AIM:-Program to convert infix to prefix.

Program:-

//Infix to prefix//

#include<stdio.h>

#include<stdlib.h>

#include<ctype.h>

#include<string.h>

#define max 100

int top=-1,a[max];

void push(char x)

{

a[++top]=x;

}

char pop()

{

if(top==-1)

return -1;

else

return a[top--];

}

int prcd(char c)

{

if (c==')')

return 0;

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

return 1;

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

return 2;

}

void strrev(char \*exp)

{

char temp[50];

int size=strlen(exp);

temp[size--]='\0';

int i=0;

while(exp[i]!='\0')

{

temp[size]=exp[i];

i++;

size--;

}

strcpy(exp,temp);

}

void infixtoprefix(char infix[max],char prefix[max])

{

char temp,x;

int i=0,j=0;

strrev(infix);

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

{

temp=infix[i];

if(isalnum(temp))

{

prefix[j++]=temp;

}

else if(temp==')')

push(temp);

else if(temp=='(')

{

while(x=pop()!=')')

{

prefix[j++]=x;

}

}

else

{

while(prcd(a[top])>=prcd(temp))

{

prefix[j++]=pop();

}

push(temp);

}

i++;

}

while(top!=-1)

prefix[j++]=pop();

prefix[j]='\0';

strrev(prefix);

}

int main()

{

char infix[max],prefix[max];

printf("Enter the infix expression\n");

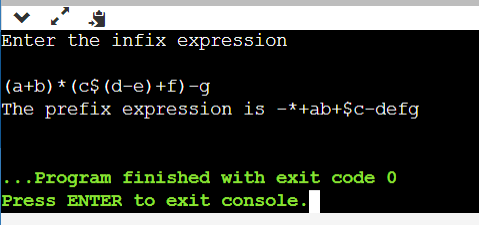
scanf("%s",infix);

infixtoprefix(infix,prefix);

printf("The prefix expression is %s\n",prefix);

return 0;

}



Algorithm:-

Step 1:First,reverse the infix expression given in the program.

Step 2:Scan the expression from left to right.

Step 3:Whenever the operand arrive,print then.

Step 4:If the operator arrives and the stack found is empty,then push the incoming operator into the stack.

Step 5:If the incoming operator has higher precedence or same precedence then replace the operator.

Step 6:When we reach the end the expression,pop and print all the operators from the top and the stack.

Step 7:If the operator is’(‘,then push into onto the stack.

Step 8:If the operator is ‘)’,then pop all the operators.

Step 9:At the end,reverse the output.