CT331 Assignment 4

Declarative Programming with Prolog

Andrew Reid East
16280042

https://github.com/reideast/ct331_assignment4

Question 1

```
% Base Case: El is the head of List
isElementInList(El, [El | _]).
% Recursive Case: List is reduced to everything except the head of the list
% I.e. Element is in the list if element is in the tail of the list...somewhere!
isElementInList(El, [_ | Tail]) :- isElementInList(El, Tail).
```

```
9 ?- isElementInList(1, []).
false.

10 ?- isElementInList(1, [1]).
true .

11 ?- isElementInList(1, [2]).
false.

12 ?- isElementInList(1, [1, 2, 3]).
true .

13 ?- isElementInList(2, [1, 2, 3]).
true.

14 ?- isElementInList(7, [1, 2, 9, 4, 5]).
false.
```

Question 2

```
%Definitions for mergeLists(List1, List2, Merged)

% Base Case: First list is now mepty. Second list is the basis of the new, merged list
mergeLists([], List2, List2).

% Recursive Case: First list still has elements.
% List1's head (an atom) onto whatever the reduced call returns as Merged will become the new merged list
% Don't care what the second list is, but pass it on down towards the base case (the string)
mergeLists([Head | Tail], List2, [Head | Merged]) :-
    mergeLists(Tail, List2, Merged).
```

```
17 ?- mergeLists([7], [1,2,3], X).
X = [7, 1, 2, 3].

18 ?- mergeLists([2], [1], X).
X = [2, 1].

19 ?- mergeLists([1], [], X).
X = [1].
```

Question 3

```
%Definitions for reverseList(List, ReversedList)

% Base Case: Empty list; must be reversed.
reverseList([], []).

% Recursive case: List still has head and tail. Remove head, and unify it with a merged list
% Recurse on the tail
reverseList([Head | Tail], ReversedList) :-
    mergeLists(SmallerList, [Head], ReversedList),
    reverseList(Tail, SmallerList).
```

```
20 ?- reverseList([1,2,3], X).
X = [3, 2, 1] .

21 ?- reverseList([1], X).
X = [1] .

22 ?- reverseList([], X).
X = [].
```

Question 4

```
%insertElementIntoListEnd(El, List, NewList)
% This one is NOT recruseive itself.
% El is inserted into NewList IF NewList is the same as List with El merged at the end
insertElementIntoListEnd(El, List, NewList) :- mergeLists(List, [El], NewList).
```

```
13 ?- insertElementIntoListEnd([2], [1,2,[3,4]], X).
X = [1, 2, [3, 4], [2]].

14 ?- insertElementIntoListEnd(7, [1,2,3], X).
X = [1, 2, 3, 7].

15 ?- insertElementIntoListEnd(2, [1], X).
X = [1, 2].

16 ?- insertElementIntoListEnd(1, [], X).
X = [1].
```