OPERATOR PRECEDENCE

CODE:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

#define STACK\_SIZE 50

char stack[STACK\_SIZE];

int top = -1;

void push(char item) {

if (top >= STACK\_SIZE - 1) {

printf("Stack Overflow!\n");

exit(1);

}

stack[++top] = item;

}

char pop() {

if (top == -1) {

printf("Stack Underflow!\n");

exit(1);

}

return stack[top--];

}

int precedence(char op) {

if (op == '+' || op == '-')

return 1;

else if (op == '\*' || op == '/')

return 2;

return 0;

}

void operatorPrecedenceParsing(char input[]) {

int i = 0;

printf("Stack\tInput\tAction\n");

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

if (input[i] == '(') {

push(input[i]);

i++;

} else if (isalnum(input[i])) {

printf("%s\t", stack);

printf("%s\t", input + i);

printf("Shift %c\n", input[i]);

push(input[i]);

i++;

} else if (input[i] == ')') {

while (top != -1 && stack[top] != '(') {

printf("%s\t", stack);

printf("%s\t", input + i);

printf("Reduce %c\n", pop());

}

if (top != -1 && stack[top] == '(') {

pop();

}

i++;

} else {

while (top != -1 && precedence(stack[top]) >= precedence(input[i])) {

printf("%s\t", stack);

printf("%s\t", input + i);

printf("Reduce %c\n", pop());

}

push(input[i]);

i++;

}

}

while (top != -1) {

printf("%s\t", stack);

printf(" \t");

printf("Reduce %c\n", pop());

}

printf("ACCEPTED\n");

}

int main() {

char input[50];

printf("Enter an arithmetic expression: ");

scanf("%s", input);

operatorPrecedenceParsing(input);

return 0;

}

OUTPUT:

Enter an arithmetic expression: A+B-C/D\*E

Stack Input Action

A+B-C/D\*E Shift A

A+ B-C/D\*E Shift B

A+B- C/D\*E Shift C

A+B-C/ D\*E Shift D

A+B-C/D\* E Shift E

A+B-C/D\*E Reduce E

A+B-C/D\*E Reduce \*

A+B-C/D\*E Reduce D

A+B-C/D\*E Reduce /

A+B-C/D\*E Reduce C

A+B-C/D\*E Reduce -

A+B-C/D\*E Reduce B

A+B-C/D\*E Reduce +

A+B-C/D\*E Reduce A

ACCEPTED

--------------------------------

Process exited after 11.72 seconds with return value 0

Press any key to continue . . .

