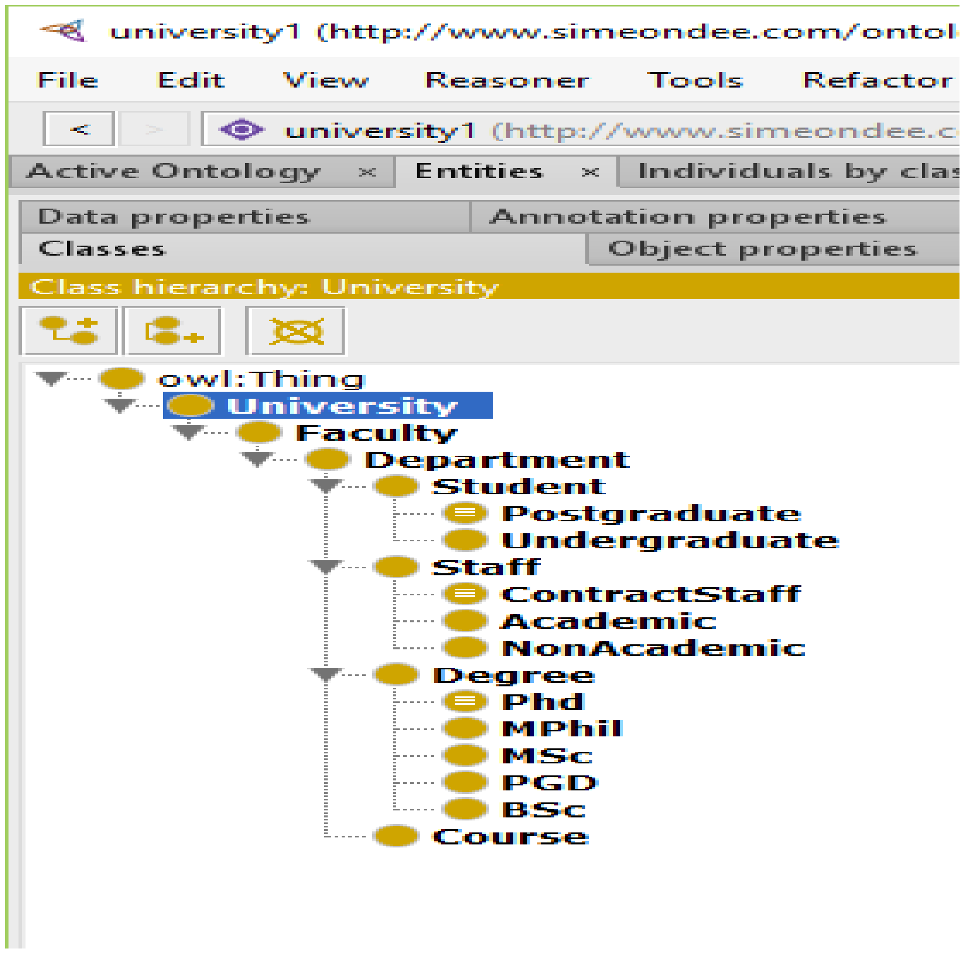


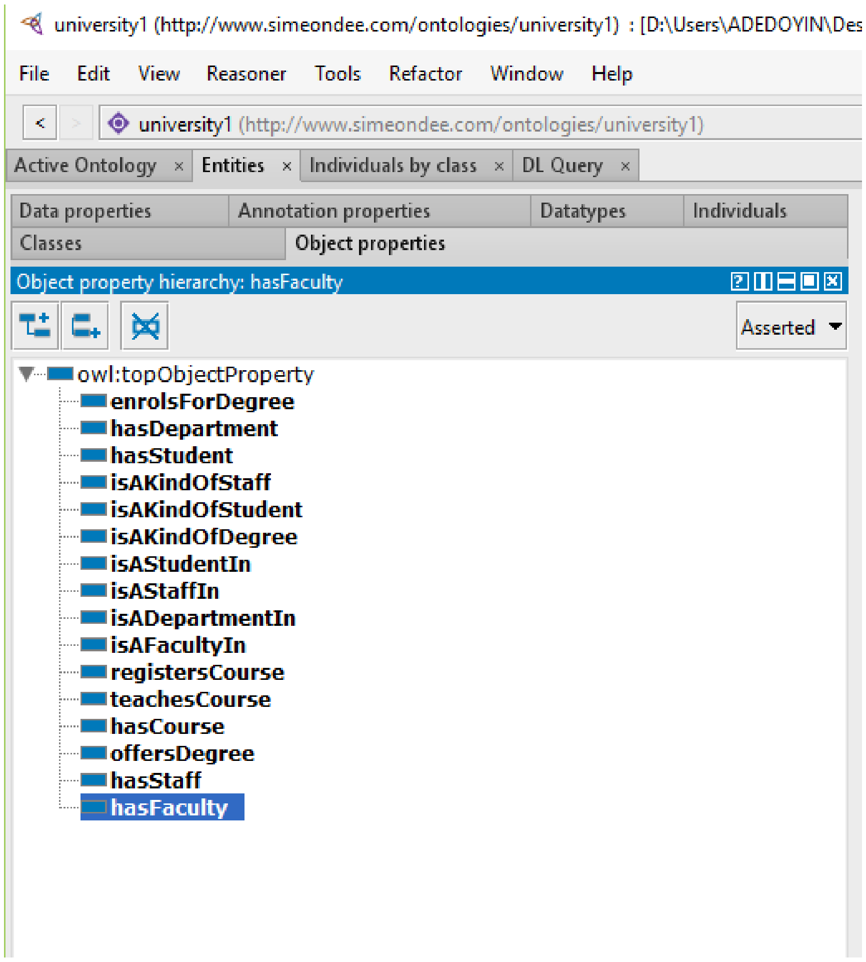
Figure 2. Class Hierarchy for the University Ontology Designed using Protégé 5.2.0

## 3. Creation of Class Properties

Properties defines the relationships that exists among components. We have two types of properties in OWL: O*bject properties* and *Data type properties*. Using Protégé, there is also another type of property called *Annotation property*. Object properties define the relationships between instances of objects. Data type properties define relationship between instances and data value while Annotation properties can be used to add metadata or additional information to classes’ individuals and objects.

## Object Properties

In the development of this ontology, I defined a number of object properties that describe the relationships among the various classes shown in Figure 2. These properties are displayed in Figure 3 below and the domain and range of each property are tabulated in Table 2 that follows.

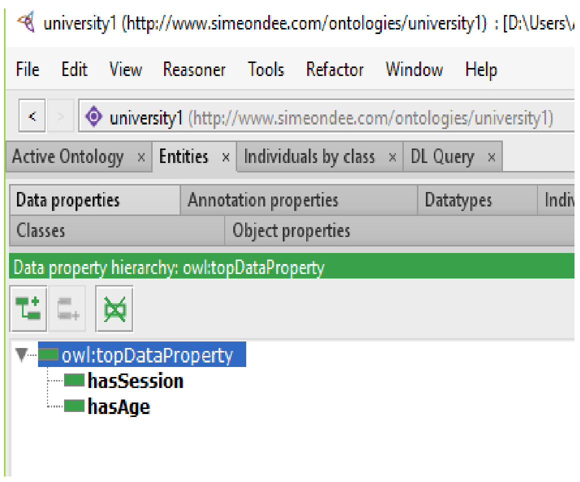

university1 (http://www.simeondee.com/ontologies/university1) : [D:\Users\ADEDOYIN\Des
File
Edit
View
Reasoner
Tools
Refactor
Window
Help
university1 (http://www.simeondee.com/ontologies/university1)
Active Ontology
Entities
Individuals by class
DL Query
Data properties
Annotation properties
Datatypes
Individuals
Classes
Object properties
Object property hierarchy: hasFaculty
Asserted

```
owl:topObjectProperty
    enrolsForDegree
    hasDepartment
    hasStudent
    isAKindOfStaff
    isAKindOfStudent
    isAKindOfDegree
    isAStudentIn
    isAStaffIn
    isADepartmentIn
    isAFacultyIn
    registersCourse
    teachesCourse
    hasCourse
    offersDegree
    hasStaff
    hasFaculty
```

Figure 3. List of Defined Object Properties Describing Relationships among Classes.

## Data type Properties

Data type properties defines relationships between instances of a class and data values. From the domain description, age is the only data property that can be related to a student. The property and its full definition is as shown in Figure 4 and Figure 5 below.


university1 (http://www.simeondee.com/ontologies/university1) : [D:\Users\

File Edit View Reasoner Tools Refactor Window Help

university1 (http://www.simeondee.com/ontologies/university1)

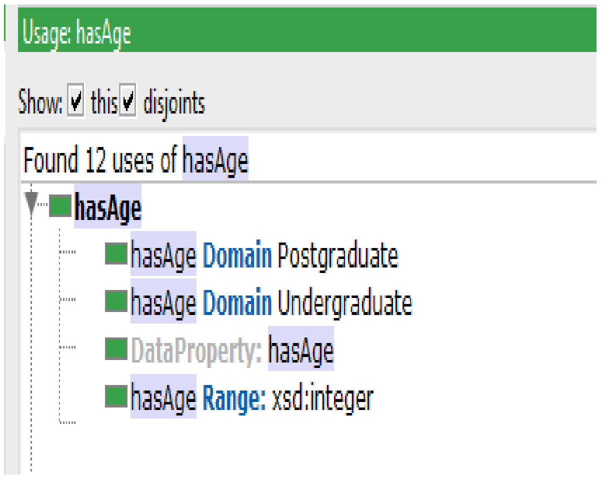
Active Ontology × Entities × Individuals by class × DL Query ×

Data properties | Annotation properties | Datatypes | Indiv

Classes | Object properties

Data property hierarchy: owl:topDataProperty

owl:topDataProperty
- hasSession
- hasAge


Usage: hasAge

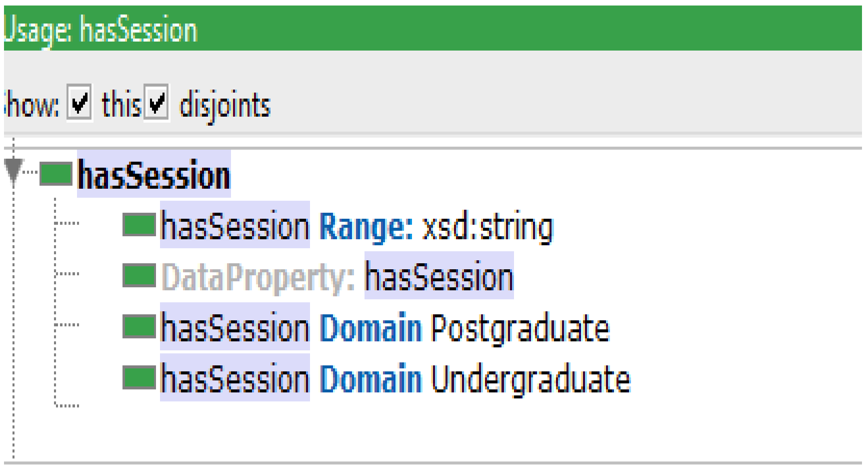
Show: ✓ this ✓ disjoints

Found 12 uses of hasAge

hasAge
- hasAge Domain Postgraduate
- hasAge Domain Undergraduate
- DataProperty: hasAge
- hasAge Range: xsd:integer

Figure 4. Data Type Property

Figure 5. hasAge Type Property Definition (a)


Usage: hasSession

Show: ✓ this ✓ disjoints

hasSession
- hasSession Range: xsd:string
- DataProperty: hasSession
- hasSession Domain Postgraduate
- hasSession Domain Undergraduate

Figure 6. hasSession Data Type Property Definition (b)

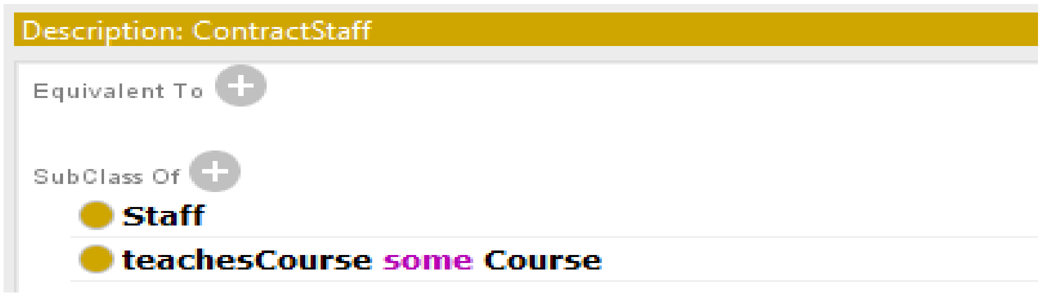
**Property Restrictions / Constraints**

These define constraints or restrictions placed on classes, their relationships or instance values. There are three categories of restrictions (constraints) in OWL Protégé namely Qantifier Restrictions (further subdivided into Existential (some) Restrictions and Universal (only) Restrictions), Cardinality Restrictions (Describing the minimum and maximum number of instances of class A that can participate in a relationship (property) with class B), and Value Restrictions (constraints placed on acceptable value for an attribute of the instance of a class).

In the University ontology, from the domain description in TABLE 1, the following restrictions can be identified:

- Academic Staff and Contract Staff each teaches some (at least a) course. ***Existential Restriction.**
- Undergraduate student registers for at least one course (Minimum of one course). ***Cardinality Restriction**.
- Postgraduate student **may** register for a course (Minimum of zero course). This is to take Phd / research students into consideration who may not need to register for courses. ***Cardinality Restriction**.
- A student must be at least 18 years of age (value of age for all instances of student class must have a value greater than or equal to 18. ***Value Restriction.**
- A student can only register for one type of degree at a time (at some session). ***Value Restriction** on Session and ***Cardinality Restriction** on Degree.

The figures below show the implementations of these constraints in Protégé.


Description: ContractStaff

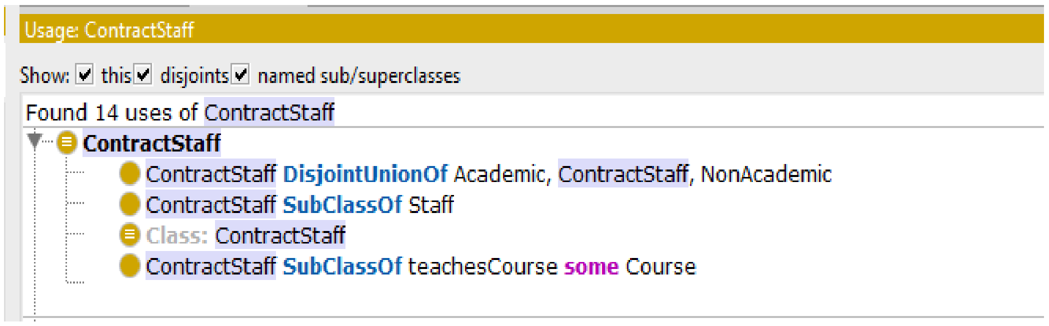
Equivalent To

SubClass Of

Staff

teachesCourse some Course

Figure 7. Contract Staff Restriction (Existential implemented as “some”).


Usage: ContractStaff

Show: ✓ this ✓ disjoints ✓ named sub/superclasses

Found 14 uses of ContractStaff

ContractStaff

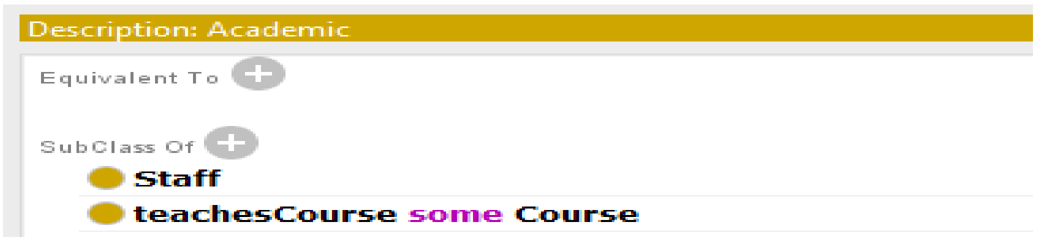
ContractStaff DisjointUnionOf Academic, ContractStaff, NonAcademic

ContractStaff SubClassOf Staff

Class: ContractStaff

ContractStaff SubClassOf teachesCourse some Course

Figure 8. Contract Staff Full Restriction Description (Existential).



Description: Academic

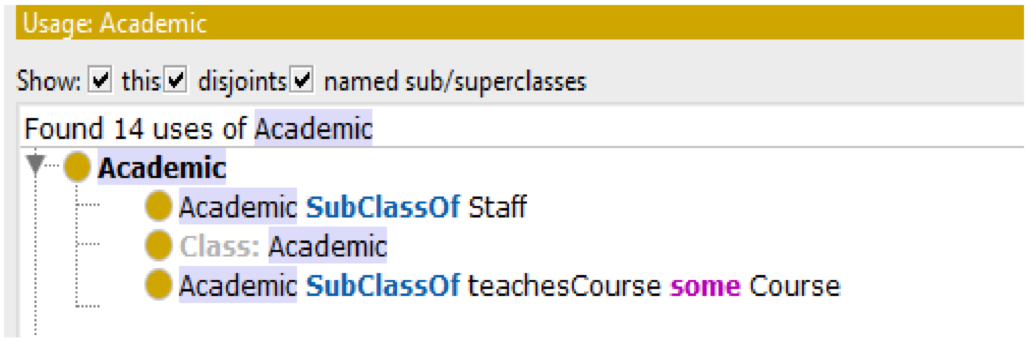
Equivalent To

SubClass Of

Staff

teachesCourse some Course

Figure 8. Academic Staff Restriction (Existential implemented as “some”).



Usage: Academic

Show: ✓ this ✓ disjoints ✓ named sub/superclasses

Found 14 uses of Academic

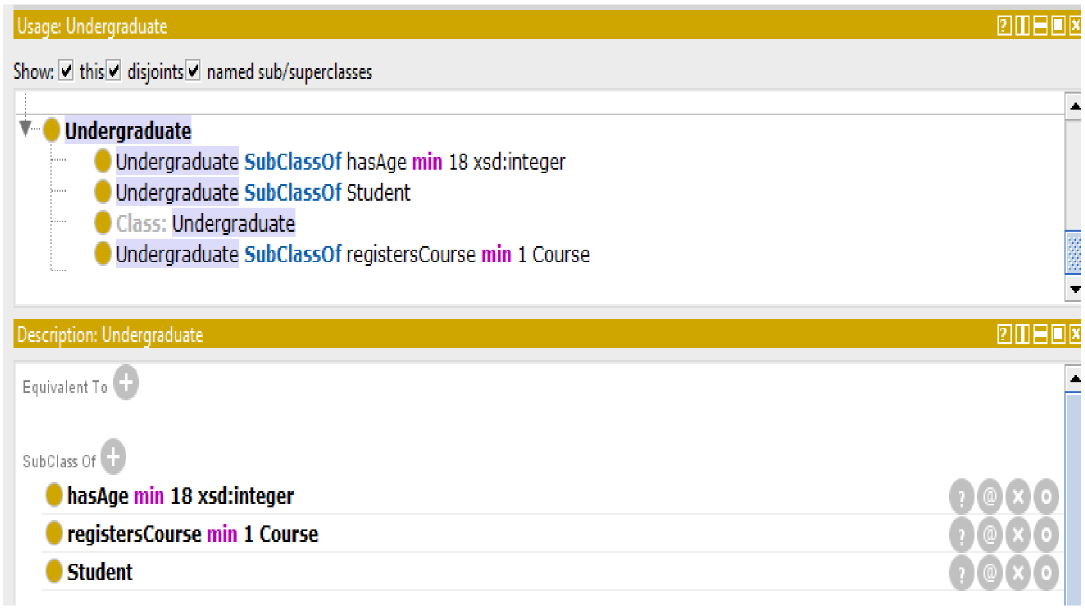
Academic

Academic SubClassOf Staff

Class: Academic

Academic SubClassOf teachesCourse some Course

Figure 9. Academic Staff Full Restriction Description (Existential [some]).



Usage: Undergraduate

Show: ✓ this ✓ disjoints ✓ named sub/superclasses

Undergraduate

Undergraduate SubClassOf hasAge min 18 xsd:integer

Undergraduate SubClassOf Student

Class: Undergraduate

Undergraduate SubClassOf registersCourse min 1 Course

Description: Undergraduate

Equivalent To

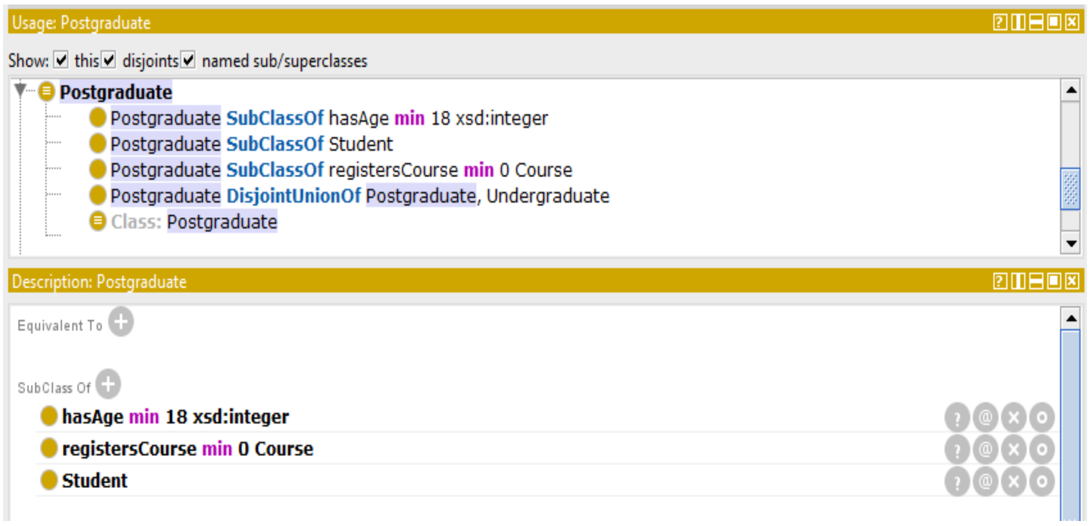
SubClass Of

hasAge min 18 xsd:integer

registersCourse min 1 Course

Student

Figure 10. Undergraduate Student Full Restriction Description (Cardinality [min 1] Restriction on Course and Value Restriction on Age).


Usage: Postgraduate

Show: ✓ this ✓ disjoints ✓ named sub/superclasses

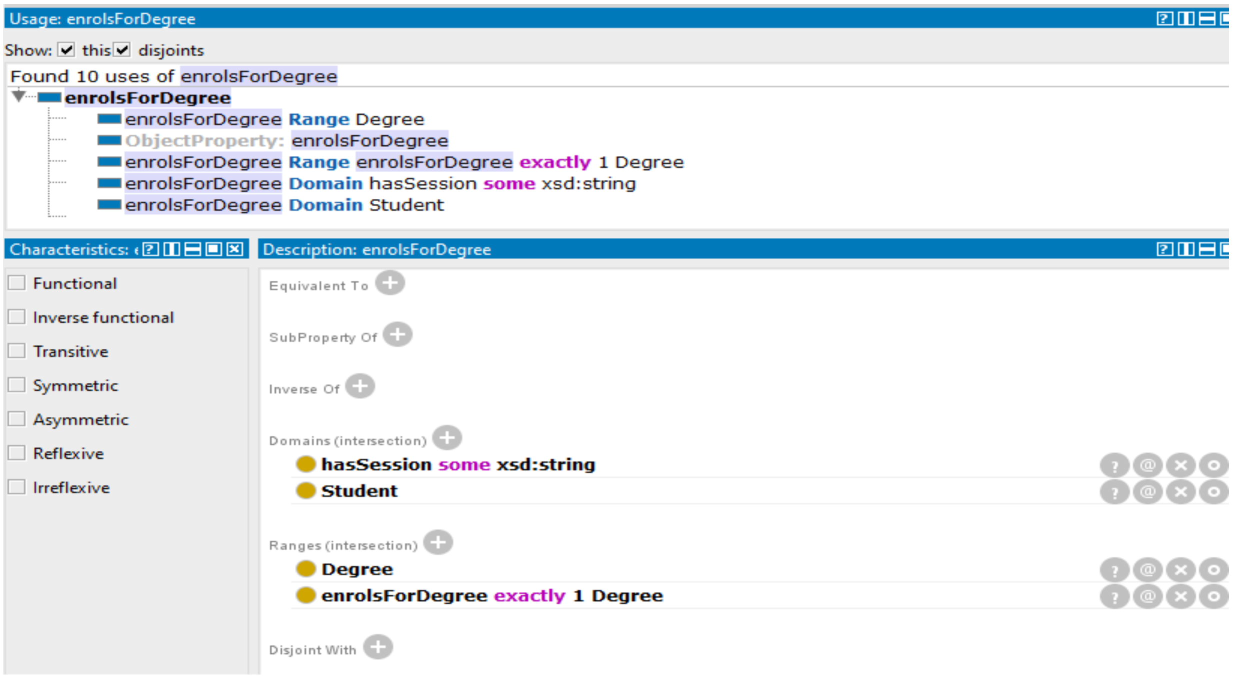
Postgraduate
- Postgraduate SubClassOf hasAge min 18 xsd:integer
- Postgraduate SubClassOf Student
- Postgraduate SubClassOf registersCourse min 0 Course
- Postgraduate DisjointUnionOf Postgraduate, Undergraduate
- Class: Postgraduate

Description: Postgraduate

Equivalent To

SubClass Of
- hasAge min 18 xsd:integer
- registersCourse min 0 Course
- Student

Figure 11. Postgraduate Student Full Restriction Description (Cardinality [min 0] Restriction on Course and Value Restriction on Age).


Usage: enrolsForDegree

Show: ✓ this ✓ disjoints

Found 10 uses of enrolsForDegree

enrolsForDegree
- enrolsForDegree Range Degree
- ObjectProperty: enrolsForDegree
- enrolsForDegree Range enrolsForDegree exactly 1 Degree
- enrolsForDegree Domain hasSession some xsd:string
- enrolsForDegree Domain Student

Characteristics:
- Functional
- Inverse functional
- Transitive
- Symmetric
- Asymmetric
- Reflexive
- Irreflexive

Description: enrolsForDegree

Equivalent To

SubProperty Of

Inverse Of

Domains (intersection)
- hasSession some xsd:string
- Student

Ranges (intersection)
- Degree
- enrolsForDegree exactly 1 Degree

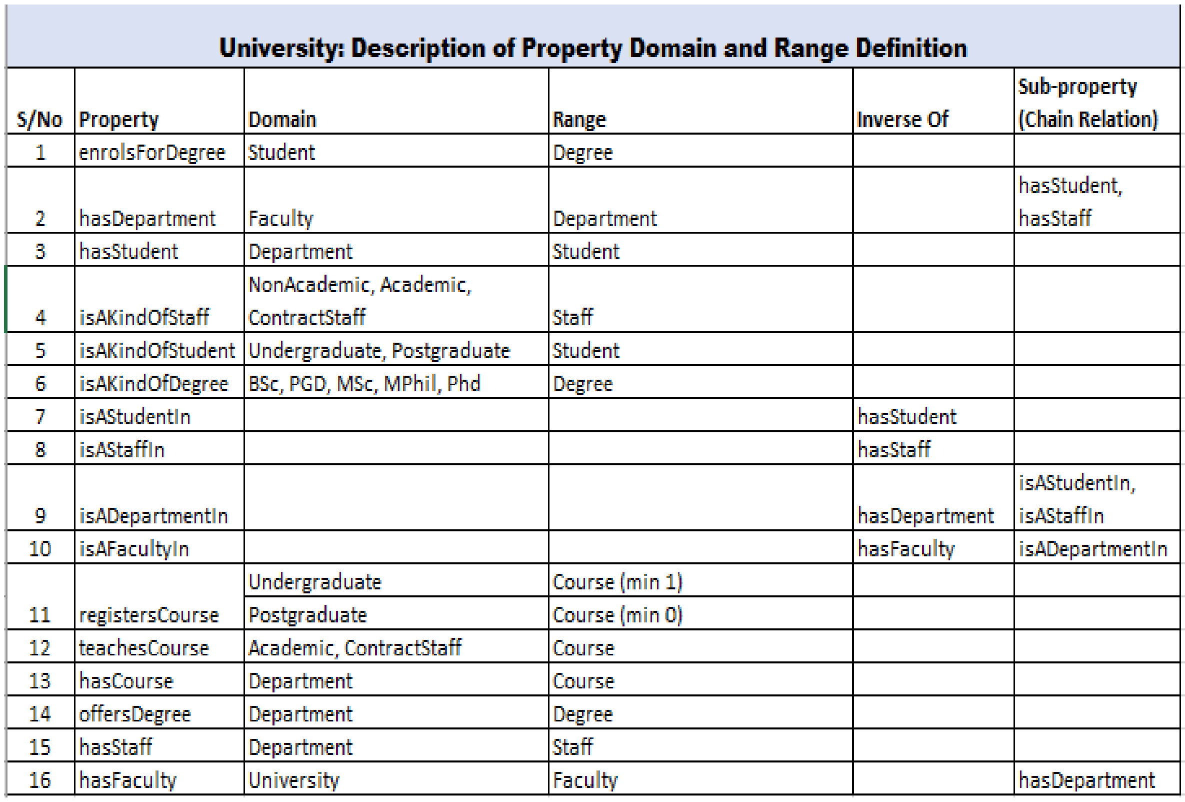
Disjoint With

Figure 12. A student can only register for exactly one Degree Program at some particular Session (Existential [some] Restriction on Value "hasSession" and Cardinality [exactly one] Restriction on class "Degree").

## 4. Property Domain and Range Definition

The domain of a property (relation) is the class having that relation with another class (actor) while the “range” of a property is the class to which a property is related (receptor). A property may be defined to be an inverse of another existing property .i.e. the relationship in opposite direction, in which case the domain becomes the range and vice-versa. The Sub-property (Chain relation) in the table below defined a chain relationship among hierarchy of classes. E.g. if a student belongs to a department and a department belongs to a Faculty and a faculty belongs to a university (chain relationship), then it can be inferred that the student belongs to the faculty and university.

TABLE 2. Description of Property Domain and Range Definition.



**University: Description of Property Domain and Range Definition**

| S/No | Property | Domain | Range | Inverse Of | Sub-property (Chain Relation) |
|---|---|---|---|---|---|
| 1 | enrolsForDegree | Student | Degree | | |
| 2 | hasDepartment | Faculty | Department | | hasStudent, hasStaff |
| 3 | hasStudent | Department | Student | | |
| 4 | isAKindOfStaff | NonAcademic, Academic, ContractStaff | Staff | | |
| 5 | isAKindOfStudent | Undergraduate, Postgraduate | Student | | |
| 6 | isAKindOfDegree | BSc, PGD, MSc, MPhil, Phd | Degree | | |
| 7 | isAStudentIn | | | hasStudent | |
| 8 | isAStaffIn | | | hasStaff | |
| 9 | isADepartmentIn | | | hasDepartment | isAStudentIn, isAStaffIn |
| 10 | isAFacultyIn | | | hasFaculty | isADepartmentIn |
| 11 | registersCourse | Undergraduate | Course (min 1) | | |
| | | Postgraduate | Course (min 0) | | |
| 12 | teachesCourse | Academic, ContractStaff | Course | | |
| 13 | hasCourse | Department | Course | | |
| 14 | offersDegree | Department | Degree | | |
| 15 | hasStaff | Department | Staff | | |
| 16 | hasFaculty | University | Faculty | | hasDepartment |

## V. FORMALIZATION OF THE CONCEPTS

The limitation of natural language in knowledge representation for human and machine-use is no longer a debate. Natural language is full of so many ambiguities and inconsistencies. This limitation requires that knowledge be represented in a more formal language for it to be usable and understandable by a machine, in order to display intelligent behavior by making inferences from this knowledge. One such formal language used in this paper is First Order Predicate Logic (FOL). The choice of this language is due to its ability to represent all forms of knowledge.

From the domain description in TABLE 1, The following axioms have been defined and the knowledge represented in First Order Predicate Logic (FOL):

1. University is composed of faculties (All universities have at least a (some) faculty).

   $\forall x{:}\ \exists y{:}\ [\text{university}(x) \land \text{faculty}(y) \rightarrow \text{hasFaculty}(x,y)].$

2. Faculty is made up of departments

   $\forall x{:}\ \exists y{:}\ [\text{faculty}(x) \land \text{department}(y) \rightarrow \text{hasDepartment}(x,y)].$

3. Department have staff (All departments have some Staff).

   $\forall x{:}\ \exists y{:}\ [\text{department}(x) \land \text{staff}(y) \rightarrow \text{hasStaff}(x,y)].$

4. Department have students (All departments have some student).

   $\forall x{:}\ \exists y{:}\ [\text{department}(x) \land \text{student}(y) \rightarrow \text{hasStudent}(x,y)].$

5. Department offers degree (All departments offers some degree).

   $\forall x{:}\ \exists y{:}\ [\text{department}(x) \land \text{degree}(y) \rightarrow \text{offersDegree}(x,y)].$

6. Department offers courses (All departments offer some course(s)).

   $\forall x{:}\ \exists y{:}\ [\text{department}(x) \land \text{course}(y) \rightarrow \text{hasCourse}(x,y)].$

7. Staff can be an academic, non-academic or contract staff (If you are a staff, then you either an academic, nonacademic or a contract staff).

   $\forall x$: [staff(x) $\rightarrow$ academic(x) $\vee$ nonAcademic(x) $\vee$ contractStaff(x)].

8. A student can be classified as either undergraduate or postgraduate (If you are a student, then you are either an undergraduate or a postgraduate student).

   $\forall x$: [student(x) $\rightarrow$ undergraduate(x) $\vee$ postgraduate(x)].

9. Degree can be classified as BSc, PGD, MSc, MPhil and Phd.

   $\forall x$: [degree(x) $\rightarrow$ bsc(x) $\vee$ pgd(x) $\vee$ msc(x) $\vee$ mphil(x) $\vee$ phd(x)].

10. Academic staff and contract staff each teaches some course(s).

    $\forall x$: [academic(x) $\vee$ contractStaff(x) $\rightarrow$ $\exists y$: teachesCourse(x,y)].

11. Undergraduate student registers for at least one course and Postgraduate students may register for a course.

    $\forall x$: $\exists y$: [undergraduate(x) $\wedge$ course(y) $\rightarrow$ registersCourse(x,y)].

    $\exists x$: $\exists y$: [postgraduate(x) $\wedge$ course(y) $\rightarrow$ registersCourse(x,y)].

12. A Student must be at least 18 years of age.

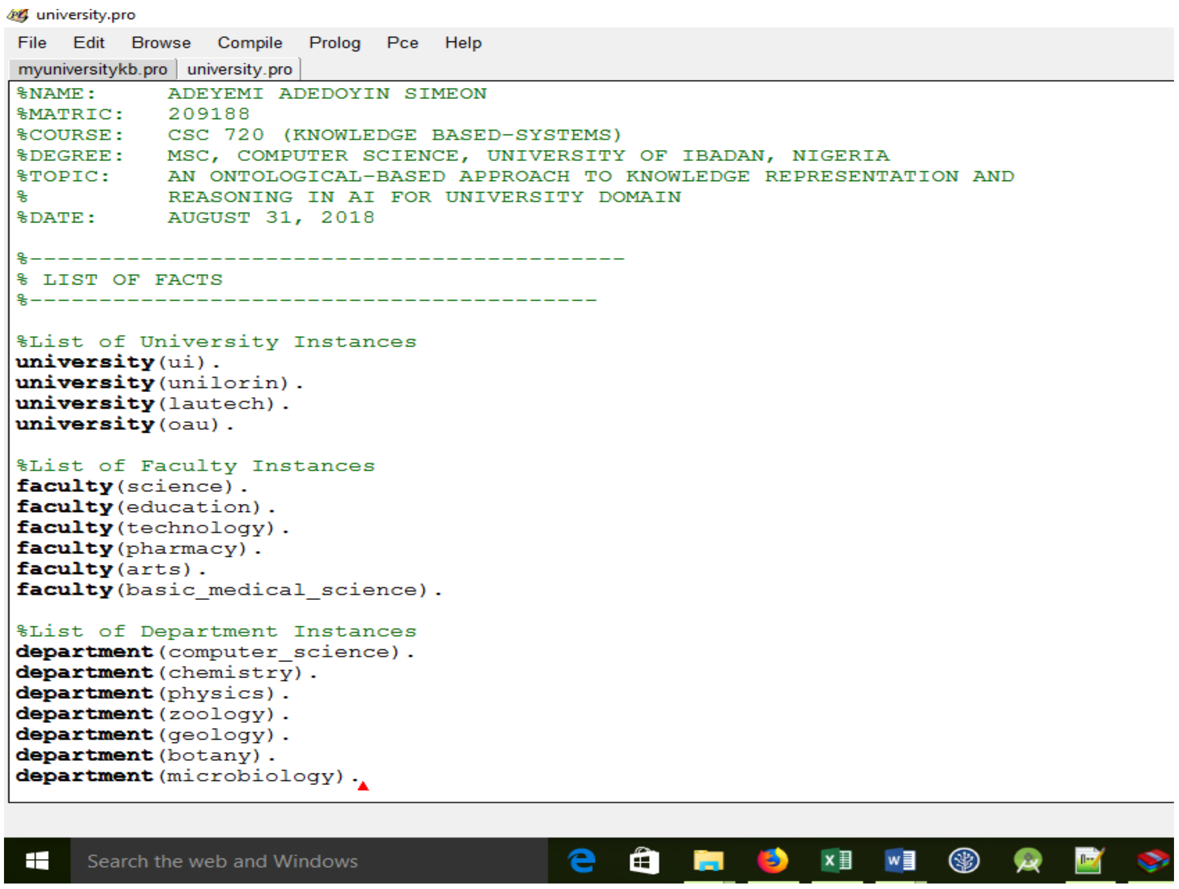
    ge(X,Y) $\leftrightarrow$ X >= 18.

    $\forall x$: [student(x) $\rightarrow$ hasAge(x,y) $\wedge$ ge(y,18)].

13. A Student can only enroll for one type of degree program at a time
    $\forall x$: $\exists y$: [student(x) $\wedge$ degree(y) $\rightarrow$ $\exists z$: [session(z) $\wedge$ enrolsForDegree(x,y) $\wedge$ enrolForMax(x,1,y) $\wedge$ hasSession(x,z)]].

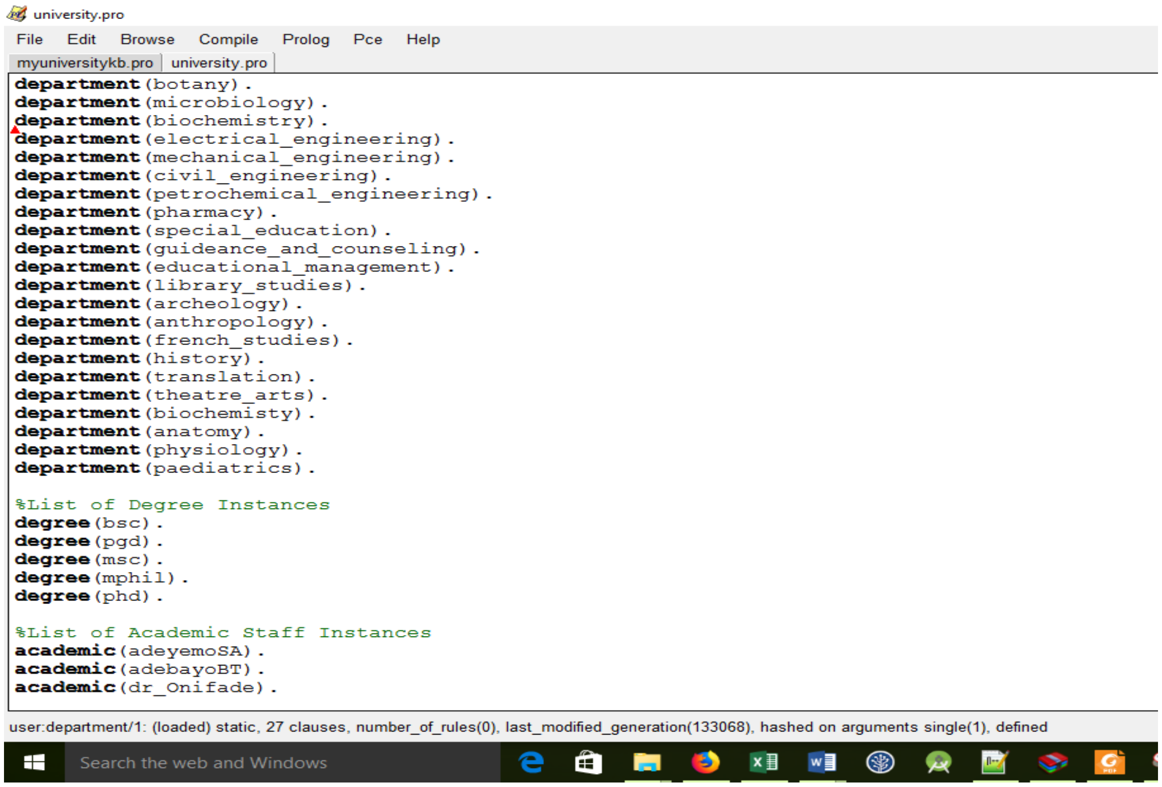
## VI. IMPLEMENTATION AND RESULTS

All the axioms represented above were implemented using Prolog Programming Language. SWI-Prolog IDE was used to write the codes, test, debug and consult the knowledge-base. The competency of the ontology was also tested by querying it with the competency questions. The prolog knowledge base, comprising of the facts and rules defined, and some of the results obtained are shown in Figure 13 – 24 below.


university.pro
File Edit Browse Compile Prolog Pce Help
myuniversitykb.pro university.pro
Search the web and Windows

```
%NAME:      ADEYEMI ADEDOYIN SIMEON
%MATRIC:    209188
%COURSE:    CSC 720 (KNOWLEDGE BASED-SYSTEMS)
%DEGREE:    MSC, COMPUTER SCIENCE, UNIVERSITY OF IBADAN, NIGERIA
%TOPIC:     AN ONTOLOGICAL-BASED APPROACH TO KNOWLEDGE REPRESENTATION AND
%           REASONING IN AI FOR UNIVERSITY DOMAIN
%DATE:      AUGUST 31, 2018

%-------------------------------------------
% LIST OF FACTS
%-----------------------------------------

%List of University Instances
university(ui).
university(unilorin).
university(lautech).
university(oau).

%List of Faculty Instances
faculty(science).
faculty(education).
faculty(technology).
faculty(pharmacy).
faculty(arts).
faculty(basic_medical_science).

%List of Department Instances
department(computer_science).
department(chemistry).
department(physics).
department(zoology).
department(geology).
department(botany).
department(microbiology).
```

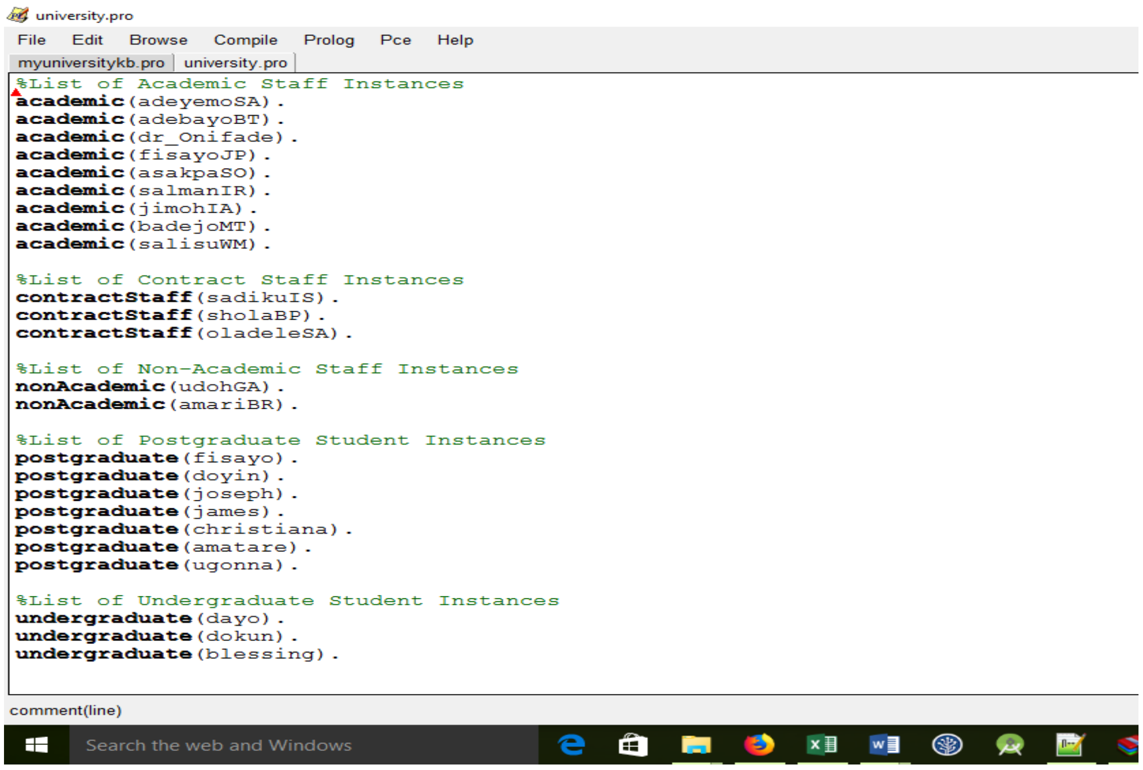
Figure 13. Screenshot of Prolog Facts (a)


university.pro
File Edit Browse Compile Prolog Pce Help
myuniversitykb.pro university.pro
Search the web and Windows

```
department(botany).
department(microbiology).
department(biochemistry).
department(electrical_engineering).
department(mechanical_engineering).
department(civil_engineering).
department(petrochemical_engineering).
department(pharmacy).
department(special_education).
department(guideance_and_counseling).
department(educational_management).
department(library_studies).
department(archeology).
department(anthropology).
department(french_studies).
department(history).
department(translation).
department(theatre_arts).
department(biochemisty).
department(anatomy).
department(physiology).
department(paediatrics).

%List of Degree Instances
degree(bsc).
degree(pgd).
degree(msc).
degree(mphil).
degree(phd).

%List of Academic Staff Instances
academic(adeyemoSA).
academic(adebayoBT).
academic(dr_Onifade).
```

```
user:department/1: (loaded) static, 27 clauses, number_of_rules(0), last_modified_generation(133068), hashed on arguments single(1), defined
```

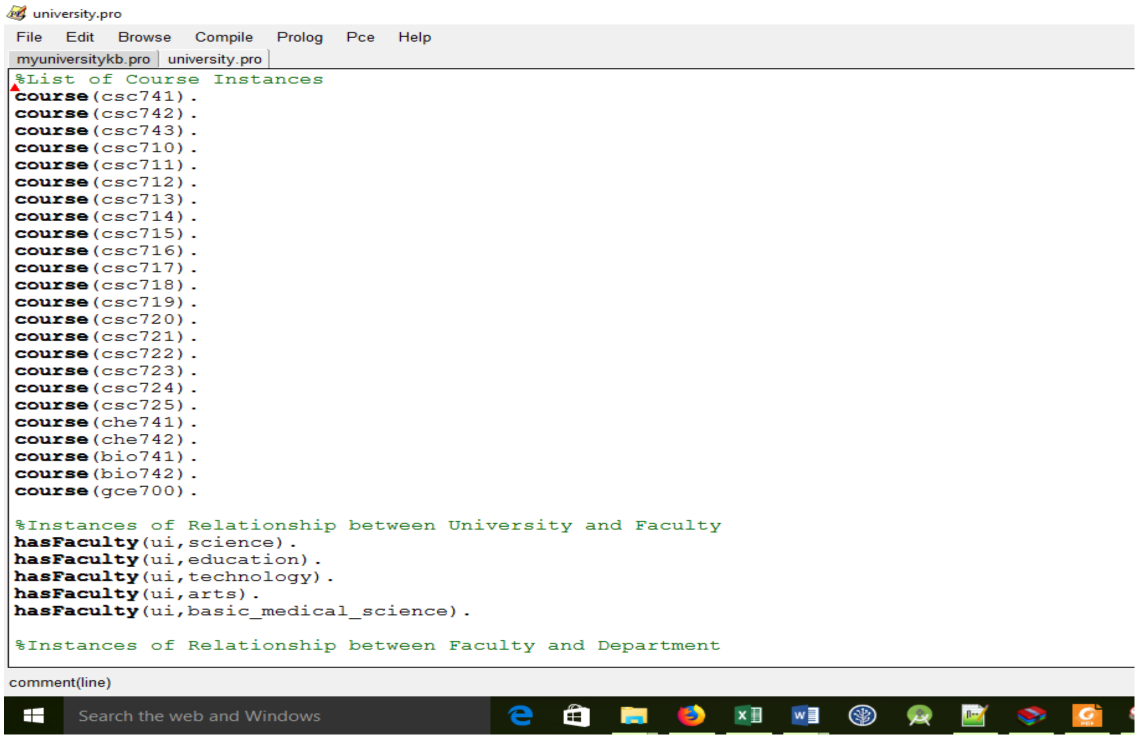
Figure 14. Screenshot of Prolog Facts (b)


university.pro
File Edit Browse Compile Prolog Pce Help
myuniversitykb.pro university.pro
Search the web and Windows

```
%List of Academic Staff Instances
academic(adeyemoSA).
academic(adebayoBT).
academic(dr_Onifade).
academic(fisayoJP).
academic(asakpaSO).
academic(salmanIR).
academic(jimohIA).
academic(badejoMT).
academic(salisuWM).

%List of Contract Staff Instances
contractStaff(sadikuIS).
contractStaff(sholaBP).
contractStaff(oladeleSA).

%List of Non-Academic Staff Instances
nonAcademic(udohGA).
nonAcademic(amariBR).

%List of Postgraduate Student Instances
postgraduate(fisayo).
postgraduate(doyin).
postgraduate(joseph).
postgraduate(james).
postgraduate(christiana).
postgraduate(amatare).
postgraduate(ugonna).

%List of Undergraduate Student Instances
undergraduate(dayo).
undergraduate(dokun).
undergraduate(blessing).
```

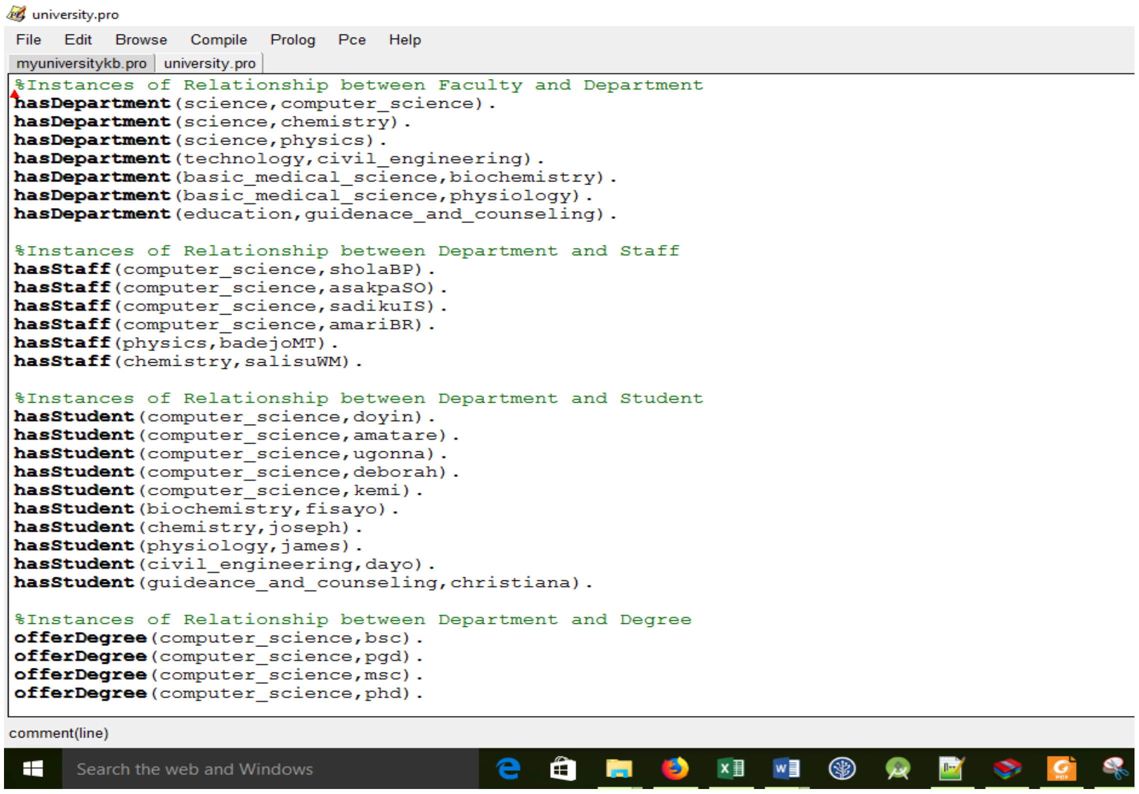
```
comment(line)
```

Figure 15. Screenshot of Prolog Facts (c)


university.pro
File Edit Browse Compile Prolog Pce Help
myuniversitykb.pro university.pro
comment(line)
Search the web and Windows

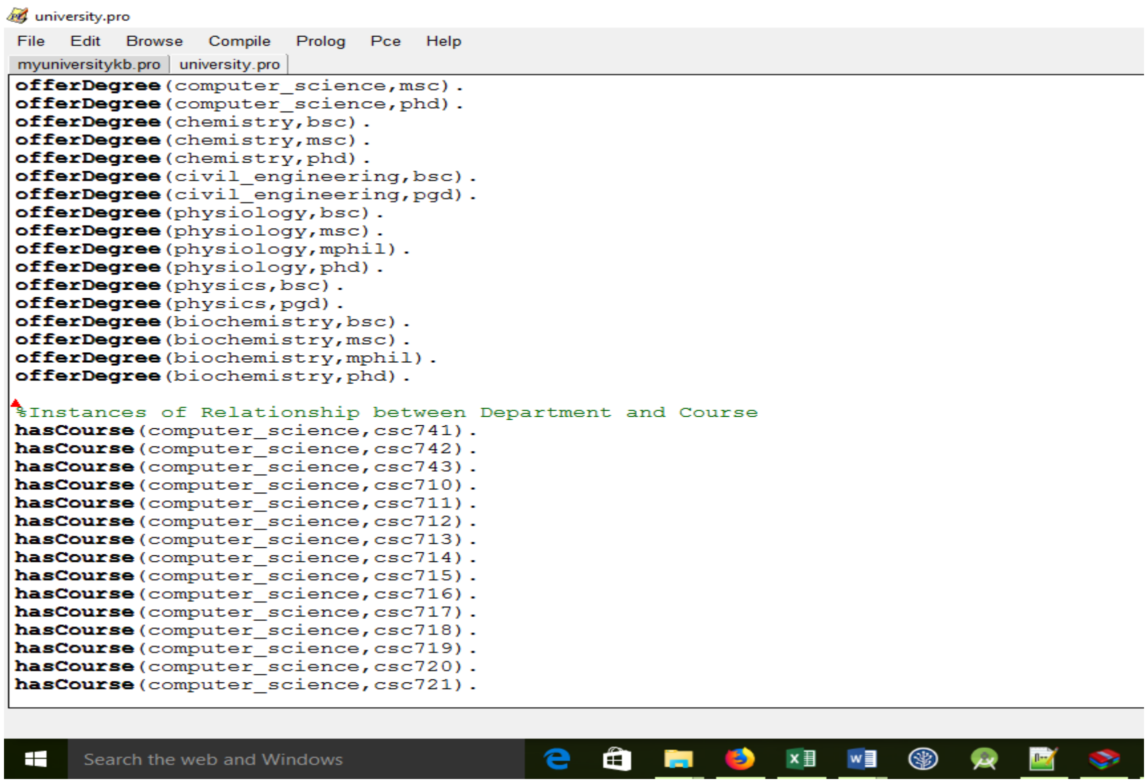
```
%List of Course Instances
course(csc741).
course(csc742).
course(csc743).
course(csc710).
course(csc711).
course(csc712).
course(csc713).
course(csc714).
course(csc715).
course(csc716).
course(csc717).
course(csc718).
course(csc719).
course(csc720).
course(csc721).
course(csc722).
course(csc723).
course(csc724).
course(csc725).
course(che741).
course(che742).
course(bio741).
course(bio742).
course(gce700).

%Instances of Relationship between University and Faculty
hasFaculty(ui,science).
hasFaculty(ui,education).
hasFaculty(ui,technology).
hasFaculty(ui,arts).
hasFaculty(ui,basic_medical_science).

%Instances of Relationship between Faculty and Department
```

Figure 16. Screenshot of Prolog Facts (d)


university.pro
File Edit Browse Compile Prolog Pce Help
myuniversitykb.pro university.pro
comment(line)
Search the web and Windows

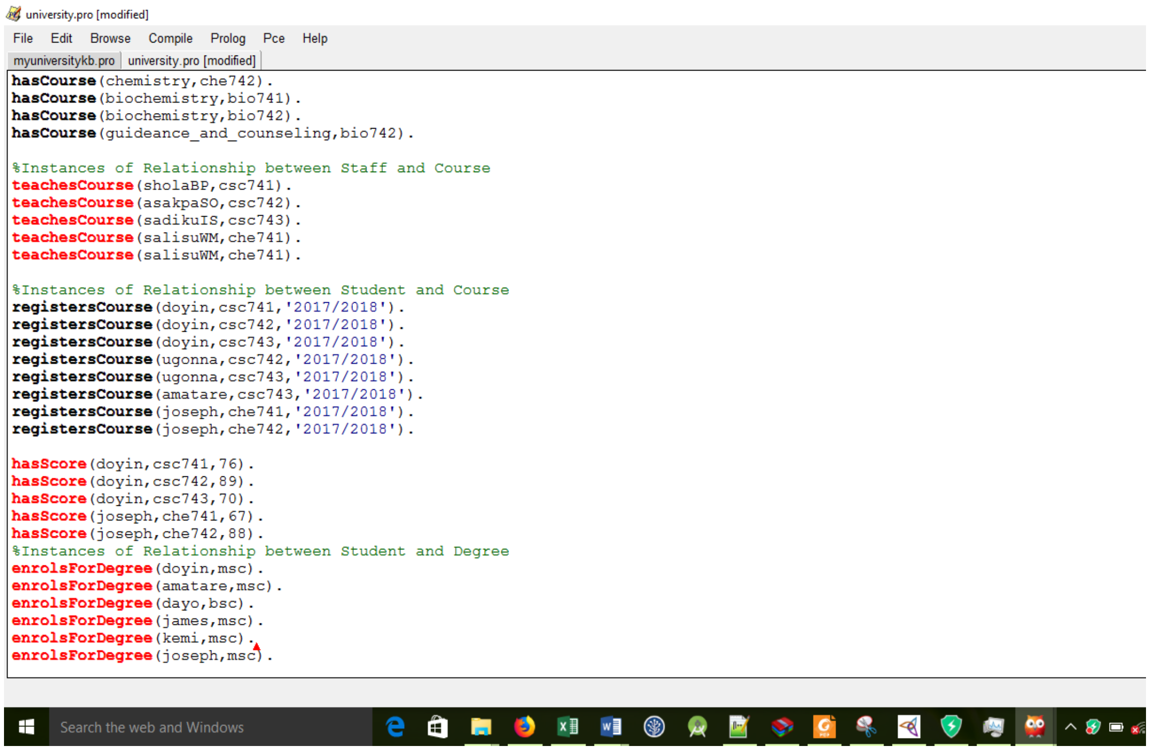
```
%Instances of Relationship between Faculty and Department
hasDepartment(science,computer_science).
hasDepartment(science,chemistry).
hasDepartment(science,physics).
hasDepartment(technology,civil_engineering).
hasDepartment(basic_medical_science,biochemistry).
hasDepartment(basic_medical_science,physiology).
hasDepartment(education,guidenace_and_counseling).

%Instances of Relationship between Department and Staff
hasStaff(computer_science,sholaBP).
hasStaff(computer_science,asakpaSO).
hasStaff(computer_science,sadikuIS).
hasStaff(computer_science,amariBR).
hasStaff(physics,badejoMT).
hasStaff(chemistry,salisuWM).

%Instances of Relationship between Department and Student
hasStudent(computer_science,doyin).
hasStudent(computer_science,amatare).
hasStudent(computer_science,ugonna).
hasStudent(computer_science,deborah).
hasStudent(computer_science,kemi).
hasStudent(biochemistry,fisayo).
hasStudent(chemistry,joseph).
hasStudent(physiology,james).
hasStudent(civil_engineering,dayo).
hasStudent(guideance_and_counseling,christiana).

%Instances of Relationship between Department and Degree
offerDegree(computer_science,bsc).
offerDegree(computer_science,pgd).
offerDegree(computer_science,msc).
offerDegree(computer_science,phd).
```

Figure 17. Screenshot of Prolog Facts (e)


university.pro
File Edit Browse Compile Prolog Pce Help
myuniversitykb.pro university.pro
Search the web and Windows

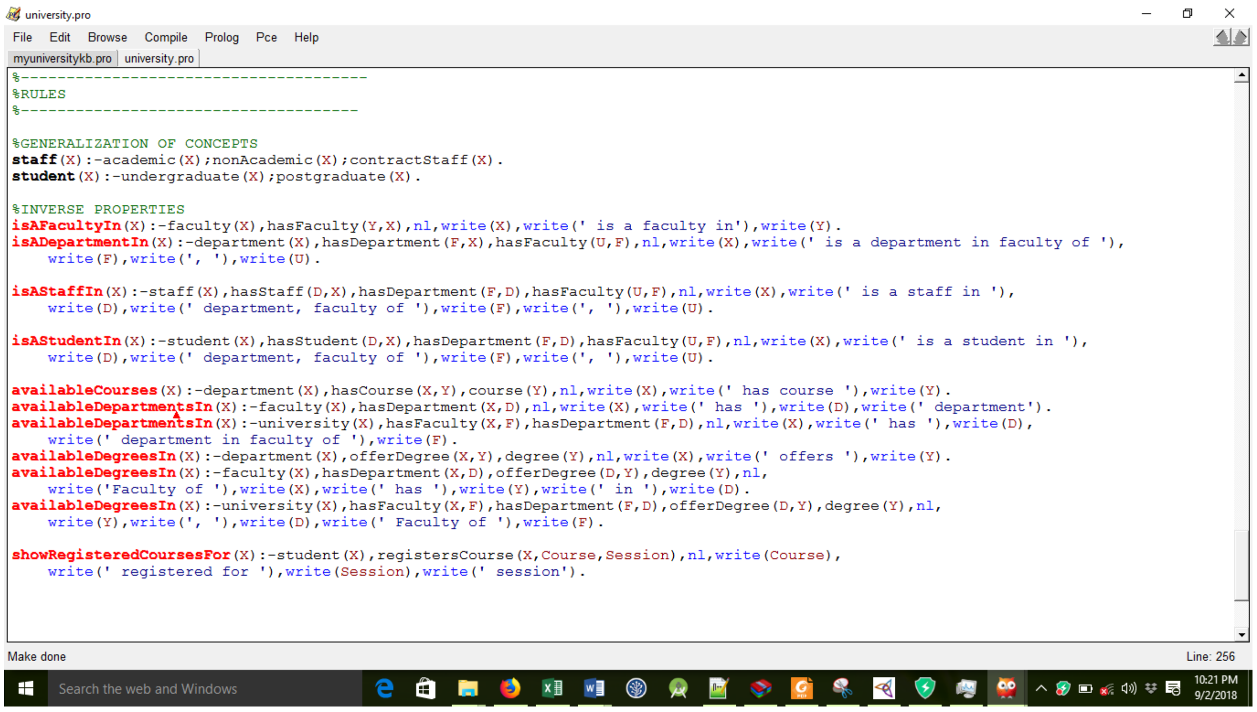
```
offerDegree(computer_science,msc).
offerDegree(computer_science,phd).
offerDegree(chemistry,bsc).
offerDegree(chemistry,msc).
offerDegree(chemistry,phd).
offerDegree(civil_engineering,bsc).
offerDegree(civil_engineering,pgd).
offerDegree(physiology,bsc).
offerDegree(physiology,msc).
offerDegree(physiology,mphil).
offerDegree(physiology,phd).
offerDegree(physics,bsc).
offerDegree(physics,pgd).
offerDegree(biochemistry,bsc).
offerDegree(biochemistry,msc).
offerDegree(biochemistry,mphil).
offerDegree(biochemistry,phd).

%Instances of Relationship between Department and Course
hasCourse(computer_science,csc741).
hasCourse(computer_science,csc742).
hasCourse(computer_science,csc743).
hasCourse(computer_science,csc710).
hasCourse(computer_science,csc711).
hasCourse(computer_science,csc712).
hasCourse(computer_science,csc713).
hasCourse(computer_science,csc714).
hasCourse(computer_science,csc715).
hasCourse(computer_science,csc716).
hasCourse(computer_science,csc717).
hasCourse(computer_science,csc718).
hasCourse(computer_science,csc719).
hasCourse(computer_science,csc720).
hasCourse(computer_science,csc721).
```

Figure 18. Screenshot of Prolog Facts (f)


university.pro [modified]
File Edit Browse Compile Prolog Pce Help
myuniversitykb.pro university.pro [modified]
Search the web and Windows

```
hasCourse(chemistry,che742).
hasCourse(biochemistry,bio741).
hasCourse(biochemistry,bio742).
hasCourse(guideance_and_counseling,bio742).

%Instances of Relationship between Staff and Course
teachesCourse(sholaBP,csc741).
teachesCourse(asakpaSO,csc742).
teachesCourse(sadikuIS,csc743).
teachesCourse(salisuWM,che741).
teachesCourse(salisuWM,che741).

%Instances of Relationship between Student and Course
registersCourse(doyin,csc741,'2017/2018').
registersCourse(doyin,csc742,'2017/2018').
registersCourse(doyin,csc743,'2017/2018').
registersCourse(ugonna,csc742,'2017/2018').
registersCourse(ugonna,csc743,'2017/2018').
registersCourse(amatare,csc743,'2017/2018').
registersCourse(joseph,che741,'2017/2018').
registersCourse(joseph,che742,'2017/2018').

hasScore(doyin,csc741,76).
hasScore(doyin,csc742,89).
hasScore(doyin,csc743,70).
hasScore(joseph,che741,67).
hasScore(joseph,che742,88).
%Instances of Relationship between Student and Degree
enrolsForDegree(doyin,msc).
enrolsForDegree(amatare,msc).
enrolsForDegree(dayo,bsc).
enrolsForDegree(james,msc).
enrolsForDegree(kemi,msc).
enrolsForDegree(joseph,msc).
```

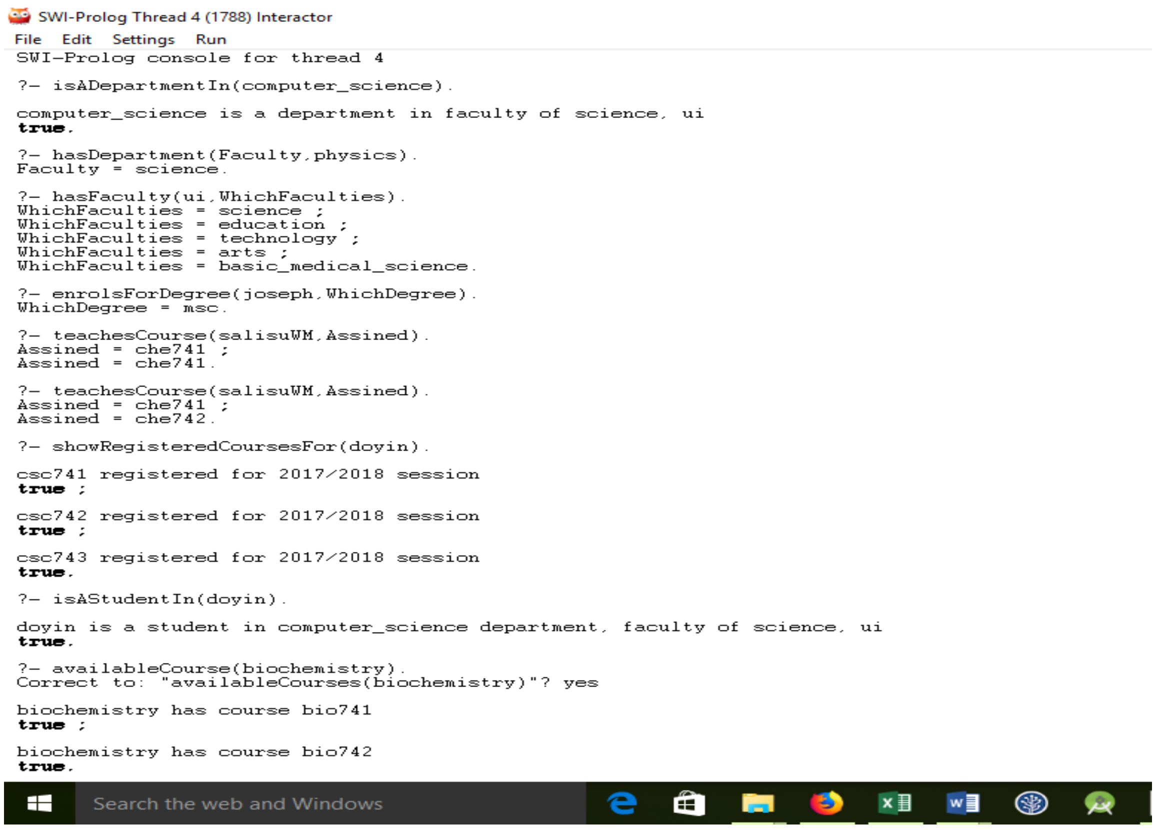
Figure 19. Screenshot of Prolog Facts (g)



```
%---------------------------------------
%RULES
%-------------------------------------

%GENERALIZATION OF CONCEPTS
staff(X):-academic(X);nonAcademic(X);contractStaff(X).
student(X):-undergraduate(X);postgraduate(X).

%INVERSE PROPERTIES
isAFacultyIn(X):-faculty(X),hasFaculty(Y,X),nl,write(X),write(' is a faculty in'),write(Y).
isADepartmentIn(X):-department(X),hasDepartment(F,X),hasFaculty(U,F),nl,write(X),write(' is a department in faculty of '),
    write(F),write(', '),write(U).

isAStaffIn(X):-staff(X),hasStaff(D,X),hasDepartment(F,D),hasFaculty(U,F),nl,write(X),write(' is a staff in '),
    write(D),write(' department, faculty of '),write(F),write(', '),write(U).

isAStudentIn(X):-student(X),hasStudent(D,X),hasDepartment(F,D),hasFaculty(U,F),nl,write(X),write(' is a student in '),
    write(D),write(' department, faculty of '),write(F),write(', '),write(U).

availableCourses(X):-department(X),hasCourse(X,Y),course(Y),nl,write(X),write(' has course '),write(Y).
availableDepartmentsIn(X):-faculty(X),hasDepartment(X,D),nl,write(X),write(' has '),write(D),write(' department').
availableDepartmentsIn(X):-university(X),hasFaculty(X,F),hasDepartment(F,D),nl,write(X),write(' has '),write(D),
    write(' department in faculty of '),write(F).
availableDegreesIn(X):-department(X),offerDegree(X,Y),degree(Y),nl,write(X),write(' offers '),write(Y).
availableDegreesIn(X):-faculty(X),hasDepartment(X,D),offerDegree(D,Y),degree(Y),nl,
    write('Faculty of '),write(X),write(' has '),write(Y),write(' in '),write(D).
availableDegreesIn(X):-university(X),hasFaculty(X,F),hasDepartment(F,D),offerDegree(D,Y),degree(Y),nl,
    write(Y),write(', '),write(D),write(' Faculty of '),write(F).

showRegisteredCoursesFor(X):-student(X),registersCourse(X,Course,Session),nl,write(Course),
    write(' registered for '),write(Session),write(' session').
```

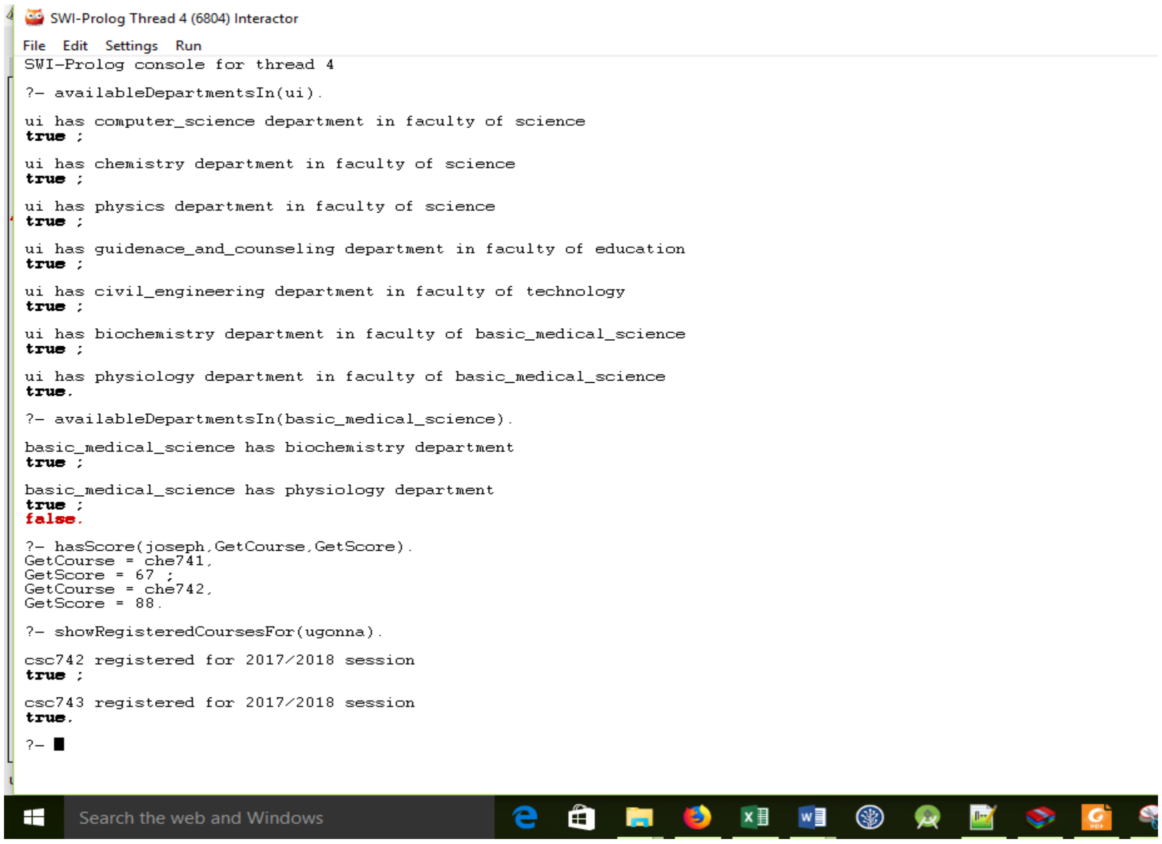
Make done

Figure 20. Screenshot of Prolog Facts (h)



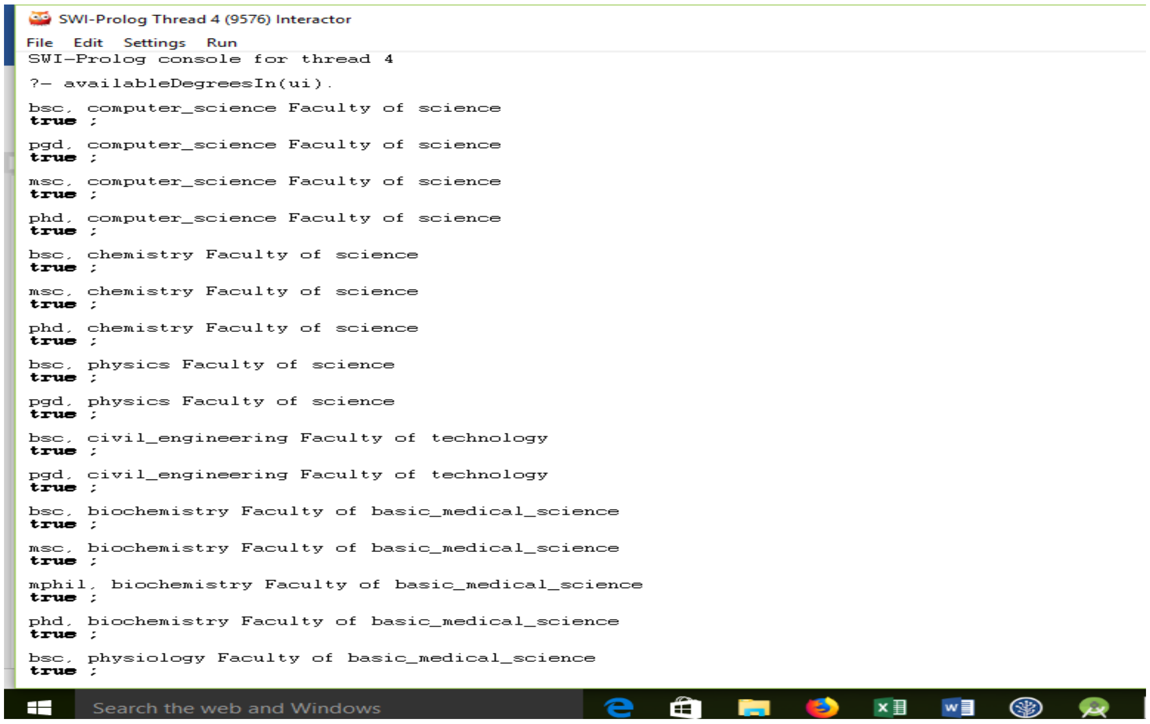
```
SWI-Prolog console for thread 4

?- isADepartmentIn(computer_science).

computer_science is a department in faculty of science, ui
true.

?- hasDepartment(Faculty,physics).
Faculty = science.

?- hasFaculty(ui,WhichFaculties).
WhichFaculties = science ;
WhichFaculties = education ;
WhichFaculties = technology ;
WhichFaculties = arts ;
WhichFaculties = basic_medical_science.

?- enrolsForDegree(joseph,WhichDegree).
WhichDegree = msc.

?- teachesCourse(salisuWM,Assined).
Assined = che741 ;
Assined = che741.

?- teachesCourse(salisuWM,Assined).
Assined = che741 ;
Assined = che742.

?- showRegisteredCoursesFor(doyin).

csc741 registered for 2017/2018 session
true ;

csc742 registered for 2017/2018 session
true ;

csc743 registered for 2017/2018 session
true.

?- isAStudentIn(doyin).

doyin is a student in computer_science department, faculty of science, ui
true.

?- availableCourse(biochemistry).
Correct to: "availableCourses(biochemistry)"? yes

biochemistry has course bio741
true ;

biochemistry has course bio742
true.
```

Figure 21. Result of Querying the Knowledge Base with Competency Questions (a).


SWI-Prolog Thread 4 (6804) Interactor
File Edit Settings Run
Search the web and Windows

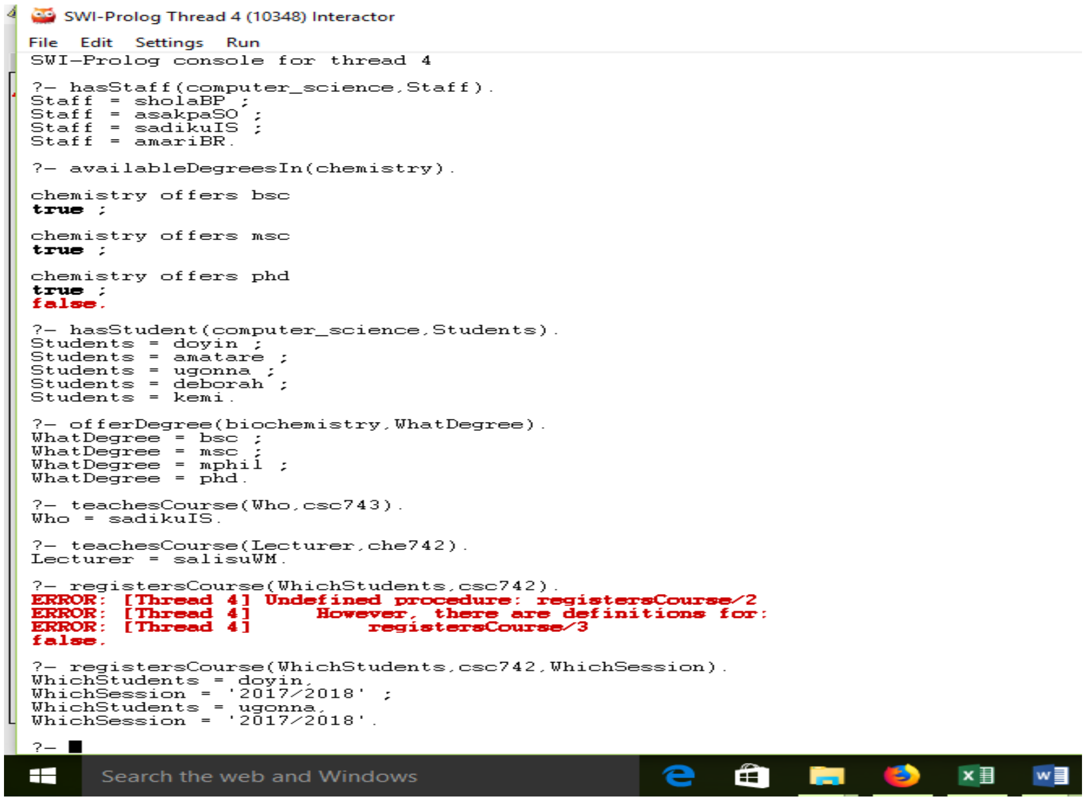
```
SWI-Prolog console for thread 4

?- availableDepartmentsIn(ui).

ui has computer_science department in faculty of science
true ;

ui has chemistry department in faculty of science
true ;

ui has physics department in faculty of science
true ;

ui has guidenace_and_counseling department in faculty of education
true ;

ui has civil_engineering department in faculty of technology
true ;

ui has biochemistry department in faculty of basic_medical_science
true ;

ui has physiology department in faculty of basic_medical_science
true.

?- availableDepartmentsIn(basic_medical_science).

basic_medical_science has biochemistry department
true ;

basic_medical_science has physiology department
true ;
false.

?- hasScore(joseph,GetCourse,GetScore).
GetCourse = che741,
GetScore = 67 ;
GetCourse = che742,
GetScore = 88.

?- showRegisteredCoursesFor(ugonna).

csc742 registered for 2017/2018 session
true ;

csc743 registered for 2017/2018 session
true.

?- 
```

Figure 22. Result of Querying the Knowledge Base with Competency Questions (b)


SWI-Prolog Thread 4 (9576) Interactor
File Edit Settings Run
Search the web and Windows

```
SWI-Prolog console for thread 4

?- availableDegreesIn(ui).

bsc, computer_science Faculty of science
true ;

pgd, computer_science Faculty of science
true ;

msc, computer_science Faculty of science
true ;

phd, computer_science Faculty of science
true ;

bsc, chemistry Faculty of science
true ;

msc, chemistry Faculty of science
true ;

phd, chemistry Faculty of science
true ;

bsc, physics Faculty of science
true ;

pgd, physics Faculty of science
true ;

bsc, civil_engineering Faculty of technology
true ;

pgd, civil_engineering Faculty of technology
true ;

bsc, biochemistry Faculty of basic_medical_science
true ;

msc, biochemistry Faculty of basic_medical_science
true ;

mphil, biochemistry Faculty of basic_medical_science
true ;

phd, biochemistry Faculty of basic_medical_science
true ;

bsc, physiology Faculty of basic_medical_science
true ;
```

Figure 23. Result of Querying the Knowledge Base with Competency Questions (c)


SWI-Prolog Thread 4 (10348) Interactor
File Edit Settings Run
Search the web and Windows

```
SWI-Prolog console for thread 4

?- hasStaff(computer_science,Staff).
Staff = sholaBP ;
Staff = asakpaSO ;
Staff = sadikuIS ;
Staff = amariBR.

?- availableDegreesIn(chemistry).

chemistry offers bsc
true ;

chemistry offers msc
true ;

chemistry offers phd
true ;
false.

?- hasStudent(computer_science,Students).
Students = doyin ;
Students = amatare ;
Students = ugonna ;
Students = deborah ;
Students = kemi.

?- offerDegree(biochemistry,WhatDegree).
WhatDegree = bsc ;
WhatDegree = msc ;
WhatDegree = mphil ;
WhatDegree = phd.

?- teachesCourse(Who,csc743).
Who = sadikuIS.

?- teachesCourse(Lecturer,che742).
Lecturer = salisuWM.

?- registersCourse(WhichStudents,csc742).
ERROR: [Thread 4] Undefined procedure: registersCourse/2
ERROR: [Thread 4]     However, there are definitions for:
ERROR: [Thread 4]         registersCourse/3
false.

?- registersCourse(WhichStudents,csc742,WhichSession).
WhichStudents = doyin,
WhichSession = '2017/2018' ;
WhichStudents = ugonna,
WhichSession = '2017/2018'.

?- 
```

Figure 24. Result of Querying the Knowledge Base with Competency Questions (d)

## VII. CONCLUSION AND FUTURE WORK

From the results obtained, it has been shown that the ontological-modeled knowledge base is highly competent in providing correct answers to the competency questions. The versatility of Prolog in ontological development, simplicity of expression in a declarative programming language like Prolog and the knowledge representational power of First Order Predicate Logic made this feasible. The university ontology has been able to capture relevant academic-oriented knowledge base for a university to preserve knowledge and conformity in knowledge expression and usage among students, staff and software agents and other relevant agents that require such a knowledge for decision making or reasoning. Future work can still be carried out to capture a more robust aspect of a complex university domain and improve upon the scope of this work.

## VIII. REFERENCES

Akande, O., Akinkunmi, B. O. & Ayorinde, I. T. (2015). An ontological informatics on development of tablet drug in pharmaceutical industry. *International Journal of Advanced research in Computer Science*, *6*(8).

Ayesha, A., Khaleel, U.R.K. & Padmaja, R. (2012). *Creation of ontology in education.* IEEE Fourth International Conference on Technology for Education.

Chimaera (2000). Chimaera Ontology Environment. Retrieved on August 29, 2018 from http://www.ksl.stanford.edu/software/chimaera.

Genesereth, M.R. & Nilsson, N.J. (1987). Logical foundations of artificial intelligence. San Mateo, CA: Morgan Kaufmann.

Ifnord (2018). Ontology (Information Science). Retrieved August 31, 2018 from https://en.m.wikipedia.org/wiki/Ontology_(information_science).

Maria, G., Akrivi, K., Costas, V., George, L. & Constantin, H. (2007). Creating an ontology for user profile: Method and applications.

Mathew, H., Holger, K., Alan, R., Robert, S., Chris, W., Simon, J., … Sebastian, B. (2011). A practical guide to building OWL ontologies using Protégé 4 and CO-ODE tools. Ed. 1.3, University of Manchester.

McGuinness, D.L., Fikes, R., Rice, J. & Wilder, S. (2000). *An environment for merging and testing large ontologies: Principles of knowledge representation and reasoning.* Proceedings of the Seventh International Conference (KR2000). A. G. Cohn, F. Giunchiglia and B. Selman(Eds.), San Francisco, CA, Morgan Kaufmann.

Natalya, F.N. & Deborah, L.M. (2001). Ontology development 101: A guide to creating your first ontology.

Thomas, R.G. (1995). Towards principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies, 43*(1), 907-028.