14. Implement the concept of Shift reduce parsing in C Programming.

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define STACK\_SIZE 100

#define INPUT\_SIZE 100

char stack[STACK\_SIZE];

int top = -1;

char input[INPUT\_SIZE];

int inputIndex = 0;

void push(char c) {

if (top < STACK\_SIZE - 1) {

stack[++top] = c;

} else {

printf("Stack overflow\n");

exit(1);

}

}

char pop() {

if (top >= 0) {

return stack[top--];

} else {

printf("Stack underflow\n");

exit(1);

}

}

void shift() {

push(input[inputIndex++]);

}

void reduce() {

if (top >= 2 && stack[top] == 'E' && stack[top-1] == '+' && stack[top-2] == 'E') {

pop();

pop();

pop();

push('E');

} else if (top >= 2 && stack[top] == 'E' && stack[top-1] == '\*' && stack[top-2] == 'E') {

pop();

pop();

pop();

push('E');

} else if (top >= 2 && stack[top] == ')' && stack[top-1] == 'E' && stack[top-2] == '(') {

pop();

pop();

pop();

push('E');

} else if (top >= 0 && (stack[top] >= 'a' && stack[top] <= 'z')) {

pop();

push('E');

} else {

printf("Reduction error\n");

exit(1);

}

}

void printStack() {

for (int i = 0; i <= top; i++) {

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

}

printf("\n");

}

int main() {

strcpy(input, "a+a\*a"); // You can change this to test different inputs

while (input[inputIndex] != '\0') {

shift();

printStack();

while (top >= 0 && (stack[top] == 'E' || stack[top] == '+' || stack[top] == '\*' || stack[top] == '(' || stack[top] == ')')) {

if (stack[top] == '+' || stack[top] == '\*' || stack[top] == '(' || stack[top] == ')') {

break;

}

reduce();

printStack();

}

}

if (top == 0 && stack[top] == 'E') {

printf("Parsing successful\n");

} else {

printf("Parsing failed\n");

}

return 0;

}

