# Final Exam Review

CSCE 322

Name: _____

## Instructions

Please solve the problems presented below. **Show your work to receive full credit; just an answer is not enough**. **No Approximations**.

**Question 1** ( points)

In the code below, the query `?- classmates(jane_doe, X)` will succeed three times: twice with `X = jane_doe`, and once with `X = ajit_chandra`. Show how to modify the `classmates(X, Y)` rule so that a student is not considered a classmate of him or herself.

```
takes(jane_doe, his201).
takes(jane_doe, cs254).
takes(ajit_chandra, art302).
takes(ajit_chandra, cs254).
classmates(X, Y) :- takes(X, Z), takes(Y, Z).
```

**Solution:**

```
takes(jane_doe, his201).
takes(jane_doe, cs254).
takes(ajit_chandra, art302).
takes(ajit_chandra, cs254).
classmates(X, Y) :- takes(X, Z), takes(Y, Z), X \= Y.
```

**Question 2** ( points)

What does the following Prolog program compute?

mystery01.pl

```
% ** is raising to a power
mystery(0,1).
mystery(A,B) :- 0 is mod(A,2) , C is A / 2 , mystery(C,D) , B is D ** 2 , !.
mystery(E,F) :- 1 is mod(E,2) , G is E - 1 , mystery(G,H) , F is H * 2 , !.
```

**Solution:**

$2^n$

**Question 3** (10 points)

Given the following code, what values of `Nebraska` will let the query `?- mystery3([u,n,l],[u,n,o],Nebraska).` succeed?

mystery03.pl

```
mystery3([],A,A).
mystery3([B|C],D,E):-
    member(B,D), % member is true if B is in list D
    !,
    mystery3(C,D,E).
mystery3([F|G],H,[F|J]):-
    mystery3(G,H,J).
```

**Solution:**

`[l,u,n,o]`

**Question 4** (10 points)

Modify the code provided below so that the goal `path(X,Y)` for arbitrarily already-instantiated `X` and `Y` will succeed no more than once, even if there are multiple paths from `X` to `Y`.

```
edge(a,b).
edge(b,c).
edge(c,d).
edge(d,e).
edge(b,e).
edge(d,f).

path(X,X).
path(X,Y):-edge(Z,Y),path(X,Z).
```

**Solution:**

```
edge(a,b).
edge(b,c).
edge(c,d).
edge(d,e).
edge(b,e).
edge(d,f).

path(X,X).
path(X,Y):-edge(Z,Y),!,path(X,Z).
```

**Question 5** (13 points)

Given these Prolog predicates, `mystery([2,0,0,2]).` evaluates to `true`, `mystery([2,0,0,4]).` evalutes to `false` and `mystery([r,X,n,n,e,r]).` will evalute to `true` when `X` is unified with `e`. How many attempts are made to match `helper2` during the evaluation of `mystery([a,l,a,s,k,a]).`?

mystery1145final01.pl

```
1  mystery([]):-!.
2  mystery(A):-
3      length(A,1),
4      !.
5  mystery([B|C]):-
6      helper(C,B),
7      helper2(C,D),
8      mystery(D).
9
10 helper([E],E):-!.
11 helper([_|F],G):-
12     helper(F,G).
13
14 helper2([_],[]):-!.
15 helper2([J|K],[J|M]):-
16     helper2(K,M).
```

**Solution:**

9

**Question 6** (13 points)

Given these Prolog predicates, `repeat([a,b],2,[a,a,b,b]).` evaluates to `true`, `repeat([a,b],1,[a,a,b,b]).` evalutes to `false` and `repeat([[[g,o],[b,i,g],[r,e,d]]],3,Result).` will evalute to `true` when `Result` is unified with `[[[g,o],[b,i,g],[r,e,d]],[[g,o],[b,i,g],[r,e,d]],[[g,o],[b,i,g],[r,e,d]]]`. How many times is `repeat([],_,[]).` successfully matched during the evaluation?

mystery1145final02.pl

```
1  repeat([],_,[]):-
2      writeln(repeat).
3  repeat(_,0,[]).
4  repeat([H|T],N,D):-
5      helper(H,N,A),
6      repeat(T,N,B),
7      append(A,B,D),
8      !.
9
10 helper([],_,[]).
11 helper(_,0,[]).
12 helper(C,E,F):-
13     G is E-1,
14     helper(C,G,H),
15     append([C],H,F).
```

**Solution:**

Once

**Question 7** (12 points)

Given the following code, what is one value of `Input` will let the query `?- mystery(Input,[s,a,s,a,t,c,e,w,a]).` succeed?

mystery1148.pl

```
num(4).
num(3).
num(2).
num(1).

mystery(In,Out):-
    num(Number),
    !,
    helper(In,Number,Out).

helper(A,N,B):-
    num(N),
    length(A,LA),
    LA < N,
    A = B,
    !.
helper(A,N,B):-
    num(N),
    P is N - 1,
    assistant(A,P,C),
    apprentice(A,N,D),
    helper(D,N,E),
    append(C,E,B).

assistant(_,0,[]):-!.
assistant([A|B],N,[A|C]):-
    num(N),
    P is N - 1,
    assistant(B,P,C).

apprentice(A,0,A):-!.
apprentice([_],1,[]):-!.
apprentice([_|A],N,B):-
    num(N),
    P is N - 1,
    apprentice(A,P,B).
```

**Solution:**

`[s,a,s,k,a,t,c,h,e,w,a,n]`

**Question 8** (10 points)

For the code below,

```
get([Row|Rows],Where,What):-
    length(Rows,RowsLength),
    0 is mod(RowsLength,2),
    length(Row,Cols),
    getFRow(Row,RowWhere,What),
    Where is RowWhere + RowsLength * Cols.
get([Row|Rows],Where,What):-
    length(Rows,RowsLength),
    1 is mod(RowsLength,2),
    writeln(executed),
    length(Row,Cols),
    getBRow(Row,RowWhere,What),
    Where is RowWhere + RowsLength * Cols.
get([_|Rows],Where,What):-
    get(Rows,Where,What).

getFRow([What|_],1,What).
getFRow([_|Tail],Where,What):-
    getFRow(Tail,TailWhere,What),
    Where is TailWhere + 1.

getBRow([What|T],Where,What):-
    length([What|T],Where).
getBRow([_|T],Where,What):-
    getBRow(T,Where,What).
```

For the query `?- get([[8,9,0],[7,8,5],[2,3,8]],Location,8).`, how many times is Line 10 executed?

**Solution:**

Once

**Question 9** (12 points)

For the code below,

```
get([Row|Rows],Where,What):-
    length(Rows,RowsLength),
    0 is mod(RowsLength,2),
    length(Row,Cols),
    getFRow(Row,RowWhere,What),
    Where is RowWhere + RowsLength * Cols.
get([Row|Rows],Where,What):-
    length(Rows,RowsLength),
    1 is mod(RowsLength,2),
    writeln(executed),
    length(Row,Cols),
    getBRow(Row,RowWhere,What),
    Where is RowWhere + RowsLength * Cols.
get([_|Rows],Where,What):-
    get(Rows,Where,What).

getFRow([What|_],1,What).
getFRow([_|Tail],Where,What):-
    getFRow(Tail,TailWhere,What),
    Where is TailWhere + 1.

getBRow([What|T],Where,What):-
    length([What|T],Where).
getBRow([_|T],Where,What):-
    getBRow(T,Where,What).
```

For the query `?- get([[8,9,0],[7,8,5],[2,3,8]],Whe,Wha).`, in what order are `Whe` and `Wha` given values if the user presses `;` to get all values that satisfy the query?

**Solution:**

7 8
8 9
9 0
6 7
5 8
4 5
1 2
2 3
3 8