```
#include<stdio.h>
#include<string.h>
#include<limits.h>
#include<stdlib.h>
#include<ctype.h>
struct stack
{
   char ele;
   struct stack *next;
};
struct stack *top=NULL;
int stack1[20];
int top1=-1;
void push1(int x)
{
   stack1[++top1]=x;
}
int pop1()
{
   return stack1[top1--];
}
void push(int);
int pop();
int precedence(char);
void evaluation(char postfix[])
{
   int n1,n2,n3,num;
   int i=0;
   while(postfix[i]!='\0')
   {
      if(isdigit(postfix[i]))
```

```
        {
            num=postfix[i]-48;
            push1(num);
        }
        else
        {
            n1=pop1();
            n2=pop1();
            switch(postfix[i])
            {
                case '+':
                  n3=n1+n2;
                  break;
                case '-':
                  n3=n2-n1;
                  break;
                case '*':
                  n3=n1*n2;
                  break;
                case '/':
                  n3=n2/n1;
                  break;
                case '^':
                  n3=n2^n1;
            }
            push1(n3);
        }
        i++;
    }
    printf("\nThe result of expression %s=%d\n\n",postfix,n3);
}
```

```
int main()
{
    char infix[20],postfix[20];
    int i=0,j=0;
    printf("Enter infix expression");
    scanf("%s",infix);
    while(infix[i]!='\0')
    {
        if(isalnum(infix[i]))
        {
            postfix[j++]=infix[i];
        }
        else
        {
            if(top==NULL)
            {
                push(infix[i]);
            }
            else
            {
                while(top!=NULL&&(precedence(top->ele)>=precedence(infix[i])))
                {
                    postfix[j++]=pop();
                }
                push(infix[i]);
            }
        }
        ++i;
    }
    while(top!=NULL)
    {
        postfix[j++]=pop();
    }
    postfix[j]='\0';
    printf("%s",postfix);
    evaluation(postfix);

    return 0;
}
int precedence(char x)
{
    switch(x)
    {
        case '^':return 4;
        case '*':
        case '/':return 3;
        case '+':
        case '-':return 2;
        default:return 0;
    }
}
void push(int x)
{
    int item;
    struct stack *temp;
    temp=(struct stack*)malloc(sizeof(struct stack));
    temp->ele=x;
    if(top==NULL)
    {
        top=temp;
    }
    else
    {
        temp->next=top;
        top=temp;
    }
}
int pop()
{
    struct stack *tmp;
    char item;
    if(top==NULL)
    {
        printf("Empty stack");
    }
    else if(top->next==NULL)
    {
        tmp=top;
        item=top->ele;
        top=NULL;
        free(tmp);
    }
    else
    {
        tmp=top;
        item=top->ele;
        top=top->next;
        free(tmp);
    }
    return item;
}
```