CSC 435 Spring 2014

Assignment 7 – Explore Logic Programming and Prolog

Due: April 29, 2014 by 11:59 p.m.

Grade: 60 points

Objective:

The objective of this assignment is for students to gain an understanding of major concepts in the logic programming paradigm, by answering questions about logic programming and Prolog in particular, and implementing solutions to simple programming problems.

Requirements:

You may discuss this assignment with your classmates, especially the students in the Prolog group, but you will complete and submit this assignment individually. You are also encouraged to ask questions on Piazza.

The written parts of the assignment should all be submitted in a single document.

Each programming problem will have its own ***.pl*** file, designated with a specific name. Create a script file to demonstrate that your solution for each programming problem works as per the specifications. You may choose to have separate script files for each problem.

See the **Deliverables** section below for more details on submission.

Grade:

Part 1 - Prolog Basics. (15 points)

Part 2 - Facts and Inferences. (15 points)

Part 3 - Graph Coloring. (15 points)

Part 4 - Recursion and Trees. (15 points)

Deliverables:

Submit on Canvas in the "Assignment 7" category, a zip file containing:

- All source code files.
- Script file(s) showing the correct loading and execution of your Prolog code.
- The written responses in a doc, docx or pdf file.
- A readme.txt (text) file detailing the programs that do and do not function properly (organized by Part)

Details:

Part 1: Research Questions

1. Describe the logic programming paradigm. What types of projects is it most useful for?
2. Name the three broad categories of objects in Prolog. Describe each category and the types of data best represented by each one.
3. Discuss the similarities and differences between compounds and data collections. Describe the two major data collections in Prolog.
4. Explain the concept of homoiconicity and how it pertains to Prolog.
5. What is tail recursion, and why should it be used? Demonstrate, with an example, why it should be used when possible.
6. Can concurrency be handled in Prolog? Compare and contrast with other languages we have discussed in this class.

CSC 435 Spring 2014

Part 2: Facts and Inferences

Create a knowledge base in a file, named `people.pl`. Express the following ideas.

- There are 4 people, Mary, Joe, Alex, and Carl.
- Mary and Alex are female, the rest are male.
- If someone is watching TV, they are happy.
- If a person is a biology major, has a job, or has children, they are busy.
- If a person is busy for two or more reasons, they do not get any sleep.
- If a male and female are married, they have children.
- If people are married, they are also happy, but not if they do not get sleep.

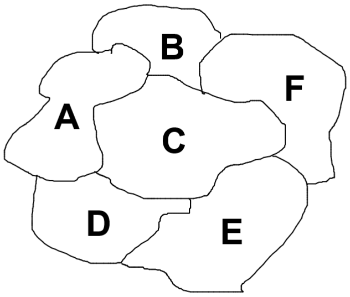
Assume that Mary and Carl are a married couple, Mary has a job, Joe is a biology major and is watching TV, and Alex does not get any sleep.

Find and express the following:

1. Is Mary happy?
2. Must Alex have a job?
3. Can any married person be happy?

Part 3: Graph Coloring:

Given 4 colors (red, green, yellow, and blue) and a series of statements, develop a program to check that a coloring for the graph has no bordering regions of the same color. Make sure your solution has facts to note which regions border, and which colors they would be. To test your solution, consider the following scenario:



Create a file, `map.pl`, to show programmatically that there is or is not a four-color solution.

Part 4: Recursion and Trees:

Create a file called `fib.pl` and write a function called `Fibonacci` (or `Fib`) to evaluate the Nth number of the Fibonacci sequence.

Then, write the clause trees for these two inputs:

```
Fibonacci (2, 99).

Fibonacci (3, N).
```

Demonstrate that the program works correctly.