Lab 2

Logic, Reasoning and Inferring

Activity 1: Basic of Logic. In this activity, we propose to practice basic of propositional logic.

A) Say whether each of the following expressions is a syntactically legal sentence of Propositional Logic.

- $p \wedge \neg p$
- $\neg\,(q \vee r)\,\neg q \implies \neg\neg p$
- $(p \wedge q) \vee (p\neg \wedge q)$

B) Using Logical Operators: Given the following list of defined propositions:

- a: *The store is open today*.
- b: *Mary is going to the store today*.
- c: *John is going to the store today*.
- d: *John is happy*.
- e: *Mary is happy*.
- f: *Paul is happy*.

Translate the following sentences into the logical notation.

- *John is going to the store today, but Mary isn't*.
- *The store is open today, and either John or Mary is going*.
- *If neither John nor Mary is happy, then Paul is happy*.

C) Translating Into Logic: Translate the following assignment from natural language to propositional logic:

- *If Charles was clever, he'd have a job*.
- *To pass philosophy it is not necessary to make notes every week*.
- *There is still some soup in the fridge if you want*.

D) Bi-conditional statement: prove that

$$p \Leftrightarrow q \text{ and is not same than } (p \implies q) \vee (q \implies p)$$

using truth tables.

Activity 2: First steps in SWI-Prolog. In this activity, we propose to introduce some of the central concepts of Prolog by accessing and running Prolog with simple examples. Here a list of useful commands:

- Under Windows, SWI-Prolog installs a start icon that can be double-clicked to initiate the interpreter.
- A Prolog goal is terminated with a period `"."`
- When the Prolog system is started, you will see a goal prompt, usually in the following form: `?-`
- SWI Prolog has extensive help information. To learn more about it, try: `?- help(help).`
- The 'halt' goal always succeeds (stop the Prolog system) and returns the user to the operating system, try: `?-halt.`
- To suspend the execution of the program, try: `?- break.`
- To find out the actual contents of the Prolog database by using the listing command: `?-listing.`
- To write a message on the screen, try the function `write('message')`
- A prolog program is a text file with a .pl ending. For example, program.pl. You may need to specify the path.
- To load a program program.pl, try: `?- consult(program.pl).`
- To save a program into a file program.pl, type the following directive. Try: `?- save(program.pl).`
- To specify a goal to be run when a saved program is restored. Try: `?- save(program.pl,start).`
- Once a program has been saved into a file program.pl, the following directive will restore the system to the saved state: `?- restore(program.pl).`
- A comment is specified by the character `"%"`

A) Greatest Common Divisor. Using Euclid's algorithm, we can compute the GCD of two positive integers in Prolog as follows:
gcd(X,Y,G) :- $X = Y$, $G = X$.
gcd(X,Y,G) :- $X < Y$, $Y1$ is $Y - X$, gcd(X,$Y1$,G).
gcd(X,Y,G) :- $X > Y$, gcd(Y,X,G).

Create a text file gcd.pl containing this program and load this file with Prolog (using the interface or by command). What is the result of the following query ? ?- gcd(5,10,D).

B) Factorial. Write in Prolog the function Factorial(A,B) where B is the factorial of A.

C) Lists. Write a logical predicate TakeOut(X,[1,2,3,4],Y) that asks that X be taken out of list [1,2,3,4] leaving remainder list Y, in all possible ways.

Activity 3: Procedure, Facts, Rule, Clauses and Queries. In this activity, we propose to implement ...

A) Clauses and Listing. Create a text file lesson1.pl containing the clauses:

- has(jack,apples).
- has(ann,plums).
- has(dan,money).
- fruit(apples).
- fruit(plums).

Load this file with Prolog. Explain what the following goals do:

- ?- [lesson1].
- ?- listing(fruit).
- ?- has(jack,X).
- ?- has(jack,_).
- ?- has(X,apples),has(Y,plums).
- ?- has(X,Y),not fruit(Y).

B)
We define the following rule:

meal(X,_) :- food(X).

What does it mean? Redefine the rules meal(X,Y) to specify that a meal is composed of a starter, side and drink. Rules. Below food table shows the facts:

- food(burger).
- food(sandwich).
- food(pizza).
- food(spaghetti).
- brevage(lemaonde).
- brevage(water).
- lunch(sandwich).
- dinner(pizza).
- starter(lunch, sandwich).
- starter(dinner, spaghetti).
- side(lunch, chips).
- side(dinner, bread).
- drink(lunch, lemonade).

- drink(dinner, water).

their english meanings are: burger is a food, sandwich is a food, pizza is a food, sandwich is a lunch, pizza is a dinner. Create a text file containing these facts.
We define the following rule:

```
meal(X,_) :- food(X).
```

What does it mean? Redefine the rules meal(X,Y) to specify that a meal is composed of a starter, side and drink.

C) Queries. Express the fallowing queries in Prolog. What is the result?

- Is pizza a food?
- Which food is meal and lunch?
- Is sandwich a dinner?

Regarding this results, what kind of mechanism do you think Prolog implements to resolve the queries?

Activity 4: Limitation of Predicate Logic. In this activity, we propose to implement a small program to test the limitations of Predicate Logic using Prolog.

A) Write in Prolog the instructions expressing the fact that: *Socrates and Plato are human*.

B) Write in Prolog the instructions expressing the rule that: *if x is human, then x is mortal*.

C) Write in Prolog the followings queries:

- ask whether Socrates and Descartes are mortal
- who are all mortal beings the system knows

D) What are your conclusions about predicate logic as a tool for representing linguistic knowledge?