CPS721: Assignment 4

Due: November 14, 2022, 9pm
Total Marks: 100 (worth 4% of course mark)
You MUST work in groups of 2 or 3

Late Policy: The penalty for submitting even one minute late is 10%. Assignments are not accepted more than 24 hours late.

Clarifications and Questions: Please use the discussion forum on the D2L site to ask questions as they come up. These will be monitored regularly. Clarifications will be made there as needed. A Frequently Asked Questions Page will also be created. You may also email your questions to your instructor, but please check the D2L forum and frequently asked questions first.

Collaboration Policy: You can only discuss this assignment with your group partners or with your CPS721 instructor. By submitting this assignment, you acknowledge that you have read and understood the course policy on collaboration as stated in the CPS721 course management form.

PROLOG Instructions: When you write your rules in PROLOG, you are not allowed to use ";" (disjunction), "!" (cut), and "->" (if-then). You are only allowed to use ";" to get additional responses when interacting with PROLOG from the command line. Note that this is equivalent to using the "More" button in the ECLiPSe GUI.

We will be using ECLiPSE Prolog release 6 to mark the assignments. If you run any other version of PROLOG, it is your responsibility to check that it also runs in ECLiPSE Prolog release 6.
Submission Instructions: You should submit ONE zip file containing 9 files:

```
computer_shack.pl
question1_queries.txt, question1_interaction.txt
question3_queries.txt, question3_interaction.txt
question4_queries.txt, question4_interaction.txt
```

Your submission should not include any other files. If you submit a `.rar`, `.tar`, `.7zip`, or other compression format aside from `.zip`, you will lose marks. The name of the zip file should be `yourLoginName.zip` where `yourLoginName` is only the group member who made the submission. Note that only one student in the group should make submissions, as D2L can present confusing information if more than one student in the group submits.

All submissions should be made on D2L. Submissions by email will not be accepted.

You are allowed to make as many submissions as you want, and you do not have to inform anyone about this. The new copy will override the old one. The time stamp of the last submission will be used to determine the submission time. However, the same group member should make all the submissions for the reasons explained above.

If you write your code on a Windows machine, make sure the files are saved on plain text and are readable on Linux machines. Ensure your PROLOG code does not contain any extra binary symbols and that they can be compiled by ECLiPSE Prolog release 6.

1 Building a Database [20 marks]

This assignment will exercise what you have learned about natural language understanding and processing. To do so, you will be developing a "smart" website for a chain of electronics stores called Computer Shack that will allow a user to make natural language queries about available products. Ultimately, we would like to be able to interact with the system with statements like "Is there a Acer laptop at a store in Toronto that costs less that $500." For the sake of simplicity, we will restrict the system to only handle noun phrases, just as we did in class.

To build such a system, you will need a database of facts, a lexicon, and a parser. In this question, you will construct the database.

a. [**10 marks**] Create a database by adding atomic propositions to the file `computer_shack.pl`. The atomic propositions should ONLY use the following predicates:

- `product(ProductName, Manufacturer, Type, Price, Rating)` - defines a product with given `ProductName` (*ie.* Macbook Pro), `Manufacturer` (*ie.* Apple), `Type` (*ie.* laptop), `Price` (*ie.* $2000), and average customer `Rating` out of 5 (*ie.* 4.2).
- `inStock(ProductName, StoreName, Count)` - indicates that the store with name `StoreName` (*ie.* Eaton Centre Computer Shack) has the given `Count` (*ie.* 10) of `ProductName` (*ie.* Macbook Pro) in stock.
- `location(StoreName, City)` - indicates that the store with `StoreName` (*ie.* Eaton Centre Computer Shack) is in the given `City` (*ie.* Toronto).
- `canShip(ProductName, City)` - whether the given `ProductName` (*ie.* Macbook Pro) can be purchased online and shipped to a person in the given `City` (*ie.* Toronto).

You should add 10-15 atomic sentences per predicate to your database. The sentences should not use strings (*ie. 'Macbook Pro'*) and should instead use underscores for spaces (*ie.* `macbook_pro`). Your sentences should be enough to make ALL of the queries in part b of this question return true for at least one value. We also suggest you read question 2 to get an idea of additional statements that may be useful for testing later.

b. [**10 marks**] Create queries for each of the 10 statements below and add them to the file `question1_queries.txt`. Ensure that you put each query under the correct comment as otherwise you may lose marks. You can only use the predicates listed above with variables or constants as arguments, conjunction, and "not" (*ie.* negation) in your queries. You may also use $<, >, <=$, or $>=$, which PROLOG uses for less than, more than, etc. Keep in mind that when using a predicate $X < Y$, both X and Y should be instantiated before the comparison.

1. Is there a store that has laptops in stock? If yes, retrieve them one-by-one.
2. Is there a store in Montreal that has a 3ft HDMI cable in stock? (Note that 3ft HDMI cable is the name of the product, not the type).
3. What is the stock of the tablet called Galaxy Tab 10 in the Square One Computer Shack?
4. Does any store in Missisauga have a laptop in stock that cannot be shipped to Toronto?
5. What is the cheapest monitor that can ship to Toronto? Note that a product is said to be the *cheapest*, if it has the minimal cost of all products of the same type. A product is said to be *expensive* if it costs at least the 2 times the price of the cheapest product in the same category
6. What are the highly rated tablets that the Eaton Centre Computer Shack has in stock? An item is said to be "highly rated" if the average rating is at least 4. Similarly, we define "lowly rated" as having a rating below 2.5 and "medium rated" it if is at least 2.5 and less than 4.

7. Is there a laptop that can be bought at at least two different stores in Toronto? If there is, what is its price?

8. Is there a highly rated tablet that costs less than $600?

9. Is there laptop that can both be shipped to Mississauga and is at the Square One Computer Shack?

10. What is the cheapest highly rated tablet that I can get shipped?

You should also submit your interaction with this database in a file called `question1_interaction.txt`.

2 Building a Lexicon [45 marks]

You will now build a lexicon, as we did in class, of articles, adjectives, proper nouns, common nouns, and prepositions. Your lexicon should at least allow for the following queries:

1. `what([a,laptop,in,stock,at,square_one_computer_shack], P).`
2. `what([any,highly_rated,apple,laptop], P).`
3. `what([an,expensive,tablet,in,stock,at,eaton_centre_computer_shack], P).`
4. `what([the,rating,of,a,rocketfish,hdmi_cable], R).`
5. `what([the,stock,of,galaxy_tab_10,at,montreal_computer_shack], C).`
6. `what([a,highly_rated,laptop,in,stock,at,a,store,in,montreal], P).`
7. `what([the,cheapest,dell,monitor], P).`
8. `what([a,laptop,that_can_ship_to,mississauga], P).`
9. `what([any,store,in,toronto,with,an,apple,laptop,in,stock], S).`
10. `what([the,cheapest,hdmi_cord,at,a,store,in,toronto], P).`

As such, your lexicon will require at least the following words:

- articles - `a, an, any, the,` ...
- common nouns - `laptop, stock, rating, tablet, hdmi_cable, city,monitor`..
- preopositions - `in, at, that_can_ship_to, with`, ...
- proper nouns - `toronto, missisauga, montreal, square_one_computer_shack, eaton_centre_computer_shack`, ...
- adjectives - `apple, rocketfish, rated, highly_rated, medium_rated, lowly_rated, expensive, cheapest,` ...

Notice that while Apple, Rocketfish, etc. are brands, they can be treated as adjectives for our purposes. For simplicity, we are also treating `that_can_ship_to` as a single preposition. The word "any" should also be treated as an article.

You should also handle proper nouns different that discussed in class. Proper nouns are a referent to themselves and thus, you should handle them with rules of the following form:

```
proper_noun(X) :- person(X).    % X is the name of the person in this context
```

Note that product names, manufacturer names, store names, and cities are are proper nouns. All numerals like price and ratings are also proper nouns.

You should **not** introduce any new atomic statements, as the database should only include statements based on the 4 predicates defined in question 1. Your lexicon should include all the words mentioned above and possibly more. There should be at least 20 words in addition to articles and proper nouns. Notice that some words are ambiguous, and for this reason, you may need several rules for them in their lexicon (*ie.* one rule per meaning). As an example, recall the 4 rules for the preposition "with" that we discussed in class.

Remember that it is easy to "defeat" a language understanding program built this way by using a word it does not include. Ensure your vocabulary covers a variety of cases for this reason.

Include your lexicon in the relevant section of `computer_shack.pl`.

3 Building and Testing a Parser [20 marks]

For this question you will test your lexicon by trying queries using the `what` predicate and the simple noun phrase parser given in class. This parser has already been added to `computer_shack.pl`.

Test the `what` predicate using queries like those given above, showing that your system is capable of identifying the entities being referred to by your noun phrases. You should test **at least 10 new phrases** in addition to those given above. They should be selected to help convince the TA that they are correct. Put these phrases in the file `question3_queries.txt` (only your new queries are needed) and put the interaction into `question3_interaction.txt`. The interaction should include both the 10 queries above and your 10 (or more) new queries. Make sure that you include comments in the queries file describing each new query in plain English. You will lose marks if you do not test your program as required.

4 Additonal English Features [15 marks]

DO NOT ATTEMPT THIS PART UNTIL ALL REMAINING PARTS ARE COMPLETE.

In this part of the assignment, you will be adding additional English features to your system. In particular, you should allow for the following:

a. Add to your system the ability to handle the article “the” correctly. The key is that a noun phrase like “the rating of an Apple Macbook Pro” should succeed in naming the unique rating for the Macbook Pro, and it should fail if there are more than one option.

b. Add to your system the ability to handle prepositional phrases of the form “between X and Y” as in “a laptop with cost between $500 and $1000” or “a tablet rated between 3.5 and 4.5.” You may use the library predicate `number(X)` to test if the given value is a number. For simplicity, we will interpret “X between Y” as including X and Y in the range.

Put both of these features at the bottom of the lexicon section of `computer_shack.pl`. You should also provide sufficient tests for these features in a file called `question4_queries.txt` and put the interaction into `question3_interaction.txt`. Make sure that you include comments in the queries file describing each new query in plain English.