CMSC671 Midterm Examination
March 23, 1994

Name ______________________________
SS# ______________________________

1 Unification - examples (10 points)

For each of the following pairs of terms, say whether or not the unification would succeed in Prolog and, if it does, what the most general common instance would be. Assume that the same variable name in two different terms represents the same variable.

1. X , loves(john,mary).
2. p(X,q(Y),r(Z)) , p(r(a), q(X), r(b))
3. loves(john,X), loves(Y,Y)
4. and(father(P1,P2),brother(P1,P3)) , and(father(X,john),brother(john,john))
5. foo(X), X

2 True/False (10 points)

Instructions: Place a T or an F in the space before each statement to indicate whether the statement is True or False. There is no penalty for incorrect answers (but then, there are no points for incorrect answers either).

1. ___The *mini-max* and *alpha-beta* procedures will **always** back up identical values to the root node of a game tree.
2. ___The major drawback of *Hillclimbing* is that it only works for two-dimentional search spaces.
3. ___It is possible for Algorithm A^* to expand a non-optimal solution node.
4. ___Algorithm A will perform a breadth-first search if $F(n) = G(n)$.
5. ___The main advantace of the Iterative Deepening A* algorithm over the standard A* algorithm is that its memory requirement is only linear in the depth of the search.

3 Unification - occur check (10 points)

What is the occur check in Unification? Does Prolog use it? Describe one advantage and one disadvantage to including it in the Unification algorithm.

4 Simple Prolog (10 points)

Suppose a Prolog database contains assertions of the form:

- **person(P)** - true if P is a person.
- **likes(P1,P2)** - true if P1 likes P2.

Write a Prolog goal which will find the *most liked* person in the database. If several people share this honor, then your predicate should find all of them on backtracking. You must define any auxiliary predicates you need, with the exception of the standard list processing predicates $append/3$, $reverse/2$, $length/2$ and $last/2$.

5 Cut (15 points)

Let the predicate *between*$(N1, N2, N3)$ be true iff $N1 < N2 < N3$ (assume all three arguments are integers). Consider the following definition of *between*/3:

```
% minMember(L,X) is true iff X is the smallest integer in the list of
% integers L.

minMember([Head|Tail],X) :- minMember1(X,Head,Tail).

minMember1(X,X,[]).
minMember1(X,SmallestSoFar,[Head|Tail]) :-
    Head<SmallestSoFar,
    !,
    minMember1(X,Head,Tail).

minMember1(X,SmallestSoFar,[Head|Tail]) :-  minMember1(X,SmallestSoFar,Tail).
```

- Is the cut in minMember/3 an instance of a *red* or a *green* cut? Recall that a *green* cut is one that does not change the semantics of the program, only its space-time performance and that a *red* cut does change the semantics.
- Describe what will happened for each of the following goals. If a solution is found, describe what would happen on backtracking.
  1. minMember([1,2,3],X).
  2. minMember([1,2,3],2).
  3. minMember(X,2).
- Rewrite the predicates without using the cut symbol.

6 Mystery predicate (15 points)

Consider the following mystery predicate:

```
p([],[]).
p([A|B],C) :- p(B,D),append(D,[A],C).
```

where append has the usual definition.

- Describe this predicate in a simple sentence.
- Describe what would happen for each of the following queries in terms of the kinds of errors that would result or the solutions(s) that would be found. If more than one solution would be found, characterize them all.
  - p([a,b,c,d,e],X).
  - p(X,[a,b,c,d,e])
  - p(X,Y).

7 Graph Search (15 points)

Consider the state space with the following nodes and arcs:

from	*to*	*cost*
A	B	3
A	C	2
A	D	4
B	E	3
C	E	5
D	F	1
E	G	2
F	G	4
G	Z	3

Assuming the initial state is A and the goal state is Z:

- What is the cost of the cheapest solution.

- In what order would the nodes be expanded if we use the graphsearch algorithm and a heuristic function $H(n) = 0$.

- In what order would the nodes be expanded if we use the graphsearch algorithm and a heuristic function $H(n) = -2 * G(n)$.

8 Alpha-Beta Algorithm (15)