PREDICTIVE PARSING TABLE

CODE:

#include <stdio.h>

#include <string.h>

#define NUM\_NON\_TERMINALS 5

#define NUM\_TERMINALS 5

enum SymbolType {

TERMINAL,

NON\_TERMINAL,

EPSILON

};

char nonTerminals[NUM\_NON\_TERMINALS] = {'S', 'E', 'E\', 'T', 'T'};

char terminals[NUM\_TERMINALS] = {'$', '+', '\*', '(', ')', 'id'};

enum SymbolType symbolType(char symbol) {

if (symbol == 'e') {

return EPSILON;

}

for (int i = 0; i < NUM\_NON\_TERMINALS; i++) {

if (symbol == nonTerminals[i]) {

return NON\_TERMINAL;

}

}

for (int i = 0; i < NUM\_TERMINALS; i++) {

if (symbol == terminals[i]) {

return TERMINAL;

}

}

return -1; // Invalid symbol

}

int main() {

char parsingTable[NUM\_NON\_TERMINALS][NUM\_TERMINALS][10];

// Initialize parsing table

for (int i = 0; i < NUM\_NON\_TERMINALS; i++) {

for (int j = 0; j < NUM\_TERMINALS; j++) {

strcpy(parsingTable[i][j], ""); // Initialize with empty production

}

}

// Fill in the parsing table

strcpy(parsingTable[0][0], "E$");

strcpy(parsingTable[1][3], "TE'");

strcpy(parsingTable[2][1], "+TE'");

strcpy(parsingTable[2][4], "e");

strcpy(parsingTable[3][3], "FT'");

strcpy(parsingTable[4][1], "e");

strcpy(parsingTable[4][2], "\*FT'");

// Display parsing table

printf("Predictive Parsing Table:\n");

printf(" ");

for (int j = 0; j < NUM\_TERMINALS; j++) {

printf("%-10c", terminals[j]);

}

printf("\n");

for (int i = 0; i < NUM\_NON\_TERMINALS; i++) {

printf("%c ", nonTerminals[i]);

for (int j = 0; j < NUM\_TERMINALS; j++) {

printf("%-10s", parsingTable[i][j]);

}

printf("\n");

}

return 0;

}