Exercise 2

Due: 05/05/2017

1 Introduction

This exercise implements an automatic checker for Linear Algebra assignments. Currently, first year students are given a linear system $Ax = b$, they apply the Gauss Jordan elimination algorithm, and then find the solution set. The purpose of our exercise is to make sure those first year students are doing their job right. For that, we will implement a tool called Lini.

2 Lini :: Details

Input Format :: Linear System

The Matrix A and the vector b are accommodated in a single $m \times (n+1)$ matrix, whose rightmost column is b. A matrix entry is either an integer, or a fraction. The linear system resides in a text file, starting with a left bracket, and ending with a right bracket. Entries are separated with either blanks or tabs, and rows are separated with a single semicolon. Fractions are written with either slash or backslash. The entire system must be written as a single line in the text file:

$$[3\ 4\ -6; 7\ -2/9\ -100; 7\ 0\ 0; -6\ 4/3\ 10]$$

Input Format :: Row Operations

The row operations sequence reside in a text file. Each line contains a single row operation. Table 1 describes the context free grammar of row operations. In addition, the following limitations are imposed on row operations: (1) Rows can not be multiplied by zero. (2) Fractions are always with a non zero denominator. (3) The indices in rule mulRow have to be the same. (4) The first two indices in mulAndAddRow have to be the same.

Input Format :: Solution Set

The solution set resides in a text file. Whenever the system has no solutions, this text file contains a single word: EMPTYSET. When the system has a single

S	::=	rowOperationList
rowOperationList	::=	rowOperation rowOperationList
	::=	rowOperation
rowOperation	::=	swapRows
	::=	mulRow
	::=	mulAndAddRow
swapRows	::=	R index leftRightArrow R index
mulRow	::=	R index leftArrow [−] num R index
mulAndAddRow	::=	R index leftArrow R index op [num] R index
index	::=	1 | 2 | 3 | 4
leftArrow	::=	$<-$
leftRightArrow	::=	$<->$
num	::=	int
	::=	int div int
int	::=	0
	::=	$[1-9][0-9]^*$
div	::=	/
	::=	\
op	::=	+
	::=	−

Table 1: Context free grammar for the language of row operations.

solution, it is written as a set with a single row vector $\{(3, -\frac{2}{9}, 200)\}$. If the system has an infinite solution set, it should be written as a sum of two vector sets: $\{(2, -6, 100)\}$ + SP($\{(1, 0, 1), (6, 6, 0)\}$).

3 Programming Assignment

Write two top down (recursive descent) parsers for the row operations and solution set. You are free to change the grammar of the row operations as you like, but the language itself must not change. In particular, you can apply any of the transformations discussed in class: (1) eliminating left recursion, (2) left factoring, and (3) substitution. The exact context free grammar for the solution set is missing intentionally. You can start by writing a grammar for it, and then fix it similarly to the way you fixed the original row operations grammar. The solution set should support row vectors with sizes $= 2, 3, 4$, and naturally, all row vectors must be the same size. For example, the solution set

$$\{(5, 6/7, -3)\} + SP(\{(4, -800, 9), (3, 4, 5, 6)\})$$

is invalid, as it contains row vectors of different sizes.

4 Setting Up the Working Environment

Your working environment should be already up from exercise 1.

5 Submission Guidelines

Simply create a sub folder EX2 that will contain a makefile building your source to a runnable program called Lini. Note, that currently Lini resides inside `FOLDER_06_Lini` so please have your makefile copy it from there before submitting. To avoid the pollution of EX2, please remove all *.o files once the target is built. The next paragraphs describe the expected output format and invocation of Lini.

Expected output of Lini If the input row operations file has correct syntax, the corresponding row operations status file should contain a single word: OK. If the input row operations file has *incorrect* syntax, the corresponding row operations status file should contain the word ERROR(*location*) with *location* being the line number of the *first* error encountered. The exact same instructions apply for the solution set too, but since the solution set is written as one line, *location* should be the offset of the first error inside that line.

Execution parameters Lini recevies 4 input file names:

RowOperations.txt
SolutionSet.txt
RowOperationsStatus.txt
SolutionSetStatus.txt