

**Compiler Design Lab**

**SUBMITTED BY: SUBMITTED TO:**

Name: Harsh Agrawal Dr. Ajay Prasad

SAP ID: 500107136

Enrollment No: R2142220326

Course- B.Tech. (CSE)

Batch: 4 (AI/ML) (Non- Hons)

**Assignment 6**

1. WAP to construct LL(1) parser for an expression in C

Code-

#include<stdio.h>

#include<string.h>

#define TSIZE 128

int table[100][TSIZE];

char terminal[TSIZE];

char non\_terminal[26];

struct product {

char str[100];

int len;

}pro[20];

int no\_pro;

char first[26][TSIZE];

char follow[26][TSIZE];

char first\_rhs[100][TSIZE];

int isNT(char c) {

return c >= 'A' && c <= 'Z';

}

void readFromFile() {

FILE\* fptr;

fptr = fopen("text.txt", "r");

char buffer[255];

int i;

int j;

while (fgets(buffer, sizeof(buffer), fptr)) {

printf("%s", buffer);

j = 0;

non\_terminal[buffer[0] - 'A'] = 1;

for (i = 0; i<strlen(buffer) - 1; ++i) {

if (buffer[i] == '|') {

++no\_pro;

pro[no\_pro - 1].str[j] = '\0';

pro[no\_pro - 1].len = j;

pro[no\_pro].str[0] = pro[no\_pro - 1].str[0];

pro[no\_pro].str[1] = pro[no\_pro - 1].str[1];

pro[no\_pro].str[2] = pro[no\_pro - 1].str[2];

j = 3;

}

else {

pro[no\_pro].str[j] = buffer[i];

++j;

if (!isNT(buffer[i]) && buffer[i] != '-' && buffer[i] != '>') {

terminal[buffer[i]] = 1;

}

}

}

pro[no\_pro].len = j;

++no\_pro;

}

}

void add\_FIRST\_A\_to\_FOLLOW\_B(char A, char B) {

int i;

for (i = 0; i< TSIZE; ++i) {

if (i != '^')

follow[B - 'A'][i] = follow[B - 'A'][i] || first[A - 'A'][i];

}

}

void add\_FOLLOW\_A\_to\_FOLLOW\_B(char A, char B) {

int i;

for (i = 0; i< TSIZE; ++i) {

if (i != '^')

follow[B - 'A'][i] = follow[B - 'A'][i] || follow[A - 'A'][i];

}

}

void FOLLOW() {

int t = 0;

int i, j, k, x;

while (t++ <no\_pro) {

for (k = 0; k < 26; ++k) {

if (!non\_terminal[k])

continue;

char nt = k + 'A';

for (i = 0; i<no\_pro; ++i) {

for (j = 3; j < pro[i].len; ++j) {

if (nt == pro[i].str[j]) {

for (x = j + 1; x < pro[i].len; ++x) {

char sc = pro[i].str[x];

if (isNT(sc)) {

add\_FIRST\_A\_to\_FOLLOW\_B(sc, nt);

if (first[sc - 'A']['^'])

continue;

}

else {

follow[nt - 'A'][sc] = 1;

}

break;

}

if (x == pro[i].len)

add\_FOLLOW\_A\_to\_FOLLOW\_B(pro[i].str[0], nt);

}

}

}

}

}

}

void add\_FIRST\_A\_to\_FIRST\_B(char A, char B) {

int i;

for (i = 0; i< TSIZE; ++i) {

if (i != '^') {

first[B - 'A'][i] = first[A - 'A'][i] || first[B - 'A'][i];

}

}

}

void FIRST() {

int i, j;

int t = 0;

while (t <no\_pro) {

for (i = 0; i<no\_pro; ++i) {

for (j = 3; j < pro[i].len; ++j) {

char sc = pro[i].str[j];

if (isNT(sc)) {

add\_FIRST\_A\_to\_FIRST\_B(sc, pro[i].str[0]);

if (first[sc - 'A']['^'])

continue;

}

else {

first[pro[i].str[0] - 'A'][sc] = 1;

}

break;

}

if (j == pro[i].len)

first[pro[i].str[0] - 'A']['^'] = 1;

}

++t;

}

}

void add\_FIRST\_A\_to\_FIRST\_RHS\_\_B(char A, int B) {

int i;

for (i = 0; i< TSIZE; ++i) {

if (i != '^')

first\_rhs[B][i] = first[A - 'A'][i] || first\_rhs[B][i];

}

}

void FIRST\_RHS() {

int i, j;

int t = 0;

while (t <no\_pro) {

for (i = 0; i<no\_pro; ++i) {

for (j = 3; j < pro[i].len; ++j) {

char sc = pro[i].str[j];

if (isNT(sc)) {

add\_FIRST\_A\_to\_FIRST\_RHS\_\_B(sc, i);

if (first[sc - 'A']['^'])

continue;

}

else {

first\_rhs[i][sc] = 1;

}

break;

}

if (j == pro[i].len)

first\_rhs[i]['^'] = 1;

}

++t;

}

}

int main() {

readFromFile();

follow[pro[0].str[0] - 'A']['$'] = 1;

FIRST();

FOLLOW();

FIRST\_RHS();

int i, j, k;

printf("\n");

for (i = 0; i<no\_pro; ++i) {

if (i == 0 || (pro[i - 1].str[0] != pro[i].str[0])) {

char c = pro[i].str[0];

printf("FIRST OF %c: ", c);

for (j = 0; j < TSIZE; ++j) {

if (first[c - 'A'][j]) {

printf("%c ", j);

}

}

printf("\n");

}

}

printf("\n");

for (i = 0; i<no\_pro; ++i) {

if (i == 0 || (pro[i - 1].str[0] != pro[i].str[0])) {

char c = pro[i].str[0];

printf("FOLLOW OF %c: ", c);

for (j = 0; j < TSIZE; ++j) {

if (follow[c - 'A'][j]) {

printf("%c ", j);

}

}

printf("\n");

}

}

printf("\n");

for (i = 0; i<no\_pro; ++i) {

printf("FIRST OF %s: ", pro[i].str);

for (j = 0; j < TSIZE; ++j) {

if (first\_rhs[i][j]) {

printf("%c ", j);

}

}

printf("\n");

}

terminal['$'] = 1;

terminal['^'] = 0;

printf("\n");

printf("\n\t\*\*\*\*\*\* LL(1) PARSING TABLE \*\*\*\*\*\*\*\n");

printf("\t--------------------------------------------------------\n");

printf("%-10s", "");

for (i = 0; i< TSIZE; ++i) {

if (terminal[i]) printf("%-10c", i);

}

printf("\n");

int p = 0;

for (i = 0; i<no\_pro; ++i) {

if (i != 0 && (pro[i].str[0] != pro[i - 1].str[0]))

p = p + 1;

for (j = 0; j < TSIZE; ++j) {

if (first\_rhs[i][j] &&j != '^') {

table[p][j] = i + 1;

}

else if (first\_rhs[i]['^']) {

for (k = 0; k < TSIZE; ++k) {

if (follow[pro[i].str[0] - 'A'][k]) {

table[p][k] = i + 1;

}

}

}

}

}

k = 0;

for (i = 0; i<no\_pro; ++i) {

if (i == 0 || (pro[i - 1].str[0] != pro[i].str[0])) {

printf("%-10c", pro[i].str[0]);

for (j = 0; j < TSIZE; ++j) {

if (table[k][j]) {

printf("%-10s", pro[table[k][j] - 1].str);

}

else if (terminal[j]) {

printf("%-10s", "");

}

}

++k;

printf("\n");

}

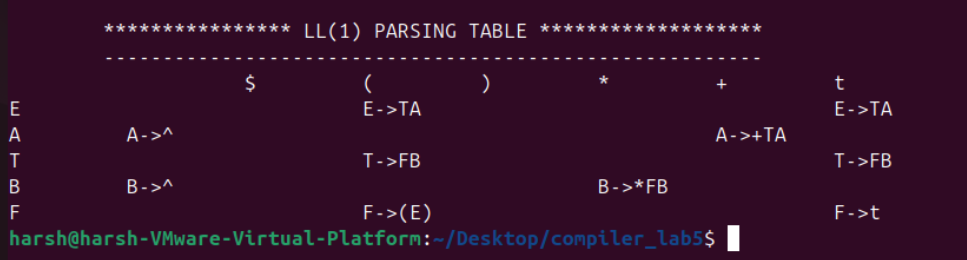
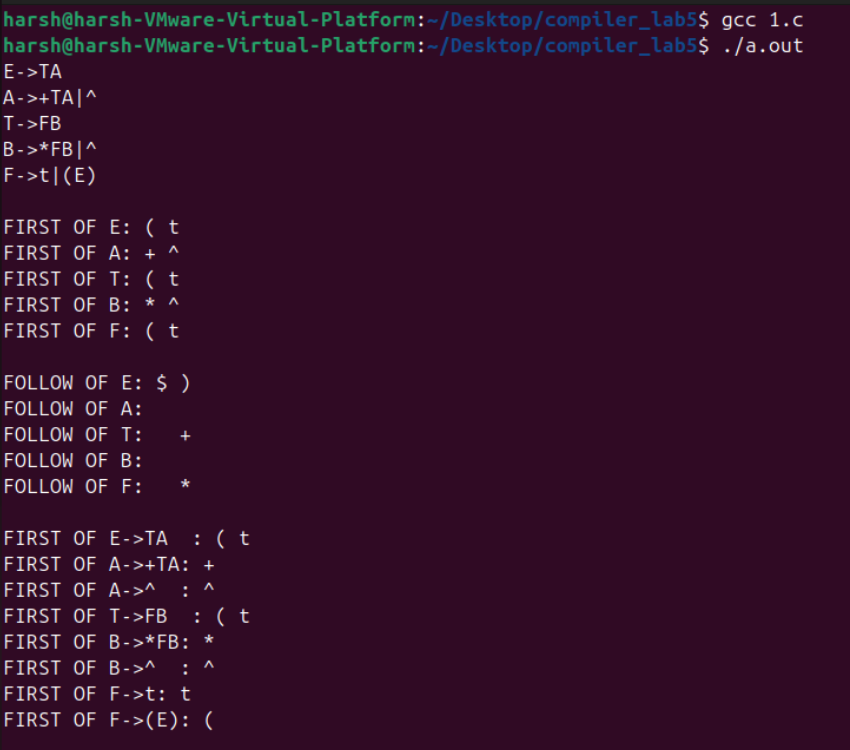
}

}

Text.txt-Top of Form



Output-



Bottom of Form