**Infix to Postfix with Stack using Array**

#include <stdio.h>

else {

while (!isEmpty(&s) && precedence(topItem(&s)) >= precedence(infix[i])) {

postfix[j++] = pop(&s);

}

push(&s, infix[i]);

}

i++;

}

while (!isEmpty(&s)) {

postfix[j++] = pop(&s);

}

postfix[j] = '\0';

}

int main(){

char infix[MAX], postfix[MAX];

printf("Enter an infix expression: ");

fgets(infix, MAX, stdin);

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

infixToPostfix(infix, postfix);

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

return 0;

}

#include <stdlib.h>

#include <ctype.h>

#define MAX 50

typedef struct {

int top;

char items[MAX];

} Stack;

void initStack( Stack\* s){

s -> top = -1;

}

int isEmpty(Stack\* s){

return s->top == -1;

}

void push(Stack\* s, char item){

if (s->top < MAX -1) s->items[++s->top] = item;

else {printf("Stack overflow\n"); }

}

char pop(Stack\* s){

if(!isEmpty(s)) return s->items[s->top--];

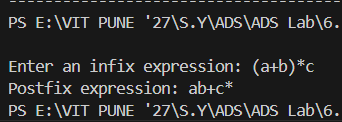
else { printf("Stack underflow\n"); return '\0'; } }

char topItem(Stack\* s){

if(!isEmpty(s)) return s-> items [s->top];

return '\0'; }

int precedence(char op){



if (op == '^' || op == '%') return 3;

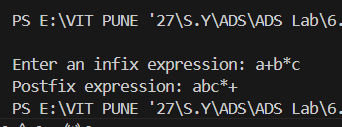
else if (op == '\*' || op == '/') return 2;

else if (op == '+' || op == '-') return 1;

else return 0;

}

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



Stack s; initStack(&s); int i = 0, j = 0;

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

if (isalnum(infix[i])){ postfix [j++] = infix [i]; }

else if (infix[i] == '('){ push (&s, infix[i]); }

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

while(!isEmpty(&s) && topItem(&s)!= '('){

postfix[j++] = pop(&s); } pop(&s);

}