**Experiment No. 2b**

**Title :** Conversion from infix to postfix expression.

**Problem Statement :** Write a C code for Conversion of infix to postfix expression

**Algorithm :**

**Step 1:** Start.

**Step 2:** Declare a stack of type integer and a top variable ,define corresponding functions for it i.e. push() and pop().

**Step 3:** Scan the Infix string from left to right.

**Step 4:** Initialize an empty stack.

**Step 5:** If the scanned character is an operand, add it to the Postfix string.

**Step 6:** If the scanned character is an operator and if the stack is empty push the character to stack.

**Step 7:** If the scanned character is an Operator and the stack is not empty, compare the precedence of the character with the element on top of the stack.

**Step 8:** If top Stack has higher precedence over the scanned character pop the stack else push the scanned character to stack. Repeat this step until the stack is not empty and top Stack has precedence over the character.

**Step 9:** Repeat 6 and 7 steps till all the characters are scanned.

**Step 10:** After all characters are scanned, we have to add any character that the stack may have to the Postfix string.

**Step 11:** If stack is not empty add top Stack to Postfix string and Pop the stack. Repeat this step as long as stack is not empty

**Stop 12:** Display the result

**Step 13:** Stop

**Program:**

#include <stdio.h>

#include <conio.h>

#include <ctype.h>

#define SIZE 10

char s[SIZE];//stack declaration

int top=-1;

void push(char elem)//function to push char to stack

{

s[++top]=elem;

}

char pop()//function to pop from stack

{

return(s[top--]);

}

int pr(char elem)//precedence function

{

switch(elem)

{

case '#': return 0;

case '(': return 1;

case '+':

case '-': return 2;

case '\*':

case '/': return 3;

}

}

void main()

{

char infx[50],pofx[50],ch,elem;

int i=0,k=0;

printf("\*\*\*\*\*\*\*\*\*\*\*\* INFIX TO POSTFIX CONVERSION \*\*\*\*\*\*\*\*\*\*\*\*\*");

printf("\nEnter infix expression : ");//input infix expression

scanf("%s",infx);

push('#');//push # to determine end while popping out

while( (ch=infx[i++]) != '\0')

{

if( ch == '(')

push(ch);

else if(isalnum(ch))

pofx[k++]=ch;

else if( ch == ')')

{

while( s[top] != '(')

pofx[k++]=pop();

elem=pop();

}

else

{

while( pr(s[top]) >= pr(ch) )

pofx[k++]=pop();

push(ch);

}

}

while( s[top] != '#')

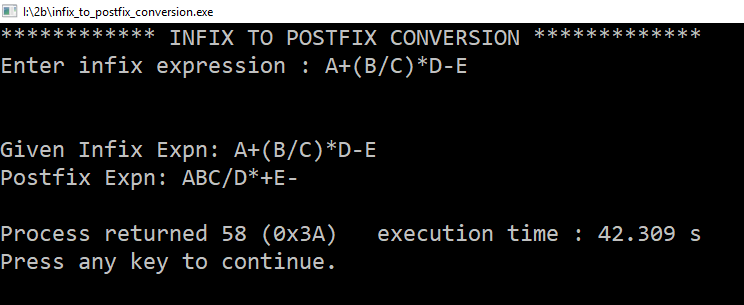
pofx[k++]=pop();

pofx[k]='\0';//add null character at the end of string

printf("\n\nGiven Infix Expn: %s \nPostfix Expn: %s\n",infx,pofx);

}

**Output:**



**Analysis:**

The program converts infix to postfix using stack data structure.

**Limitations:**

* Fails to convert if power operator is involved
* Only evaluates single digit expressions or single variable expression