ANAGHA ACHARYA

1BM19BT005

LAB-3

WAP to convert a given valid parenthesized infix arithmetic expression to postfix expression. The expression consists of single character operands and the binary operators + (plus), - (minus), \* (multiply) and / (divide)

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

#include<ctype.h>

#define SIZE 50

char stack[SIZE];

int top=-1;

push(char ele)

{

stack[++top]=ele;

}

char pop()

{

return(stack[top--]);

}

int pr(char symb)

{

if(symb == '^')

{

return (3);

}

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

{

return (2);

}

else if(symb == '+' || symb == '-')

{

return (1);

}

else

{

return(0);

}

}

void main()

{

int i=0, k=0;

char infix[50],postfix[50],ch,ele;

printf("Enter infix expression\n");

scanf("%s",infix);

push('#');

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

{

if(ch =='(')

push(ch);

else if(isalnum(ch))

postfix[k++]=ch;

else if(ch==')')

{

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

postfix[k++]=pop();

ele=pop();

}

else

{

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

{

postfix[k++]=pop();

}

push(ch);

}

}

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

postfix[k++]=pop();

postfix[k]='\0';

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

}

