**Same:** Node based structure and implementaiton of nodes exec methods will be the same.

**In iOS Project:**

**Structure:** Stored in a linked list

**Compilation Phase:** Parse command text and then instantiate corresponding Syntax Node. Then stored the Syntax node in a linked list, adding to tail every time.

**Execution Phase:** Iterate through linked list, running exec command on all and then non block nodes returned false and block nodes returned true, and if returned true next execution would be in body of block node not from root node. Indexer was just an integer.

**Variables:** Stored in an array, and were really just iterators, formatted like this iterator\_n where n is number u gave it.

**In Unity Project:**

**Structure:** Start from root and just traverse children, the method to traverse will be in class called SyntaxTree. All variables will also be stored in this class in a dictionary.

**Compilation Phase:** Parse command text, instantiate Corresponding Syntax Node. Assign the new Syntax Node it's parent which would be the current node we're assigning children to, this changes everytime we run into a blocknode.

**Execution Phase:** I now have a class called Executor, whose job is to point to function node is doing. I will be using Executor to traverse, it will be assigned returned value of exec by every SyntaxNode it goes through. *eg:* ***first****: executorPtr = root.exec();* ***repeat****: executorPtr = executorPtr();* Checkig if command type node or variable declaration, basically execute or declarative, just like normal compilers.

**Variables:** Will have normal variables along with iterators, and they will be stored within SyntaxTree object within a dictionary of dictionaries. Environment name key to variable names within that Environment.