**Name:**

# CS 421 — Prolog Cut Activity

Mattox Beckman

Here is a list of friends. We'll use them in the next few problems. To avoid circularity problems, assume friendship is one directional.

```
friends(a,b).
friends(a,d).
friends(b,c).
friends(b,d).
friends(c,x).
friends(c,y).
friends(d,x).
friends(d,y).
```

**Problem 1)**

What will the query `friends(A,x)` produce? In what order?

**Problem 2)**

What do you think would happen if we added the rule

```
friends(A,B) :- friends(B,A).
```

**Problem 3)**

Write a function `foaf(A,B)` that is true when `A` if a friend of a friend of `B`.

```
?- foaf(a,c).
foaf(a,c).
true ;
```

**Problem 4)**

What order will friends be listed if I submit this query? Check with a neighbor to see if you agree.

```
?- foaf(a,X).
```

**Problem 5)**

There is an interesting predicate called `var(X)` which is true when `X` is a variable (i.e., has not been unified yet).

```
?- var(X).
true.

?- var(c).
false.

?- X = a , var(X).
false.
```

Write a function `bff(A,B)` that is true when `friend(A,B)` is true, but only the very first match. You'll need both `var` and cut to make this work.

```
?- bff(a,b).
true.

?- bff(a,d).
false.

?- bff(a,X).
X = b.
```

**Problem 6)**

Here is some prolog code to flatten a list. It runs okay, but successive answers unflatten the list. Explain to your neighbor why this happens.

```
myflatten([H|T],X) :- is_list(H), append(H,T,R), myflatten(R,X).
myflatten([H|T],[H|X]) :- myflatten(T,X).
myflatten([],[]).

?- myflatten([[2,3],[3,4,[5,6],4],3],X).
X = [2, 3, 3, 4, 5, 6, 4, 3] ;
X = [2, 3, 3, 4, [5, 6], 4, 3] ;
X = [2, 3, [3, 4, [5, 6], 4], 3] ;
X = [[2, 3], 3, 4, 5, 6, 4, 3] ;
X = [[2, 3], 3, 4, [5, 6], 4, 3] ;
X = [[2, 3], [3, 4, [5, 6], 4], 3] ;
No
```

Can you use a cut operator to fix this?