For my implementation, I created two inner classes used to store the data: Atom and Sentence. An Atom contains the variables, name, and the substitutions made based on the rules entered. A sentence contains a list of Atoms, an array that keeps track of which Atoms experienced a substitution from the rules/facts entered, and the Atom on the right hand side of the FOL equation. For each rule/fact entered, I then loop through each sentence and each Atom in that sentence to look for a substitution using the Unify algorithm provided in the project description. Each time the number of substitutions was equal to the number of Atoms in a sentence, an inference could be made. Once all the inferences were made, it was possible to again loop through the Sentences to assert the goal state was reached.

The incremental version follows the same pattern, but is more efficient. This is because the algorithm will not attempt to make a substitution if a substitution was not made on the previous iteration. For example, if a sentence does not contain “Instrument” in one of the Atoms in that Sentence, no substitutions will even be attempted in this Sentence. This prevents unnecessary work being done in the most computationally heavy part of the algorithm due to the fact there is no redundant rule checking.