LL(1) Parser

# Components:

* Grammar
* ParsingTable

# Grammar:

For the most part it remained the same as it was.

The only addition to this class is the productionIndex map, which holds the index of each production. This was needed because the Grammar class holds the productions in a hash set which changes their order.

# ParsingTable:

## Components:

* Grammar
* FirstAndFollowCalculator – holds first and follow functions
* Symbols – a set of every symbol in the grammar(union of terminals and non-terminals and the symbol “$”)
* parsingTable – map which will keep for each symbol another map containing values of the parsingTable from that symbol to the terminals.

A value of the parsingTable is an object of class ParsingTableCell. It holds the right hand side of the production and its index.

# FirstAndFollowCalculator:

Both functions will be represented using a map with pairs of string and set of strings. The first is the symbol and the second is the result of the function.

When instantiated, an object from this class will compute the first and follow functions which are reachable through getters.

## Follow Function:

* We stop the algorithm when, after an iteration, nothing changes.
* To model this into code, we created the class Iteration which holds the follow function (of that iteration).
* The algorithm will hold a list of these iterations and will stop when the current iteration is equal to the previous one.

Here we also implemented the concatenation of length one function. Since it will be used in computing the parsing table too, we made is static.