HARRISBURG UNIVERSITY OF SCIENCE AND TECHNOLOGY

# CISC 340 – Lab 1

# Logic Programming

Instructors:

Brian Grey M.S.

~~Chad Van Chu M.S.~~

**Introduction:**

In today's lab, you will apply what we learned in class to a real-world problem. In any school, there are courses you must take. Specifically, in the computer science department, you must take prerequisite courses before being able to enroll in a specific course. We will construct a logic program using SWI-Prolog which is able to query which classes are required to be taken in order to take a certain given class.

**Resources:**

http://www.swi-prolog.org/

http://www.swi-prolog.org/pldoc/doc_for?object=manual

http://lpn.swi-prolog.org/lpnpage.php?pageid=online

**Background Information:**

In class, we explored the logic programming paradigm. After realizing this new paradigm of programming, we put it to use by implementing some small knowledge bases through the logic programming language named Prolog. We talked about traditional artificial intelligence where we rediscovered the purpose of propositional logic, first order logic, rules of inference, propositional functions and quantifiers. We now know the importance of knowledge representation and reasoning and both of their roles in artificial intelligence.

**Getting Started:**

In order to get started, we must become familiar with the problem at hand. Please take this moment to download the current school catalog from https://myhu.harrisburgu.edu/ics.

From the Home tab of the MyHu webpage, there will be a "Quick Links" navigation bar on the left of the page. You can click the link labeled Archived Catalogs there (or here). From this page, you can download a PDF of the **Undergraduate Catalog**.

Once you have downloaded and opened the catalog, go to the CISC coded courses within the Course Descriptions section. Become familiar with the courses, reading the descriptions if you would like, but noticing the prerequisites required for each of the courses. All of the courses in CISC will become the domain of discourse that we will use in our logic program. Note that you might also need to take a look at some of the MATH courses too, but do not go any farther than this.

**Planning:**

Now that you are somewhat familiar with the courses in CISC and the prerequisites of these courses, there are a few constraints that you should consider before jumping into the programming. Come back to these after you've read the entire lab.

- Only code facts for the CISC courses. i.e. CISC courses may have prerequisites of other programs, but you are not required to find the prerequisites of any course outside of CISC.
- To reiterate, the domain consists of all CISC courses and some of the MATH prerequisites of those CISC courses.
- Note that some courses may have complications within the prerequisites. It will be sufficient to handle the following conditions as below:
    - Course grades (CISC 120 with a C or better)
        - The grade aspect may be ignored
    - Options for course prerequisites (CISC 120 or CISC 140)
        - You may choose one of the courses or turn the "or" into an "and," functionally denoting both courses as prerequisites.
    - Prerequisite that isn't a course (60 semester hours of courses completed)
        - Code it in a logical way.

**The Problem:**

1. The problem we are setting out to solve is to create a logic program that contains all the facts regarding which courses are prerequisites of the CISC courses.
2. Next it will be required to write a query such that the query will return ALL courses required to take in order to take the course given in the query.

For example, if we have the following ***facts*** (written in pseudocode here) included in your knowledge base:

CISC 120 is a prerequisite of CISC 211

CISC 211 is a prerequisite of CISC 301

CISC 120 is a prerequisite of CISC 160

CISC 160 is a prerequisite of CISC 233

CISC 233 is a prerequisite of CISC 301

After you have constructed an ***if-condition*** (omitted) that logically computes prerequisites, you can construct a ***query*** (written in pseudocode here) such as:

-? What are the courses I need to take to take CISC 301?

Which should expect a result such as:

CISC 120;

CISC 211;

CISC 120;

CISC 160;

CISC 233.

Please not that the form and order of the records returned in the example is not 100% precise. The form and order will be determined by the specific form of the facts of your universe and the order that the facts unify to the query. Additionally, note that if the same course is the prerequisite for two different courses in the series of prerequisites, this record will return multiple times. (See: CISC 120 being a prerequisite for both CISC 160 and CISC 211 and returning twice above.)

**Submission Requirements:**

The files for the database as well as an accompanying document explaining the design of your solution. While this document does not need to be formalized in any way, it should explain how to use all aspects of your submission. This includes the general form of facts your database and what they mean, how to query the database for the recursively generate course prerequisites, and, if you decide to expand your application (see below), what any additional facts you add are, what those facts mean, how to use any additional functionality added, and how to understand any additional output if it is not obvious from the previous descriptions or the output itself.

**Extra Credit:**

As this is a different programming paradigm, this lab is designed to be accessible and completed by students who are having trouble adjusting to a declarative approach. This means that students who have experience with or are just able to adapt to the approach quickly may find that you complete this lab quickly and have much time leftover. Even if this is the case, we want to encourage you to continue to think about the declarative paradigm and the design of the system.

Therefore, you may continue to develop the database as you see fit. You may expand your system in ways that are logical to the initial database. (A great example of this would be to, in some way, be able to express the classes that you have already completed. You could then create a query that determines the classes that you are eligible for next. Logical functionality of such an expansion would be that any class returned would not be in your database as already completed but would have all recursively defined prerequisite courses completed.)

You may expand your rule sets and conditions to accommodate these changes. However, anything added to the lab should not interfere with the database or query structure for the main lab deliverable. Additionally, anything added to the functionality should be described in the lab's accompanying documentation, as described above. Failure to do so will result in points being deducted from the lab's points for documentation and may result in a reduction to any extra credit given for the un/poorly documented features.

**Grading:**

The requirements explicitly described in this lab are the explicit focus. And while we do want you to explore new and additional options, features, and aspects of the language and paradigm, do keep in mind that any extra credit that you receive for additional features will be secondary to the points for the primary use case. You will have significant difficulty "coding your grade" out of a hole if the required facts and queries are inaccurate or fail to work as specified.

The value of any particular expansion or extension of application functionality will be at the discretion of the instructor and will represent a significantly smaller portion of the grade than the required aspects of the lab.