| S.No: 38 | Exp. Name: ***Write a C program to Convert an Infix expression into Postfix expression*** | **Date:2023-07-09** |
|---|---|---|

## Aim:

## Source Code:

Infix2PostfixMain.c

```
#include<string.h>
#include<stdio.h>
#include<ctype.h>
#define STACK_MAX_SIZE 20
char stack[STACK_MAX_SIZE];
int top=-1;
int isEmpty()
{
   if (top<0)
   return 1;
   else
   return 0;
}
void push(char x)
{
   if(top==STACK_MAX_SIZE-1)
   {
      printf("Stack is overflow.\n");
   }
   else
   {
      top=top+1;
      stack[top]=x;
   }
}
char pop()
{
   if(top < 0)
   {
      printf("Stack is underflow : unbalanced parenthesis\n");
      exit(0);
   }
   else
   return stack[top--];
}
int priority(char x)
{
   if(x == '(')
   return 0;
   if(x == '+' || x == '-')
   return 1;
   if(x == '*' || x == '/' || x == '%')
   return 2;
}
void convertInfix(char *e)
{
    int x;
    int k=0;
    char * p=(char *)malloc(sizeof(char)*strlen(e));
    while(*e!='\0')
    {
       if(isalnum(*e) )
       p[k++]=*e;
       else if(*e == '(')
       push(*e);
       else if(*e == ')')
       {
          while(!isEmpty() && (x= pop()) != '(')
          p[k++]=x;
       }
       else if (*e == '+' || *e == '-' || *e == '*' || *e == '/' || *e == '%')
       {
          while(priority(stack[top]) >= priority(*e))
          p[k++]=pop();
          push(*e);
          }
          else
          {
             printf("Invalid symbols in infix expression. Only alphanumeric and { '+',
 '-','*', '%%', '/' } are allowed.\n");
             exit(0);
             }
             e++;
             }
             while(top != -1)
             {
                x=pop();
                if(x == '(')
                {
                   printf("Invalid infix expression : unbalanced parenthesis.\n");
                   exit(0);
                   }
                   p[k++] = x;
                   }
                   p[k++]='\0';
                   printf("Postfix expression : %s\n",p);
                   }
                   int main()
                   {
                      char exp[20];
                      char *e,x;
                      printf("Enter the expression : ");
                      scanf("%s",exp);
                      e=exp;
                      convertInfix(e);
                      }
```

Execution Results - All test cases have succeeded!

Test Case - 1

| User Output |
|---|
| `Enter the expression :  A+B*(C-D)` |
| `Postfix expression : ABCD-*+` |

| Test Case - 2 |
|---|
| User Output |
| `Enter the expression :  A+B*C` |
| `Postfix expression : ABC*+` |