**Write a C program to find FOLLOW( ) - predictive parser for the given grammar  
S → AaAb / BbBa  
A → ∈  
B → ∈**

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

#include <string.h>

#define MAX 20

void followFirst(char c, int c1, int c2);

void follow(char c);

void addToFollow(char c);

int count;

char production[MAX][MAX], followSet[MAX];

void followFirst(char c, int c1, int c2) {

if (!(isupper(c))) { // If the character is a terminal symbol

addToFollow(c);

} else {

int j = 0;

while (j < count) {

if (production[j][0] == c) { // If it matches the non-terminal

if (production[j][2] == '#') { // Case for epsilon production

if (production[c1][c2] == '\0') {

follow(production[c1][0]);

} else if (production[c1][c2] != '\0' && c != production[c1][c2]) {

followFirst(production[c1][c2], c1, c2 + 1);

}

} else if (!isupper(production[j][2])) { // If it's a terminal

addToFollow(production[j][2]);

} else {

followFirst(production[j][2], j, 3); // Recurse for non-terminal

}

}

j++;

}

}

}

void follow(char c) {

int i, j;

if (production[0][0] == c) {

addToFollow('$'); // End of input marker

}

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

for (j = 2; j < strlen(production[i]); j++) {

if (production[i][j] == c) {

if (production[i][j + 1] != '\0') { // If there’s a symbol after c

followFirst(production[i][j + 1], i, j + 2);

}

if (production[i][j + 1] == '\0' && c != production[i][0]) {

follow(production[i][0]); // If it's the last symbol, follow the non-terminal on the left

}

}

}

}

}

void addToFollow(char c) {

if (strchr(followSet, c) == NULL) { // Only add if it's not already in the set

strncat(followSet, &c, 1);

}

}

int main() {

int i;

count = 6; // Number of production rules

// Given productions

strcpy(production[0], "S=AB");

strcpy(production[1], "S=BB");

strcpy(production[2], "A=aA");

strcpy(production[3], "A=#");

strcpy(production[4], "B=bB");

strcpy(production[5], "B=#");

printf("FOLLOW of each non-terminal\n");

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

strcpy(followSet, ""); // Reset followSet for each non-terminal

follow(production[i][0]);

printf("FOLLOW(%c) = { ", production[i][0]);

for (int j = 0; j < strlen(followSet); j++) {

printf("%c ", followSet[j]);

}

printf("}\n");

}

return 0;

}

A screenshot of a computer program

AI-generated content may be incorrect.