Lab 6 DOCUMENTATION

https://github.com/lola23/LFTC/tree/master/Lab_08/src

LL 1:

Fields:

grammar: Grammar

FIRST: map of non terminal with the non terminals that it has as first

FOLLOW: map of every non terminal and next symbol generated after it

numbererdProductions: map of all productions with an index

alpha: stack for terminals

beta: stack for nonterminals

pi: stack of indexes of productions

parseTable()

PRE: -

POST: constructs the parsing table

parse():

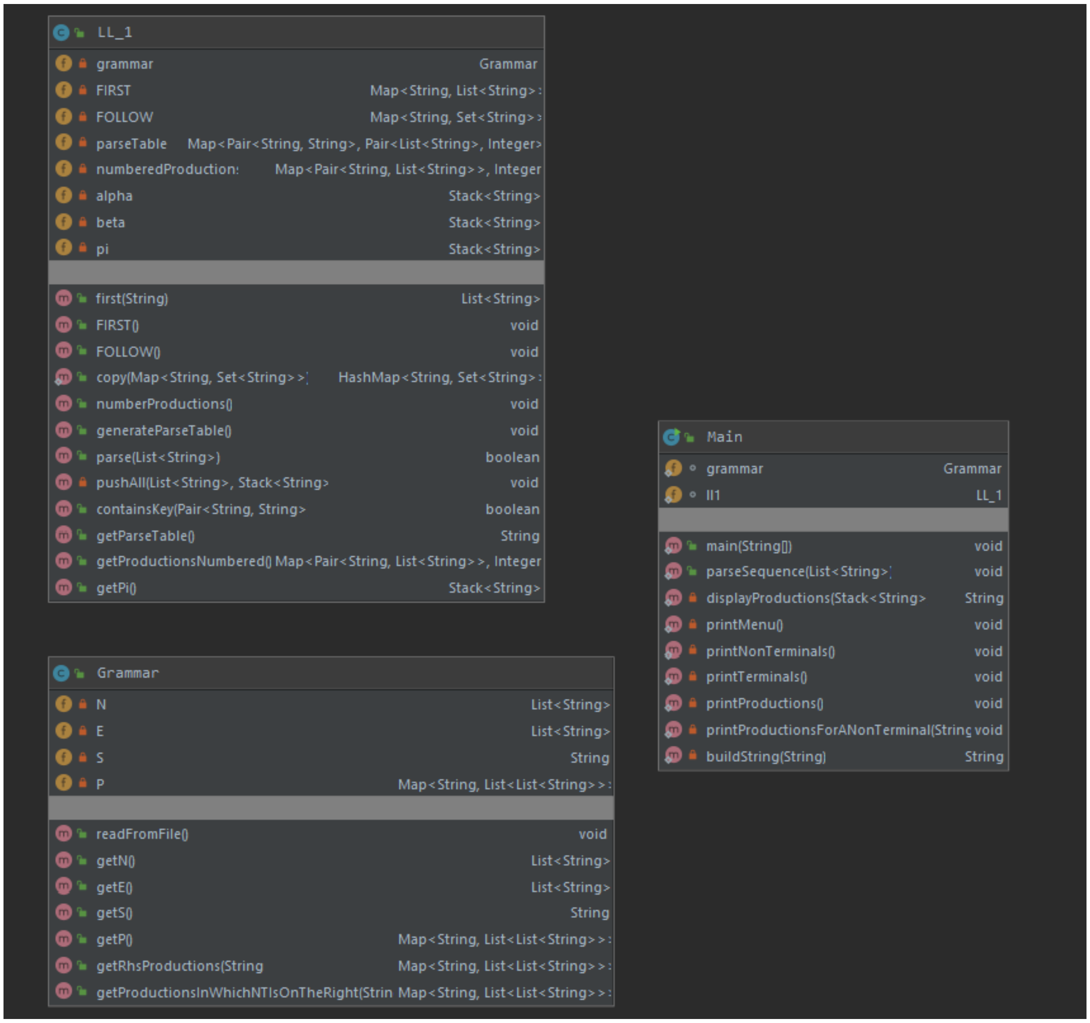
In: sequence to be checked

Out: true or false if sequence is accepted or not

numberProductions():

pre: -

post: numberedProductions is populated with every production and its index



LL_1

grammar	Grammar
FIRST	Map<String, List<String>:
FOLLOW	Map<String, Set<String>>
parseTable	Map<Pair<String, String>, Pair<List<String>, Integer>
numberedProduction:	Map<Pair<String, List<String>>, Integer
alpha	Stack<String>
beta	Stack<String>
pi	Stack<String>
first(String)	List<String>
FIRST()	void
FOLLOW()	void
copy(Map<String, Set<String>>	HashMap<String, Set<String>:
numberProductions()	void
generateParseTable()	void
parse(List<String>)	boolean
pushAll(List<String>, Stack<String>	void
containsKey(Pair<String, String>	boolean
getParseTable()	String
getProductionsNumbered()	Map<Pair<String, List<String>>, Integer
getPi()	Stack<String>

Grammar

N	List<String>
E	List<String>
S	String
P	Map<String, List<List<String>>:
readFromFile()	void
getN()	List<String>
getE()	List<String>
getS()	String
getP()	Map<String, List<List<String>>:
getRhsProductions(String	Map<String, List<List<String>>:
getProductionsInWhichNTIsOnTheRight(Strin	Map<String, List<List<String>>:

Main

grammar	Grammar
ll1	LL_1
main(String[])	void
parseSequence(List<String>)	void
displayProductions(Stack<String>	String
printMenu()	void
printNonTerminals()	void
printTerminals()	void
printProductions()	void
printProductionsForANonTerminal(Strinç	void
buildString(String)	String