# Midterm 330-002 Fall 2019

Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(on back as well)

1. For the following KB, what are the queries for a,b, and c. It is fine if your query “returns” (sets) variables; that does not change the success or failure of a query if that is all that is requested.

**%book( stock# , title, edition, year). – stock # is a unique identfier**

**book( id1942, 'The C Programming Language', 1, 1978 ).**

**book( id2491, 'The C Programming Language', 2, 1988 ).**

**book( id9791, 'Pride and Prejudice' ,1 , 1813 ).**

**%... [remember, “…” means “many more like this”]**

**%author( stock#, fname, lname).**

**author( id1942, brian , kernighan).**

**author( id1942, dennis, ritchie).**

**author( id9791, jane, austen).**

**%...**

**%in\_stock(stock#, yes/no)**

**in\_stock( id1942 , no ).**

**in\_stock( id2491, yes).**

**In\_stock( id9791, no).**

**%...**

* 1. (5) **Query** for “what is the title of book with stock# **id9791**”?

**?-**

* 1. (5) **Query** for “what books are in stock and published in the year 1900?”.

**?-**

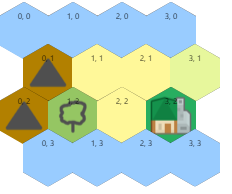
* 1. (5) **Query** for “is there any book where the same edition was published in multiple years?”

**?-**

1. Describe how to represent a list of historic events such as the following.

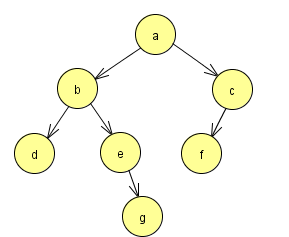
* First Continental Congress: September 5, 1774
* Signing of Articles of Confederation: November 15, 1777
* […]
* Gulf of Tonkin Incident, August 2, 1964
* […]
  1. (5) Represent the layout in a manner appropriate for solving b and c. Ellipses, “%…” are fine, obviously, if your pattern is obvious. Show the above three, at least.
  2. (5) Define a rule, **before(EventA,EventB)** that succeeds if EventA occurred before Event B chronologically (assume events on the same day occur simultaneously, not before).
  3. (5) Define a rule for **between(A,B,C)**, such that event B happens between events A and C. You cannot assume A happened before C. You may assume a working solution to b, above, i.e. that you have a working **before(A,B)** rule.

1. (15) Map Coloring: Solve the map coloring problem for the following Hexagonal map.
   1. The top and bottom rows of hexagons should be **sea** tile.
   2. The middle eight hexagons can be anything *but* **sea**.
   3. No two **town**s can be adjacent.



**tile(town). tile (sea). tile (mountain). tile (forest). tile(desert).**

1. Trees, in this case binary trees, are recursive structures, so naturally it’s easy to work with them in Prolog. (these are not binary *search* trees). Write the descendant rule for binary trees:

****

**%EXAMPLE tree**

**node(a,b,c).**

**node(b,d,e).**

**node(c,f,nil).**

**node(d,nil,nil).**

**node(e,nil,g).**

**node(f,nil,nil).**

**node(g,nil,nil).**

* 1. (5) Define a **root(R)** rule, where R is the root of the tree.
  2. (10) Define a **leaf\_of(Leaf,Node)** that returns true if **Leaf** is a leaf node in one of the **Node**’s subtrees.

1. (10) [Extra Credit] Write a **begins\_and\_ends** rule that holds if a list begins and ends with the same item.
2. (3) A language with high orthogonality is going to have a \_\_\_\_\_\_\_\_\_\_\_\_\_\_ number of similar/redundant features, e.g. to add numbers, perform iteration, etc.
3. (3) Prolog is an example of a(n) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ language.
4. (3) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ is an example of an imperative language.
5. (3) Define “binding”?
6. (3 ) What is the difference between late and early binding?
7. (3) The syntax of most programming languages are specified with a \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ grammar.
8. (3) Checking that the types of variables and the values being assigned the them are correct, for a statically typed language, is done by the \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ stage of compilation.
9. (3) The output of the lexer, the program that perfoms lexical analysis, is …
10. (6) Give the parse tree for **bbdxxggxx** from **A**,as the start symbol. The alphabet is .

**A->BG**

**B->bB | C**

**C->cC | D**

**D-> dD | ε // ε is “nothing”, the empty string**

**G->xGx | gGg | // ε is “nothing”, the empty string**

1. (5) Give EBNF for a string having any number of **a**s and **b**s but having **ab**  as a substring.
   1. Positive Examples: **ab, aaabbbb, ababababababba**
   2. Negative Examples: **ba**, **bbbaaaa**, **ε**
2. (5) Match a the EBNF on the left to the strings they match on the right.

**xyzzzyx**

**[y]{x|y|z}x**

**zyxxxyz**

**(y|z) {x|y|z}**

**yxzyxzyx**

**{x|y}y{y|z}**

**xxxx**