```
#include <stdio.h>
#include <string.h>
#include<ctype.h>
int top=-1,k=0;
char stack[30],post[30];
int precedencecheck(char op);
void push(char ch);
char pop();
void postfix(char expr[]);
void reverse(char *exp);
void brackets(char* exp);
void InfixtoPrefix(char *exp);
void evaluate(char exp[]);
int main()
{
     int i;
     char exp[50];
     printf("Enter expression");
     scanf("%s",exp);
     exp[strlen(exp)]=')';
     for(i=strlen(exp)-1;i>=0;i--)
     exp[i+1]=exp[i];
     exp[0]='(';
     InfixtoPrefix(exp);
     printf("prefix\n");
     for(i=0;i<k;i++)
     printf("%c",exp[i]);
     evaluate(post);
     return 0;
}
int precedencecheck(char op)
{
     if(op == '^')
          return 3;
     else if(op == '*' || op == '/')
          return 3;
     else if(op == '+' || op == '-')
          return 1;
     else
          return 0;

}
void push(char ch)
{
```

```
     if(top>=29)
          printf("Error");
     else
     {
          top++;
          stack[top]=ch;
     }
}
char pop()
{
     char item ;
     if(top<0)
          printf("error");
     else
     {
          item = stack[top];
          top--;
          return(item);
     }
}
void postfix(char expr[])
{
     int i=0;
     for (i = 0; i < strlen(expr); i++)
     {

if(expr[i]=='+'||expr[i]=='-'||expr[i]=='*'||expr[i]=='/'||exp
r[i]=='^')
          {

while(precedencecheck(stack[top])>=precedencecheck(expr[i]))
               {
                    post[k]=pop();
                    k++;
               }
               push(expr[i]);
          }
          else if(expr[i]==')'){
               char trash;
               while(stack[top]!='(')
               {
                    post[k]=pop();
                    k++;
               }
```

```
                trash = pop();
            }
            else if(expr[i]=='(')
                push(expr[i]);
            else{
                post[k]=expr[i];
                k++;
            }
     }
    strcpy(expr,post);
}
void reverse(char *exp){

    int size = strlen(exp);
    int j = size, i=0;
    char temp[size];

    temp[j--]='\0';
    while(exp[i]!='\0')
    {
        temp[j] = exp[i];
        j--;
        i++;
    }
    strcpy(exp,temp);
}
void brackets(char* exp){
    int i = 0;
    while(exp[i]!='\0')
    {
        if(exp[i]=='(')
            exp[i]=')';
        else if(exp[i]==')')
            exp[i]='(';
        i++;
    }
}
void InfixtoPrefix(char *exp){

    int size = strlen(exp);
    reverse(exp);
    brackets(exp);
    postfix(exp);
    reverse(exp);
```

```
}
void evaluate(char exp[]){
     char *e;
    int n1,n2,n3,num;
    e = exp;
    while(*e != '\0')
    {
        if(isdigit(*e))
        {
            num = *e - 48;
            push(num);
        }
        else
        {
            n1 = pop();
            n2 = pop();
            switch(*e)
            {
            case '+':
            {
                n3 = n1 + n2;
                break;
            }
            case '-':
            {
                n3 = n2 - n1;
                break;
            }
            case '*':
            {
                n3 = n1 * n2;
                break;
            }
            case '/':
            {
                n3 = n2 / n1;
                break;
            }
            }
            push(n3);
        }
        e++;
    }
    printf("\nThe result of expression =  %d\n\n",pop());
```

```
}
```