# REPORT

In this assignment our goal was to create an interpreter for prolog language. A database is given to us (**program.pl** used here). We created a table of atoms and rules out of it (**table** used here).

Whenever given any query , our interpreter return either **Yes** or **No** or a set of answers. If any variable is involved the output is a set of answers mapped to the variables which were asked by the user. If any answer is not found the program returns **No**.

The interpreter consists of a lexer (**lexer.mll**), a parser(**parser.mly**) and a backend program(**tree.ml**). First of all we read the database and create our table then we parse through the input. In case of rules, we need to check multiple conditions ,which are separated by “,” , interpreted as and in prolog. The interpreter returns possible values of the variables after searching through the database one by one. A query gives the first set of possibilities. Using “;” we could access other possibilities too. Using”.”we could ens the existing one and start a new query.

**TEST CASE**

INPUT

father(vin,aby).
father(ral,nic).
father(chris,vin).
father(vin,sky).
mother(gina,vin).
mother(nic,aby).
mother(ala,nic).
mother(nic,sky).
male(aby).
female(nic).
female(ala).
female(gina).
female(sky).
male(ral).
male(vin).

```
male(chris).
son(X,Y):-father(Y,X),male(X).
grandfather(X,Y):-father(X,K),father(K,Y).
grandson(X,Y):-son(X,K),son(K,Y).
married(X,Y):-father(X,K),mother(Y,K).
```

These are some of the queries and their output values.

```
?-grandfather(X,Y).
(X:chris); (Y:sky);
?-;
(X:chris); (Y:aby);
?-;
No
?-grandfather(chris,B).
(B:sky);
?-;
(B:aby);
?-;
No
?-.
?-father(chris,aby).
No
?-father(chris,vin).
Yes
?-male(aby).
Yes
?-son(vin,chris).
Yes
?-married(vin,nic).
Yes
?-married(nic,vin).
No
?-grandfather(vin,Y).
No
?-grandfather(chris,Y).
(Y:sky);
?-;
(Y:aby);
?-;
No
?-.
```