CD Lab Assignment 6

Name : Nalanaga Yasaswin

Roll no : 20CS01040

Q2) for the question the output for the sample input grammar

1. S -> E$
2. E -> E + T | E – T | T
3. T -> (E) | i

I have printed all the states in the dfa with each state having it's list of productions, state num, it's adjacency list and reductions if any

Basically, my state in the dfa, I declared it as a class which have the above attributes along with accept attribute, which is 1 only for the state that has $. At the end of the production

At the end I have used this dfa to make a parse table which is basically a map of state, terminal/non-terminal pair as a key field and value is an integer vector and printed the table with keeping understandability of the user in mind

```
D:\Compiler_Design_Assignment_1>cd
"d:\Compiler_Design_Assignment_1\20CS01040_Nalanaga_Yasaswin_A6\" && g++
20CS01040_q2.cpp -o 20CS01040_q2 &&
"d:\Compiler_Design_Assignment_1\20CS01040_Nalanaga_Yasaswin_A6\"20CS01040_q2

NOTE : all the terminals and non terminals should be single characters

enter the no.of terminals

5

enter the list of all terminals

+ - ( ) i

enter the no.of non-terminals

3

enter the list of all non terminals
```

S E T

Enter the no.of productions

3

Enter the production rules

S -> E$

E -> E + T | E - T | T

T -> (E) | i

Grammar

E -> E+T|E-T|T|

S -> E$|

T -> (E)|i|

The rules are as follows

1)E->E+T

2)E->E-T

3)E->T

4)S->E$

5)T->(E)

6)T->i

The automaton for the LR(0) parser is as shown below

For state : 0

(: 5

E : 1

T : 11

i : 10

List of all the productions in the node 0 is :

S->.E$

E->.E+T

E->.E-T

E->.T

T->.(E)

T->.i

Reduce actions are :

For state : 1

$: 2

+ : 3

- : 8

List of all the productions in the node 1 is :

S->E.$

E->E.+T

E->E.-T

Reduce actions are :

For state : 2

List of all the productions in the node 2 is :

S->E$.

Reduce actions are : 4

For state : 3

(: 5

T : 4

i : 10

List of all the productions in the node 3 is :

E->E+.T

T->.(E)

T->.i

Reduce actions are :

For state : 4

List of all the productions in the node 4 is :

E->E+T.

Reduce actions are : 1

For state : 5

(: 5

E : 6

T : 11

i : 10

List of all the productions in the node 5 is :

T->(.E)

E->.E+T

E->.E-T

E->.T

T->.(E)

T->.i

Reduce actions are :

For state : 6

) : 7

+ : 3

- : 8

List of all the productions in the node 6 is :

T->(E.)

E->E.+T

E->E.-T

Reduce actions are :

For state : 7

List of all the productions in the node 7 is :

T->(E).

Reduce actions are : 5

For state : 8

(: 5

T : 9

i : 10

List of all the productions in the node 8 is :

E->E-.T

T->.(E)

T->.i

Reduce actions are :

For state : 9

List of all the productions in the node 9 is :

E->E-T.

Reduce actions are : 2

For state : 10

List of all the productions in the node 10 is :

T->i.

Reduce actions are : 6

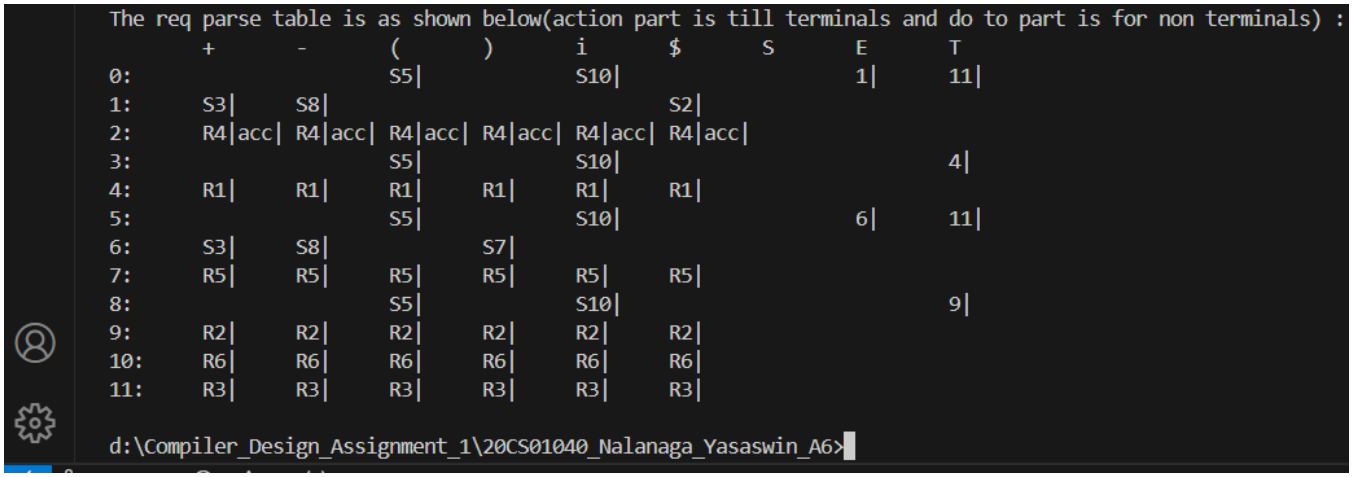
For state : 11

List of all the productions in the node 11 is :

E->T.

Reduce actions are : 3

The req parse table is as shown below(action part is till terminals and do to part is for non terminals) :



d:\Compiler_Design_Assignment_1\20CS01040_Nalanaga_Yasaswin_A6>