CSCI 4525 / 5525 **Name: ________________________________**

Introduction to Artificial Intelligence
Midterm Exam Study Guide

The exam is made up of four sections. Each section is worth *approximately* 25% of the total exam (plus or minus a point here or there).

Section 1: Short Answer

This section will ask you a number of questions, where you need to write 2-3 sentences answering them. These questions aren't meant to be tricky! If the answer seems straightforward to you, it probably is!

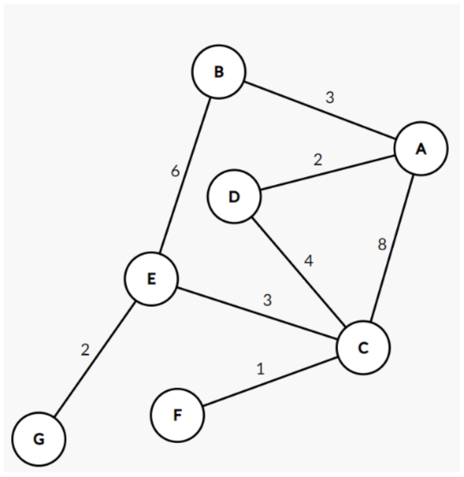
The catch is that anything from the semester is fair game to ask here. Things that we've talked about, split up by lecture, include:

1.) The Philosophical Underpinnings of AI
2.) Agents and Environments
3.) Uninformed Search
4.) Informed Search
5.) Adversarial Search
6.) Local Search and Genetic Algorithms
7.) Constraint Satisfaction Problems
8.) Logic (intro to propositional, predicate, and first-order logic, with Boolean logic)
9.) Model Checking (our introduction to satisfiability problems)
10.) Logical Inference (our introduction to unification and resolution theorem proving)

Again, for this first section, you don't need to worry about knowing anything *too* deep in any topic, but having, at the very least, a surface level understanding about what each topic is about will serve you well.

Section 2: Graph Search

Consider the graph below. Each node is labeled with a letter. Each edge is labelled with a number that represents the cost of traveling along that edge. Also provided is a heuristic, *h*, which estimates the distance between any given node and node G. It won't be the following graph exactly, but you can imagine something like it:



h(A, G) = 11

h(B, G) = 9

h(C, G) = 5

h(D, G) = 4

h(E, G) = 1

h(F, G) = 1

Assume you always start at node A and you always have the goal of reaching node G. Also assume that we are tracking which nodes have been visited and we never visit the same node twice. You do not need to track the frontier for this question, though you may find it helpful.

For each kind of search below, list the nodes in the order they would be visited. When you are faced with more than one node that could be visited next, always pick the node that "comes first" in alphabetical order (i.e., all else being equal, visit B before C). If you reach node G before visiting all nodes, stop. In other words, you may not fill in every black space for search.

For each search, if there are implementation details that you believe would change the result (e.g., using a Priority Queue over a FIFO Queue), please mention any implementation assumptions you are making, and I will grade accordingly (I don't believe it will impact).

Search 1	**Search 2**	**Search 3**
________	________	________
________	________	________
________	________	________
________	________	________
________	________	________
________	________	________
________	________	________

Section 3: First Order Logic and Conjunctive Normal Form

For each of the following first order predicate logic expressions, please write a (reasonable) interpretation in English, and then convert it to an equivalent expression in conjunctive normal form. That means no quantifiers and no implications! Use lower-case letters to indicate variables and predicates, and capitalized words/upper-case letters to represent constants and functions.

Expression 1: $Some\ Beautiful\ expression\ in\ FOPL.$

In English: ____

CNF: ____

Expression 2: $Some\ Other\ Beautiful\ expression\ in\ FOPL.$

In English: ____

CNF: ____

… and so on for some number of expressions.

Section 4: Resolution Theorem Proving

Given a knowledge base in first order predicate logic (already in conjunctive normal form and standardized apart), prove a query using resolution. For each step in the process, do the following:

1.) Identify which two expressions are being used by their numbers.
2.) Write out the two complimentary literals (before unification) which are being resolved
3.) Write the unifier needed to make the literals complementary.
4.) Write the resulting combined expression (substituting variables whose values are known in the unifier).

Please observe the following:

1.) Note the query has already been negated and added to the knowledge base. You should not modify the initial knowledge base (beyond adding to it via resolution, of course).
2.) You must complete the proof in the number of steps provided. You may want to sketch out a rough draft on the back of this paper first before filling in your final answer.
3.) You may only resolve two literals at a time (i.e., one from each clause).
4.) Remember about factorization: If a clause contains two instances of the same literal, you may remove one instance of it. For example, if a clause would be $X \vee Y \vee Y$ you may re-write it as simply $X \vee Y$ *Hint: You will have to do this at least once.*
5.) Everything is already standardized apart – you won't have to worry about that here.

The Problem:

[Imagine some light-hearted English sentences that establish the knowledge base and ask you to prove something]

Knowledge Base:

- [A knowledge base in FOPL]
- [This is a translation of the problem above.]
- [You don't need to make this yourself! It's just given to you!]

Query:
- [The Thing The Problem Asks You to Prove]

Knowledge Base in Conjunctive Normal Form with Negated Query:
1. [This is the knowledge base above, translated into CNF]
2. [Each clause on its own line]
3. [You don't need to do this either, it is given to you!]
4. [The last item will be the negated query above]

[THIS IS THE PART THAT YOU *DO* DO]

Step 1:

Literal ________________________________ from clause #__________ resolves with
literal ________________________________ from clause # __________
via unifier ____
to produce clause #X: ____

Step 2:

Literal ________________________________ from clause #__________ resolves with
literal ________________________________ from clause # __________
via unifier ____
to produce clause #X+1: ____

And so on… until…

Step ?:

Literal ________________________________ from clause #__________ resolves with
literal ________________________________ from clause # __________
via unifier ____
to produce the empty clause! Q.E.D!