BC General Notes 4

My program is very similar to the pseudo code that Jim gave for us to use, I use the same idea to use a goals list. At the start the query is the only atom in the goal list, and as we go through iterations of the code we add and remove atoms from the list until either none remain, or the query can't be proved. I use two different for loops, the first is to go through the rules list and find rules that have the same head as the top of the list. The second for loop is where we do the recursion, recursing for the atoms in the body as part of the goal list.

Limitations:
The limitations on my backward chaining code are much more so than my forward chaining code. My code does not keep track of what is already known, so you can often see it trying to prove things in later iteration that is has already proved earlier. The consequence of this is not only time but also the chance of creating infinite loops. For example in testcase2, if the query = l instead of a, it will create an infinite loop. This occurs when the rules list contains rules where the query is in the head(r1) and the body(r2). Basically it will keep trying to prove the same things over and over creating an infinite loop.