Alan Cortez
Raxel Ortiz

CPSC 323 Project 2

For our group project consisting of Alan Cortez and Raxel Ortiz we decided to use python to build our parser. To get started be sure to have python installed and pip install the tabulate library because we use it to visualize our output. Besides that to run our code you just need to execute the main.py file and view the output in output.txt.

The following logic of how our parser works can also be seen in logic.py as python psuedocode. To begin we need to initialize to constant variables, one called parsing_table() which is a 2d vector of the provided table and the other is reduction_rules() which is used to determine which reduction rule we need to use. In our implementation we created multiple functions to separate the logic of our code such as formatting functions for the input and for our stack so that its more legible to read in the output. Additionally, we have a function called check_parsing_table() which acts as a way to see if the input character exists in the parsing table. Finally, we have a begin_parse() function which executes the following logic to determine if we need to shift, reduce, accept, or reject the input grammar. Inside of this function we maintain two stacks, one for the input grammar and one for the current call stack which is a list of numbers and characters. After that we just keep on looping for every character in the input stack and reference the parsing table to determine our next move as previously mentioned. During this process we append the current step, call stack, input stack, and action into a data variable which we use for our output. After that we just go to main(), loop through each input example provided, return a list of outputs so that we can tabulate it and write that into output.txt