7. Write a C program to find FIRST( ) - predictive parser for the given grammar

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <ctype.h>

#define MAX 10

void findFirst(char, int, int);

void addToResult(char);

int numProductions;

char productions[MAX][MAX];

char first[MAX];

int main() {

int i;

int choice;

char c;

char result[MAX];

// Example Grammar: S -> AB, A -> a | ε, B -> b

printf("Enter number of productions: ");

scanf("%d", &numProductions);

printf("Enter the productions (e.g., S=AB): \n");

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

scanf("%s", productions[i]);

}

do {

printf("Find FIRST of: ");

scanf(" %c", &c);

findFirst(c, 0, 0);

printf("FIRST(%c) = { ", c);

for (i = 0; first[i] != '\0'; i++) {

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

}

printf("}\n");

printf("Do you want to continue (1/0)? ");

scanf("%d", &choice);

} while (choice == 1);

return 0;

}

void findFirst(char c, int q1, int q2) {

int j;

if (!(isupper(c))) {

addToResult(c);

return;

}

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

if (productions[j][0] == c) {

if (productions[j][2] == '$') {

addToResult('$');

} else if (islower(productions[j][2])) {

addToResult(productions[j][2]);

} else {

findFirst(productions[j][2], j, 3);

}

}

}

return;

}

void addToResult(char c) {

int i;

for (i = 0; first[i] != '\0'; i++)

if (first[i] == c)

return;

first[i] = c;

first[i + 1] = '\0';

}

