Programming Assignment 6
(Logic Programming - PROLOG)
Due date: December 10, 2013

- This assignment is to be turned in via Blackboard before 11:59 PM on December 10, 2013. All problems must be solved using PROLOG programming language. You must download and use SWI Prolog to test your programs.
- You should submit one (text) file with all your prolog programs (separated using comments). In the same file, for each problem, you should also submit a few sample runs on the Prolog interpreter (you can copy/paste your session OR use the Unix script command OR use the windows print screen feature to capture your session).

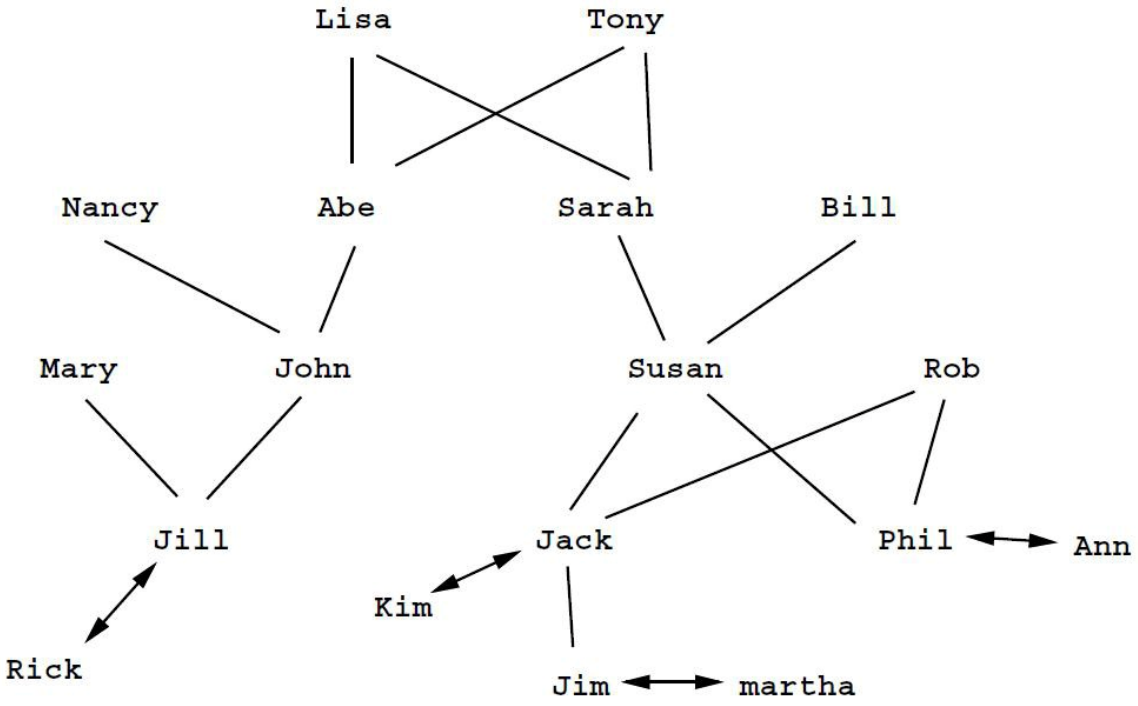

Figure1: Family Tree

1. (25 points)

Transcribe the above diagram into father and mother relationships as Prolog facts. Write a predicate called same_generation(X, Y) which succeeds if individuals X and Y are siblings/cousins of the same generation (i.e., they are descendants of a common person and both are same number of links down from the common predecessor). You

can ignore the double arrows as they depict the relationship `married' that you may not need for this problem. [HINT: Use `parent' relation to implement `same generation'. Two persons are of the same generation if they have a common parent or if their parents are of the same generation.]

---

2. (25 points)

Program the Tower of Hanoi puzzle. You must use a, b, c as your peg names. Your output should be a bunch of `move' statements like `move a to b' (which means moving the top disk from peg a to peg b). You can use Prolog's `write' predicate to output a string or a variable to the screen. The `nl' predicate outputs a newline to screen.

---

3. (25 points)

Write a predicate sumlists(L1, L2, L3) where L1, L2, L3 are lists of integers. The predicate sumlists succeeds when L3 has a list of integers where each element is the sum of the corresponding elements from L1 and L2. For example sumlists([1,2,3], [3,4,5], L3) should return L3 = [4,6,8]. Handle the case where L1 and L2 are of different lengths (the output list L3 would have the length of the longer input list, and the elements in it after the shorter list ran out would just be equal to the elements in the longer list).

---

4. (25 points)

Write a Prolog program for solving the crypt-arithmetic puzzle AM + PM = DAY. That is, write predicate solve([A, M, P, D, Y]) which binds each of the variables [A, M, P, D, Y] with values from 0 to 9 such that it satisfies the equation AM + PM = DAY. Make sure that all the variables have a different value assignment and A, P, D cannot take the value 0.

BONUS PROBLEM (30 points):

Write a Prolog program for performing Merge Sort. That is, write predicate mergesort(L, R) which succeeds when R is the sorted (using merge sort) list for the input list L. [HINT: First split L into two halves say L1 and L2. Then recursively mergesort L1 to get R1 and L2 to get R2. Now merge R1 and R2 to get R (remember that R1 and R2 are sorted).]
