Homework #4: Chapters 9 and 10

1. [2 pts. each]
   a) \( P(B,A,B), P(x,y,z) \)
      \[ x/B, y/A, z/B \]
   b) \( P(x,x), Q(A,A) \)
      failure, predicate \( P \) and \( Q \) don’t match!
   c) \( Q(y,G(A,B)), Q(G(x,x),y) \)
      failure, need \( y/G(x,x) \) and \( y/G(A,B) \), but \( x \) can’t be both \( A \) and \( B \)
   d) \( \text{Older(Father(y),y)}, \text{Older(Father(x),John)} \)
      \[ x/\text{John}, y/\text{John} \]
   e) \( Q(G(y,z),G(z,y)), Q(G(x,x),G(A,B)) \)
      failure, need \( y/B \) and \( z/A \) and both=\( x \)!

2. First, we standardize apart the sentences, and give each a label:

   S1: \( \forall x, y \ P(x_1,y_1) \Rightarrow T(x_1,y_1) \)
   S2: \( \forall x, y \ P(x_2,y_2) \Rightarrow T(y_2,x_2) \)
   S3: \( \forall x, y, z \ T(x_3,y_3) \wedge Q(y_3,z_3) \wedge S(z_3,x_3) \Rightarrow R(x_3,y_3,z_3) \)
   S4: \( P(\text{Red}, \text{Blue}) \)
   S5: \( P(\text{Green}, \text{Red}) \)
   S6: \( P(\text{Blue}, \text{Green}) \)
   S7: \( Q(\text{Blue}, \text{Green}) \)
   S8: \( Q(\text{Green}, \text{Blue}) \)
   S9: \( Q(\text{Red}, \text{Green}) \)
   S10: \( Q(\text{Blue}, \text{Red}) \)
   S11: \( Q(\text{Green}, \text{Red}) \)
   S12: \( S(\text{Blue}, \text{Red}) \)
   S13: \( S(\text{Green}, \text{Red}) \)
   S14: \( S(\text{Green}, \text{Green}) \)
   S15: \( S(\text{Green}, \text{Blue}) \)

   Answers to 2a) and 2b) are shown on pages 2 and 3.
2a) [10 pts.] Roughly, 1 point per node, and 1 point for each match/substitution combination...
2b) [20 pts.] Approximately 1pt per node and 1pt. per match/substitution combination. -2 for not summarizing answers

answers = \{ \{x/Blue, y/Green\}, \{x/Green,y/Blue\} \}
a) [10pts] 35 facts, -1 for every three missing or significantly wrong, min of -1, max of -9 unless no answer given

wife(mum, george).
wife(kydd, spencer).
wife(elizabeth, philip).
wife(diana, charles).
wife(anne, mark).
wife(sarah, andrew).
wife(sophie, edward).
daughter(margaret, mum).
daughter(margaret, george).
daughter(elizabeth, mum).
daughter(elizabeth, george).
daughter(anne, elizabeth).
daughter(anne, philip).
daughter(zara, anne).
daughter(zara, mark).
daughter(beatrice, andrew).
daughter(beatrice, sarah).
daughter(eugenie, andrew).
daughter(eugenie, sarah).
daughter(louise, sophie).
son(charles, elizabeth).
son(charles, philip).
son(andrew, elizabeth).
son(andrew, philip).
son(edward, elizabeth).
son(edward, philip).
son(william, diana).
son(william, charles).
son(harry, diana).
son(harry, charles).
son(peter, anne).
son(peter, mark).
son(james, edward).
son(james, sophie).
b) [25 pts.] 22 rules, 1pt. each

husband(M,F) :- wife(F,M).
spouse(X,Y) :- husband(X,Y).
spouse(X,Y) :- wife(X,Y).

child(C,A) :- daughter(C,A).
child(C,A) :- son(C,A).
parent(P,C) :- child(C,P).

grandChild(C,GA) :- child(A,GA), child(C,A).
greatGrandParent(GGA,C) :- grandChild(GC,GGA), child(C,GC).

brother(B,C) :- child(C,A), son(B,A), \+(B=C).
% to avoid duplicates, could consider only children of one parent
% this could be accomplished by adding wife(A,_) as a clause above
sister(S,C) :- child(C,A), daughter(S,A), \+(S=C).

aunt(AN,C) :- parent(P,C), sister(AN,P).
% circular def would cause inf. loop: "aunt(AN,C) :- uncle(U,C), wife(AN,U)"
% the wife of the brother of one's parents
aunt(AN,C) :- parent(P,C), brother(B,P), wife(AN,B).

uncle(U,C) :- parent(P,C), brother(U,P).
% the husband of the sister of one's parents
uncle(U,C) :- parent(P,C), sister(S,P), husband(U,S).

brotherInLaw(BL,Y) :- spouse(S,Y), brother(BL,S).
brotherInLaw(BL,Y) :- sister(S,Y), husband(BL,S).
brotherInLaw(BL,Y) :- spouse(SP,Y), sister(SIS,SP), husband(BL,SIS).

sisterInLaw(SL,Y) :- spouse(SP,Y), sister(SL,SP).
sisterInLaw(SL,Y) :- brother(BR,Y), wife(SL,BR).
sisterInLaw(SL,Y) :- spouse(SP,Y), brother(BR,SP), wife(SL,BR).

firstCousin(F,Y) :- uncle(U,Y), child(F,U).
firstCousin(F,Y) :- aunt(AN,Y), child(F,A).
c) [10 pts., 2pts. each] Note, order does not matter below

1 ?- husband(H,sarah).
   H = andrew ;
   No

2 ?- grandChild(GC,philip).
   GC = zara ;
   GC = peter ;
   GC = william ;
   GC = harry ;
   GC = beatrice ;
   GC = eugenie ;
   GC = louise ;
   GC = james ;
   No

3 ?- greatGrandParent(GGP,peter).
   GGP = mum ;
   GGP = george ;
   No

4 ?- sisterInLaw(SL,diana).
   SL = anne ;
   SL = sarah ;
   SL = sarah ;
   SL = sophie ;
   SL = sophie ;
   No

   Note the duplicates above appear b/c anne is charles’ sister via both parents. To remove the
duplicates we would have to change the definition of sister() to only consider another child of just
the mother or of just the father. However, since this not intuitive, we choose to live with the
duplicates.

5 ?- uncle(U,beatrice).
   U = charles ;
   U = edward ;
   U = charles ;
   U = edward ;
   U = mark ;
   U = mark ;
   No

   Note the duplicates above appear for the same reasons as the previous query.
4. [15 pts.] Note, the event notation is needed to ensure that the event of competing can be properly
described. With the exception of “MemberOf”, the actual names of links and nodes need but match
precisely.
1pt each node and each link.
-4 if the information is present, but there is no connection between the founder, the year, and the
university.

![Graphical representation of the event notation]

- FoundingEvent
- IndustrialPioneer
- Philanthropist
- AsaPacker
- LehighUniversity
- 1865
- Entrepreneur