

cps721: Assignment 4 (100 points).

Due date: Electronic file - Monday, November 12, 2018, 9:00pm (sharp).

You must work in groups of TWO, or THREE, you cannot work alone.

YOU SHOULD NOT USE ";", "!", AND "->" IN YOUR PROLOG RULES.

You can discuss this assignment only with your CPS721 group partners or with the CPS721 instructor.

By submitting this assignment you acknowledge that you read and understood the course Policy on Collaboration in homework assignments stated in the CPS721 course management form.

This assignment asks you to build a simple natural language system for answering "who" and "what" questions about noun phrases, much like the example in class, but instead of using the database about parks, hats, people, etc, you have to expand a database about bookstores and shipping that you have created in the first assignment.

1. Before we start using any English words, reuse (or expand) your simple database of facts (e.g., at least 10 facts of each type) with the following predicates:

- *hasBook(Bookstore,Author,Title,ListPrice)* – the *Bookstore* sells a book with the *Title* written by the *Author* for the *ListPrice*,
- *lives(Person,City)* – the *Person* lives in the *City*,
- *shipping(Bookstore,City,Cost)* – the shipping cost from the *Bookstore* to the *City* is the *Cost*.

Keep your database in the file **nlul.pl**. The facts included in your database can be imaginary; make it up so that most queries will get positive answers and will retrieve some data from your data base. However, note that some of the examples below use names of the authors who wrote AI textbooks (mentioned in the course management form).

Show that your database works properly by formulating the testing queries in Prolog (similar to what you did in the first assignment), and obtaining suitable answers. (These queries should *not* use English noun phrases!) It is up to you to formulate a range of queries to convince yourself and the TA. However, it is strongly recommended to use as queries Prolog versions of the noun phrases, so that you will show to the TA what answers should be expected from your database when it will be subsequently queried using English noun phrases. Formulate at least 10 Prolog queries to your database. Keep your queries and answers computed by Prolog in the file **nlul.txt**

Once you have this working, you are ready to consider English noun phrases and the book titles, authors and shipping costs they refer to. Here are some examples of the kinds of queries your system should be able to handle:

1. *who([an,author,from,toronto], A).*
2. *who([an,author,of,an,expensive,book,at,amazon], A).*
3. *who([a,person, from, newYork], P).*
4. *what([a,bookstore, with,a,high,shipping,to,toronto], B).*
5. *what([a,shipping,from,indigo,to,toronto], C).*
6. *what([a,book,from,amazon], B).*
7. *what([an,expensive,book,by,russell], B).*
8. *what([a,bookstore,with,an,expensive,harry_potter], B).*
9. *what([a,bookstore,with,a,harry_potter,for,110], B).*

10. `what([a,book,by,russell,at,amazon,with, a, high,price], B).`

11. `what([a,low,price,of,a,book,by,poole], P).`

12. `what([a,price,of,computational_intelligence], P).`

2. Build a lexicon, as we did in class, of articles, adjectives, proper nouns, common nouns, and prepositions. To cover the above examples, you will need at least these words :

articles: `a, an`

common nouns: `city, person, author, bookstore, book, title, shipping, price, ...`

prepositions: `with, of, for, from, to, by, at, ...`

proper nouns: `toronto, montreal, poole, levesque, computational_intelligence, ...`

adjectives: `cheap, expensive, low, high, moderate, ...`

For the purposes of this assignment, *shipping* is considered as a noun representing the shipping cost. For simplicity, assume that books with list prices less than or equal to 50 are considered cheap, or having low listed price, but the books with prices higher than (or equal) 100 are expensive, or, in other words, have a high price. In addition, all shipping costs below 2 are defined to be low, all shipping costs above 6 are considered to be high, and all other shipping costs are moderate.

You may actually need more words than those mentioned explicitly above. At this stage, you might wish to elaborate your database by adding several new facts with other predicates: these predicates will be new concepts linked to English words described in the lexicon. Remember that it is easy to defeat a language understanding program by using a word that it does not know about. Vocabulary is important in these systems. Make yours as smart as you can.¹

3. Copy the noun phrase parser/interpreter given in class (or write your own), and define the `what` predicate used above. The parser must be also in the same file **`nlu.pl`**

4. Test the `what` and `who` predicates on a variety of noun phrases, like those above, showing that it is capable of identifying the entities being referred to by your noun phrases. It is up to you to choose noun phrases for testing, but you must convincingly demonstrate that your program works properly (try at least **10 new** different noun phrases in addition to the phrases given to you). Remember that testing your program is very important part of the software development cycle. You lose marks if you do not test your program as required. Copy all results of your tests into **`nlu.txt`**

5. Do not attempt this work until the previous parts of your assignment are complete. The English noun phrases described above are quite limited. Handle the article “the” properly. The trick here is that a noun phrase like “the cheap book by levesque from amazon” should succeed in naming a book only if there is a *unique* book of the appropriate kind. But if there are several different books characterized by this phrase, then the query should fail. Similarly, “the shipping from indigo to toronto” should retrieve the unique number representing the shipping cost from indigo to toronto.

Handing in solutions. An electronic copy of: (a) your database, lexicon and parser in a single file (the name of the file must be **`nlu.pl`**), (b) your session with Prolog, showing the queries you submitted and the answers returned (the name of the file must be **`nlu.txt`**). It is up to you to formulate a range of queries that demonstrates that your program is working properly.

¹Three (3) best Prolog programs that will do more than specified explicitly in this assignment will get bonus marks (as announced in class). Bonus marks will be given at the discretion of the TA: minor variations will not be awarded any extra marks. Demonstrate your creativity to get bonus marks!

How to submit this assignment. Read regularly *Frequently Answered Questions* and replies to them that are linked from the Assignments Web page at

<http://www.scs.ryerson.ca/~mes/courses/cps721/assignments.html>

If you write your code on a Windows machine, make sure you save your files as plain text that one can easily read on Linux machines. Before you submit your Prolog code electronically make sure that your files do not contain any extra binary symbols: it should be possible to load `nlu.pl` into a recent release 6 of ECLiPSe Prolog, compile your program and ask testing queries. TA will mark your assignment using ECLiPSe Prolog. If you run any other version of Prolog on your home computer, it is your responsibility to make sure that your program will run on ECLiPSe Prolog (release 6 or any more recent release), as required. For example, you can run a command-line version of *eclipse* on moon remotely from your home computer to test your program (read handout about running *ECLiPSe Prolog*). To submit files electronically do the following. First, create a **zip** archive on moon:

```
zip yourLoginName.zip nlu.pl nlu.txt
```

where `yourLoginName` is the login name of the person who submits this assignment from a group. Remember to mention at the beginning of each file *student*, *section numbers* and *names* of all people who participated in discussions (see the course management form). You may be penalized for not doing so. Second, upload **your ZIP** file only (**No individual files!**) **yourLoginName.zip** into the “Assignment 4” folder on D2L.

Revisions: If you would like to submit a revised copy of your assignment, then run simply the submit command again. (The same person must run the submit command.) A new copy of your assignment will override the old copy. You can submit new versions as many times as you like and you do not need to inform anyone about this. Don’t ask your team members to submit your assignment, because TA will be confused which version to mark: only one person from a group should submit different revisions of the assignment. The time stamp of the last file you submit will determine whether you have submitted your assignment on time.