Stack | Set 2 (Infix to Postfix)

March 23, 2013

Infix expression: The expression of the form a op b. When an operator is in-between every pair of operands.

Postfix expression: The expression of the form a b op. When an operator is followed for every pair of operands.

Why postfix representation of the expression?
The compiler scans the expression either from left to right or from right to left.

Consider the below expression: a op1 b op2 c op3 d
If op1 = +, op2 = *, op3 = +

The compiler first scans the expression to evaluate the expression b * c, then again scan the expression to add a to it. The result is then added to d after another scan.

The repeated scanning makes it very in-efficient. It is better to convert the expression to postfix(or prefix) form before evaluation.

The corresponding expression in postfix form is: abc*d++. The postfix expressions can be evaluated easily using a stack. We will cover postfix expression evaluation in a separate post.

Algorithm
1. Scan the infix expression from left to right.
2. If the scanned character is an operand, output it.
3. Else,
…..**3.1** If the precedence of the scanned operator is greater than the precedence of the operator in the stack(or the stack is empty), push it.

…..**3.2** Else, Pop the operator from the stack until the precedence of the scanned operator is less-equal to the precedence of the operator residing on the top of the stack. Push the scanned operator to the stack.
4. If the scanned character is an '(', push it to the stack.
5. If the scanned character is an ')', pop and output from the stack until an '(' is encountered.
6. Repeat steps 2-6 until infix expression is scanned.
7. Pop and output from the stack until it is not empty.

Following is C implementation of the above algorithm

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

// Stack type
struct Stack
{
    int top;
    unsigned capacity;
    int* array;
};

// Stack Operations
struct Stack* createStack( unsigned capacity )
{
    struct Stack* stack = (struct Stack*) malloc(sizeof(struct Stack));

    if (!stack)
        return NULL;

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

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

    if (!stack->array)
        return NULL;
    return stack;
}
int isEmpty(struct Stack* stack)
{
    return stack->top == -1 ;
}
char peek(struct Stack* stack)
{
    return stack->array[stack->top];
}
char pop(struct Stack* stack)
{
    if (!isEmpty(stack))
        return stack->array[stack->top--] ;
    return '$';
}
void push(struct Stack* stack, char op)
```

```
{
    stack->array[++stack->top] = op;
}


// A utility function to check if the given character is operand
int isOperand(char ch)
{
    return (ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z');
}

// A utility function to return precedence of a given operator
// Higher returned value means higher precedence
int Prec(char ch)
{
    switch (ch)
    {
    case '+':
    case '-':
        return 1;

    case '*':
    case '/':
        return 2;

    case '^':
        return 3;
    }
    return -1;
}


// The main function that converts given infix expression
// to postfix expression.
int infixToPostfix(char* exp)
{
    int i, k;

    // Create a stack of capacity equal to expression size
    struct Stack* stack = createStack(strlen(exp));
    if(!stack) // See if stack was created successfully
        return -1 ;

    for (i = 0, k = -1; exp[i]; ++i)
    {
         // If the scanned character is an operand, add it to output.
        if (isOperand(exp[i]))
            exp[++k] = exp[i];

        // If the scanned character is an ‘(‘, push it to the stack.
        else if (exp[i] == '(')
            push(stack, exp[i]);
        //  If the scanned character is an ‘)’, pop and output from the stack
        // until an ‘(‘ is encountered.
        else if (exp[i] == ')')
        {
            while (!isEmpty(stack) && peek(stack) != '(')
                exp[++k] = pop(stack);
            if (!isEmpty(stack) && peek(stack) != '(')
                return -1; // invalid expression
            else
                pop(stack);
        }
        else // an operator is encountered
        {
            while (!isEmpty(stack) && Prec(exp[i]) <= Prec(peek(stack)))
                exp[++k] = pop(stack);
            push(stack, exp[i]);
        }

    }

    // pop all the operators from the stack
    while (!isEmpty(stack))
        exp[++k] = pop(stack );

    exp[++k] = '\0';
    printf( "%s\n", exp );
}

// Driver program to test above functions
int main()
{
    char exp[] = "a+b*(c^d-e)^(f+g*h)-i";
    infixToPostfix(exp);
    return 0;
}
```

Output:

```
abcd^e-fgh*+^*+i-
```

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

Related Questions:

- Binary Heap
- Delete all occurrences of a given key in a linked list
- How to create mergable stack?
- Deque | Set 1 (Introduction and Applications)
- A data structure for n elements and O(1) operations
- Convert left-right representation of a bianry tree to down-right

- [Print level order traversal line by line]
- [C Program for Red Black Tree Insertion]

Like 5 Tweet 0 +1 0

30 Comments GeeksQuiz Login

Recommend Share Sort by Best

Join the discussion…

aadvik • 8 months ago

else if (exp[i] == ')')
{
while (!isEmpty(stack) && peek(stack) != '(')
exp[++k] = pop(stack);
if (!isEmpty(stack) && peek(stack) != '(')
return -1; // invalid expression else
pop(stack);

Isn't the if statement wrong??..
Shouldn't it be lyk
else if (exp[i] == ')')
{
while (!isEmpty(stack) && peek(stack) != '(')
exp[++k] = pop(stack);
if (isEmpty(stack))
return -1; // invalid expression
else
pop(stack)

9 • Reply • Share ›

AK aadvik • 4 months ago

you are correct,that should be check for stack underflow

Reply • Share ›

deepak aadvik • 6 months ago

yes. I also think so.

Reply • Share ›

ANA • 9 months ago

According to the above algorithm
a+b*c+d should be abc*+d+ instead of abc*d++, correct ?

3 • Reply • Share ›

vipinkaushal ANA • 8 months ago

both expressions are correct the matter is how you indent

the expression
(a+b*c)+d=abc*+d+ and

a+(b*c+d)=abc*d++

1 • Reply • Share ›

S2K ANA • 8 months ago

a+b*c+d

-> a+(b*c)+d //coz * has higher Precedence over +

-> a+bc*+d

->a+(bc*+d) //here Associativity is from Right to Left

->a+(bc*d+) -> abc*d++

• Reply • Share ›

Mohit ANA • 9 months ago

yes, i also think the same. I think the post-fix expression: abc*d++ is correct if we perform "bc* + d" first and then add a to this.

• Reply • Share ›

Max Chipov • a year ago

there is a memory leak here
if (!stack->array)
return NULL;

you have to free the stack object

1 • Reply • Share ›

arjun_gowm • a month ago

The above code gives the output as acbc*++(+1 as postfix expression instead of a c b c * + 1 + + for infix expression a+(c+(b*c)+1)

• Reply • Share ›

cok • 2 months ago

how abt this:

how abt this:

passes most test cases including invalid expressions!!

#include<stdio.h>

#include<stdlib.h>

#include<malloc.h>

#include<string.h>

struct stack

{

int top;

int capacity;

char *arr;

see more

^ | ˅ • Reply • Share ›

himani • 2 months ago

@GeeksforGeeks... it is passing this test case: char exp[] = "a+b*(c^d-e)^(f+g*(h-i"; whereas it shouldn't.

^ | ˅ • Reply • Share ›

himani → himani • 2 months ago

it should be an invalid expression

^ | ˅ • Reply • Share ›

Bhagyashree • 3 months ago

This program gives following output for the expression "a+b*c-d^e^f":
abc*+de^f^-

But the expected answer is
abc*+def^^-

Please do reply

^ | ˅ • Reply • Share ›

any → Bhagyashree • 2 months ago

In reality this should be calculated like you told. But fyi
there is no such ('^') operator in c.

^ | ˅ • Reply • Share ›

Swati • 5 months ago

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

this returns an array of integer, while in peep,push and pop function we are using char instead of integer.
Isn't it wrong?

• Reply • Share ›

Ashish Jaiswal ➦ Swati • 2 months ago

Well i have the same doubt....but i did it for char...it worked anyhow... @geeksforgeeks

• Reply • Share ›

piyush jain • 5 months ago

why have u made the code so complicated?

• Reply • Share ›

Guest • 5 months ago

InfixtoPostfix has return type as int....still...no integer returned...y so??

• Reply • Share ›

Ashish Jaiswal ➦ Guest • 2 months ago

integer is returned in Case expression is invalid...which returns -1 in case '(' is not mached with closing bracket ')'.

• Reply • Share ›

RBK050 • 5 months ago

Can someone explain the precedence of ^ operator?

• Reply • Share ›

Ashwani • 7 months ago

where is the handling condition for invalid expressions like (d+)c+(a+b) ?

• Reply • Share ›

Guest • 7 months ago

in else if (exp[i] == ')')
this if (!isEmpty(stack) && peek(stack) != '(') expression will never be true so why it is used?
because at the same time stack cant be empty and its top element will be"(". is it possible??
i think we can remove this line

• Reply • Share ›

Kishore rajan • 7 months ago

Can we assign an character to an integer array without an typecast statement in push function ?

:

• Reply • Share ›

avik • 8 months ago

if exp[i] == ')' then what the if-else case doing ??

• Reply • Share ›

Abhishek chandel • 8 months ago

what changes should i make if i 've to convert a real expression.. i.e. number are used instead of variables(i.e. alphabets)...

• Reply • Share ›

RK ↪ Abhishek chandel • 7 months ago

just add a space after every number to make the output unambiguous.

• Reply • Share ›

amateur • 8 months ago

what is the running time of this function infixtopostfix?

• Reply • Share ›

RK ↪ amateur • 7 months ago

O(n), where n is the number of characters in the expression.

• Reply • Share ›

geek • 10 months ago

i am getting segmentation fault on using above code...can nyone resolv?

• Reply • Share ›

GeeksforGeeks Mod ↪ geek • 10 months ago

It seems to be working fine. Please see http://ideone.com/lWcAC3

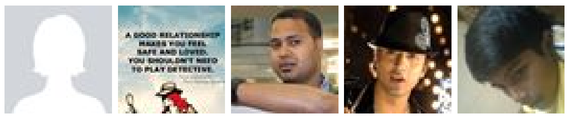
Could you let us know the compiler you used.

1 • Reply • Share ›

- GeeksQuiz

  Like

  13,821 people like GeeksQuiz.

  Facebook social plugin
-
- ## Categories
  - Articles (124)
    - Algorithms (24)
    - C (16)
    - C++ (17)
    - Data Structures (29)
    - DBMS (1)
    - Interview Experiences (6)
    - Java (2)
    - Operating Systems (1)
    - Puzzle (12)
    - Searching and Sorting (10)
  - Programs (35)
  - Quizzes (2,106)
    - Aptitude (2)
    - Computer Science Quizzes (2,103)
      - Algorithms (147)
      - C (207)
      - C++ (129)
      - Computer Organization and Architecture (1)
      - Data Structures (132)
      - DBMS (2)
      - GATE (1,406)
      - Java (51)
      - Operating Systems (28)
    - Web technologies (1)
-
- ## Recent Discussions
  - Vidhi

    Process P2 will start it's last I/O operation...

    Operating Systems | Process Synchronization | Question 4 · 1 day ago

- xerosanyam

  This article is slightly less clear in one...

  A Programmer's approach of looking at Array vs. Linked List · 1 day ago

- Aditya Goel

  Please include Heap overflow condition as well....

  Queue | Set 2 (Linked List Implementation) · 1 day ago

- Aditya Goel

  We can initialize rear as -1 with no change in...

  Queue | Set 1(Introduction and Array Implementation) · 1 day ago

- Sg

  Implementing this question leaves an extra...

  GATE | GATE-CS-2015 (Set 3) | Question 22 · 2 days ago

- jyuan92

  What if I want to handle the situation that,...

  Remove comments from a given C/C++ program · 2 days ago

-