CS479/CS579 – Test 1

5pm – 6:10 pm

October 27, 2003

Note:

- There are 3 questions (100 points: 100 points for 30% of the total grade).
- Be short and precise in your answers. (Remember: Time is precious and writting takes time.)

Name:

Signature:

Grades Points Total: ....

1. ....
2. ....
   2.1 ....
   2.2 ....
   2.3 ....
   2.4 ....
3. ....
   3.1 ....
   3.2 ....
   3.3 ....
   3.4 ....
   3.5 ....

1. (10 points) Write a Prolog program that defines the predicate $max(?X, +Y)$ which is true if and only if X is a maximal element belonging to the list of integers Y. You can use the predicate $length(N, Y)$ – which is true if and only if N is the number of elements in a list of integers Y – in your program.

2. (40 points) Given a set of students `ana, peter, john, ema, tyler` and a set of facts of the form `like(X,Y)` and `dislike(X,Y)`:

```
like(ana, peter).
like(peter, john).
like(john, ema).
dislike(ema, tyler).
```

Intuitively, `like(X,Y)` states that X likes Y while `dislike(X,Y)` states that X does not like Y.

2.1 Define the predicate `friend(X,Y)` that is true if and only if X likes Y or Y likes X or X and Y are friends of someone. What would be the answer of your program to the query `friend(ana, john)`? Explain the answer using your program.

2.2 Define the predicate `enemy(X,Y)` that is true if and only if X dislikes Y or Y dislikes X or X has a friend who is an enemy of Y. What would be the answer of your program to the query `enemy(ana, john)`? Explain the answer using your program.

2.3 Suppose that we replace the fact `dislike(ema, tyler)` with the rule

```
dislike(X,Y):- not like(X,Y)
```

What is the answer of your (new) program to the above two queries? Explain the answer using your program. Do you think that this modification is good? Explain your answer.

2.4 The rule in 2.3 will yield incorrect answer for the query `?  dislike(ema, table)`. What should we do to avoid this problem?

3. (50 points) Consider the story:

> A coffee delivery robot is working in the CS building. Its job is to bring coffee from the front office (room 100) to the professors's offices whenever there is a request. The robot has a map of the building and can navigate from one place to another place with ease. Coffee is available only in room 100. The robot has only one cup holder and therefore can carry only one cup of coffee at anytime. The robot should go back to room 100 after the completion of each delivery. Initially, the robot is in the room 100. Coffee is available anytime.

3.1 Develop a situation calculus theory for this domain. Your theory should be able to answer questions like: "What should the robot do when Prof. A requests for a coffee?" or "How do I (the robot) get back to the room 100?" etc. You can use predicates $room(X)$ to denote the room number X and $nextTo(X, Y)$ to denote that room X is next to room Y. For simplicity, we will assume that the rooms are arranged in a straight line and are numbered from 100 to 110.

3.2 Write a GOLOG program, say $control$, that tells the robot what to do given the information available in the initial situation (An example of the initial situation: the robot is at the room 100, there is a request at room 102, and the robot does not have coffee). Use the notation you are familiar with.

3.3 Suppose that we attach to the robot a container that allows the robot to carry finitely many cups of coffee. What should be changed in the situation calculus theory to reflect this change?

3.4 How does the modification in (3.3) affect your GOLOG program in (3.2)?

3.5 What should be changed in your GOLOG program so that the robot can delive more than one cup of coffee if necessary?