**DATA STRUCTURE**

**EXP NO:5**

**NAME:**MOHAMMED FAISAL SM

**REG NO:** 2116231801105

**DEPT:**AI&DS

WEEK 5

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

#define MAX\_SIZE 100

struct Stack {

int top;

unsigned capacity;

char \*array;

};

struct Stack\* createStack(unsigned capacity) {

struct Stack\* stack = (struct Stack\*) malloc(sizeof(struct Stack));

stack->capacity = capacity;

stack->top = -1;

stack->array = (char\*) malloc(stack->capacity \* sizeof(char));

return stack;

}

int isFull(struct Stack\* stack) {

return stack->top == stack->capacity - 1;

}

int isEmpty(struct Stack\* stack) {

return stack->top == -1;

}

void push(struct Stack\* stack, char item) {

if (isFull(stack))

return;

stack->array[++stack->top] = item;

}

char pop(struct Stack\* stack) {

if (isEmpty(stack))

return '\0';

return stack->array[stack->top--];

}

int precedence(char op) {

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

return 1;

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

return 2;

else

return -1;

}

void infixToPostfix(char\* infix, char\* postfix) {

struct Stack\* stack = createStack(strlen(infix));

int i, j;

for (i = 0, j = -1; infix[i]; ++i) {

if (isalnum(infix[i]))

postfix[++j] = infix[i];

else if (infix[i] == '(')

push(stack, '(');

else if (infix[i] == ')') {

while (!isEmpty(stack) && stack->array[stack->top] != '(')

postfix[++j] = pop(stack);

if (!isEmpty(stack) && stack->array[stack->top] != '(')

return;

else

pop(stack);

} else {

while (!isEmpty(stack) && precedence(infix[i]) <= precedence(stack->array->top))

postfix[++j] = pop(stack);

push(stack, infix[i]);

}

}

while (!isEmpty(stack))

postfix[++j] = pop(stack);

postfix[++j] = '\0';

}

int main() {

char infix[MAX\_SIZE];

char postfix[MAX\_SIZE];

printf("Enter an infix expression: ");

fgets(infix, MAX\_SIZE, stdin);

infix[strcspn(infix, "\n")] = 0;

infixToPostfix(infix, postfix);

printf("Postfix expression: %s\n", postfix);

return 0;

}

Output

Enter the infixt expression:((a+b)\*(c+d)\*(e/f)\*

Postfix expression is:ab+cd+\*ef/\*g^