Hacettepe University Computer Science and Engineering Department

Name and Surname : Mustafa ÇOBAN

Identity Number : 21627155

Course : BBM-203

Experiment : Assignment-3

Subject : Stack

Data Due : 17.12.2021

Advisors : Ahmet ALKILIÇ

e-mail : b21627155@cs.hacettepe.edu.tr

Main Program : Main.cpp

2. Software Using Documentation

2.1. Software Usage

In this assignment, we were given 2 input files and after reading these files, we were asked to store the elements in the input in the stack structure in accordance with the rules, and empty the stack structure by following the rules.

if input files ending with end states in accordance with the rules are accepted, otherwise rejected will be printed.

2.2 Error Messages

Error [1] : DPDA description is invalid!

* In the Transition rule, if there is an input that is not in the input file or a state that is not in the state set, this error is printed to notify the user.

* Changing incorrect entries in dpda file.

3.Software Design Notes

3.1. Desctiption of the program

3.1.1. Problem

Reading 2 input files given to us and storing them in the stack structure according to the desired rules and unloading them.

3.1.2.Solution

I first started by reading the 2 input files given from the command line argument. In the 1st file, the dpda file contains state set, input set, stack set, transition set, data showing which state a rule starts and ends with.

I kept all of these data in separate vectors. then I read the second input file, dpda-input . and I assigned the data in it to vector. I placed the data in the stack using these inputs.

Finally, I stored this data in the stack according to the transition rules until the data in the input file was finished, then I continued to delete it until the stack was empty.

3.2. System Chart

INPUT	PROGRAMS	OUTPUT
-command line argumand +dpda file +dpda-input file	-Main.cpp -ReadFile.h +void readDPDA() +void readDPDAinput() -Dpda.h +printStack() +ss() +rr()	-txt file +output file

3.3. Main Data Structures

+Main.cpp

-string dpdaFile: recived input file from command line argumand

-string dpdaInputFile: recived input file from command line argumand

-string output: recived output file from command line argumand

+ReadFile.h

-vector<string> Q: set of states

- vector<string> startEndPoints: start and finish state of DPDA

- vector<string> A: list of input alphabet

- vector<string> Z: list of stack alphabet

-vector<vector<string>> T: set of transition rule

-vector<vector<string>> dpdaInput: input data from dpdaInputFile

-readDPDA(): function to reading dpdaFile

-readDPDAinput(): function to reading dpdaInputFile

+Dpda.h

-string tut: next state in singletransition rule

-printStack(): print data in stack

-ss(): continue DPDA with tut data from rr()

-rr(): start DPDA with start point in startEndPoints

3.4. Algorithm

1. Make initialisation.
   1.1. Move zero to Student Count, spaces to StuSuccesfu
   1.2. Open input: 'STU.DAT', Output: 'STU.REPORT'

2. For every student in ‘STU.DAT’
   2.1. Read student data.
   2.2. Calculate mark average of this student
   2.3. Print student with her/his average to ‘STU.REPORT’
   2.4. For the processed students
      2.4.1. Find the maximum average among these.
      2.4.2. Store the name of this student to StuSuccesful.

3. Print StuSuccesful to ‘STU.REPORT’.
4. Close files