**Group Activity 4; CS 3060**

**Names** of students in your group: **Sidney Sanders, Amanda Collert, Quinci Drain, Madeline Reeves, Emily Endlish**

Points: 10

**Task 1**: (7 points) Let’s first build a knowledge-base in Prolog, representing the following family tree.

grandma

man1

man2

woman1

girl1

boy1

boy2

boy3

girl2

girl3

Let us use the following skeleton of code. Let’s say the prolog file name is family.pl

female(grandma). /\* this fact tells Prolog that grandma is a female \*/

female(girl1).

male(man1).

… /\* add more facts like above. How many such facts are there? \*/

descParent(boy1, man1). /\* this fact codes a desc-parent relationship \*/

… /\* add more facts like above. How many such facts are there? \*/

Now implement the following rules in the same prolog file (family.pl). Note that your rules should be generic and should work for any family tree (that is written in similar format).

*isGrandParent(X) :- descParent(Z, Y), descParent(Y,X) /\* checks if X is a grand parent \*/*

*grandParent(X, Y) :- /\* this rule checks if Y is a grand parent of X \*/*

*nephewUncle(X, Y) :- /\* this rule checks if Y is an uncle of nephew X \*/*

*hasDescendant(X) :- /\* this rule checks if X has any descendant \*/*

For each of the above rules, show (in README) the result of a sample (interesting) query.

**Task 2**: (3 points) Say there are 3 baseball teams (tiger, lion, and panther), and each pair of teams play two matches (home and away). Say, as the game scheduler, you need the list of all of the possible matches. Write Prolog code to serve your purpose. You may build on the following code.

*team(tiger). team(lion). team(panther).*

*match(X, Y) :-*

Test your code and show the result in README.

**Submission**: Submit one copy (per group) of the prolog code (that can be shown on this document itself) and a README file to Canvas.