15. Write a C Program to implement the operator precedence parsing.

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define SIZE 100

char stack[SIZE];

int top = -1;

char precedence\_table[5][5] = {

/\* Operators: + - \* / $

\* ----------------

\* + | > > < < >

\* - | > > < < >

\* \* | > > > > >

\* / | > > > > >

\* $ | < < < < =

\*/

{'>', '>', '<', '<', '>'},

{'>', '>', '<', '<', '>'},

{'>', '>', '>', '>', '>'},

{'>', '>', '>', '>', '>'},

{'<', '<', '<', '<', '='}

};

int get\_operator\_index(char c) {

switch (c) {

case '+': return 0;

case '-': return 1;

case '\*': return 2;

case '/': return 3;

case '$': return 4;

default: return -1;

}

}

void push(char c) {

if (top >= SIZE - 1) {

printf("Stack Overflow\n");

exit(1);

}

stack[++top] = c;

}

char pop() {

if (top == -1) {

printf("Stack Underflow\n");

exit(1);

}

return stack[top--];

}

char peek() {

if (top == -1)

return '\0';

return stack[top];

}

void operator\_precedence\_parser(char \*expr) {

push('$'); // Start with end marker

int i = 0;

char a, b;

char symbol;

printf("Parsing Steps:\n");

while (expr[i] != '\0') {

symbol = expr[i];

if (symbol >= '0' && symbol <= '9') { // Operand, just move forward

printf("Read operand: %c\n", symbol);

i++;

continue;

}

int stack\_top\_index = get\_operator\_index(peek());

int input\_index = get\_operator\_index(symbol);

if (input\_index == -1) {

printf("Invalid symbol: %c\n", symbol);

return;

}

switch (precedence\_table[stack\_top\_index][input\_index]) {

case '<':

case '=':

push(symbol);

printf("Shift: %c\n", symbol);

i++;

break;

case '>':

b = pop();

printf("Reduce: %c\n", b);

break;

default:

printf("Error in parsing\n");

return;

}

}

while (peek() != '$') {

b = pop();

printf("Reduce: %c\n", b);

}

printf("Parsing completed successfully.\n");

}

int main() {

char expression[SIZE];

printf("Enter an arithmetic expression: ");

scanf("%s", expression);

strcat(expression, "$");

operator\_precedence\_parser(expression);

return 0;

}

