# Practical-5

## 1. Write a program to evaluate postfix expression.

Aim:

To evaluate postfix expression.

Theory:

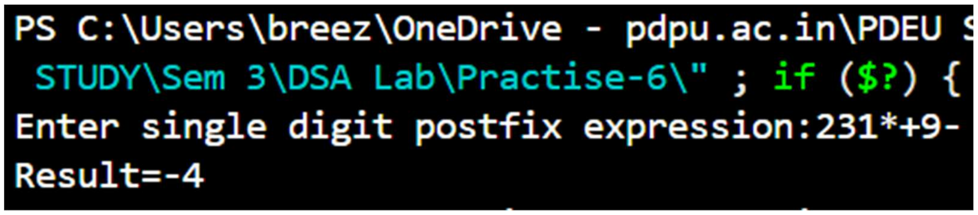
We will use stack to evaluate postfix expression in the given problem statement using structure pointer and also malloc function to assign memory to it.

Code:

```
#include<stdio.h>

#include<stdlib.h>

#include<ctype.h>

#include<string.h>


typedef struct stack

{

  int top,size;

  int *array;

}stack;


stack * createstack(int size)

{

  stack *s=(stack *)malloc(sizeof(stack));

  s->array=(int *)malloc(sizeof(int) * size);

  s->top=-1;

  s->size=size;

  return s;

}


int isFull(stack *s)
```

```
{
 if(s->top==s->size-1)
 {
  return 1;
 }
 else
 {
  return 0;
 }
}

int isEmpty(stack *s)
{
 if(s->top==-1)
 {
  return 1;
 }
 else
 {
  return 0;
 }
}

void push(stack *s,int item)
{
 if(!isFull(s))
 {
  s->top++;
  s->array[s->top]=item;
 }
}

int pop(stack *s)
{
 if(!isEmpty(s))
 {
  int item=s->array[s->top];
  s->top--;
  return item;
 }
 return 0;
}

int evaluate(char *expr,stack *s)
{
 int i=0;
 while(expr[i]!=')')
 {
  if(isdigit(expr[i]))
  {
   push(s,expr[i]-'0');
  }
  else
  {
   int A=pop(s);
   int B=pop(s);
   switch(expr[i])
   {
        case '+':push(s,B+A);break;
        case '-':push(s,B-A);break;
        case '*':push(s,B*A);break;
        case '/':push(s,B/A);break;
        case '^':push(s,B^A);break;
      }
    }
    i++;
  }
  return pop(s);
}

int main()
{
  char expr[100];
  printf("Enter single digit postfix expression:");
  scanf("%s",expr);
  int len=strlen(expr);
  expr[len]=')';
  expr[len+1]='\0';
  stack *s=createstack(len+1);
  int result=evaluate(expr,s);
  printf("Result=%d\n",result);
  return 0;
}
```

Output:



```
PS C:\Users\breez\OneDrive - pdpu.ac.in\PDEU S
 STUDY\Sem 3\DSA Lab\Practise-6\" ; if ($?) {
Enter single digit postfix expression:231*+9-
Result=-4
```

## 2. Convert a given expression from infix to postfix.

Aim:

To convert infix expression to postfix expression.

Theory:

We will use stack data structure to implement above problem statement.

Code:

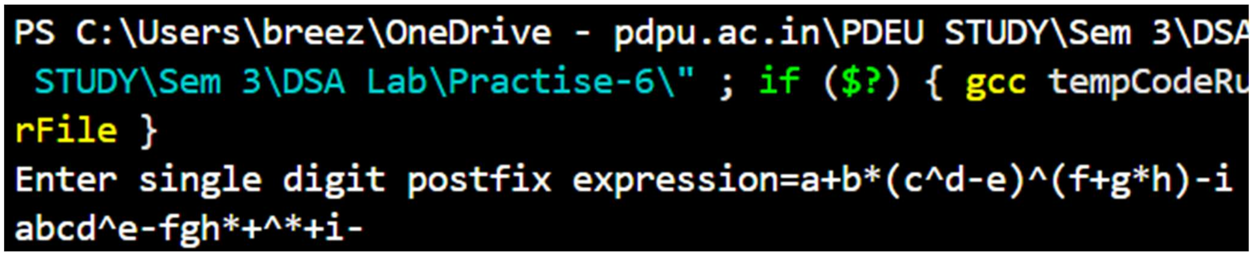
```
#include<stdio.h>

#include<stdlib.h>

#include<ctype.h>

#include<string.h>


struct Stack

{

 int top,size;

 char *array;

};


struct Stack* createStack(int size)

{

 struct Stack* s=(struct Stack*)malloc(sizeof(struct Stack));

 s->size=size;

 s->top=-1;

 s->array=(char*)malloc(sizeof(char)*size);

 return s;

}


int isEmpty(struct Stack* s)

{

 return s->top==-1;
```

```
}

void push(struct Stack* s,char c)
{
 s->array[++s->top]=c;
}

char pop(struct Stack* s)
{
 return s->array[s->top--];
}

char peek(struct Stack* s)
{
 return s->array[s->top];
}

int prec(char c)
{
 if(c=='^') return 3;
 if(c=='*'||c=='/') return 2;
 if(c=='+'||c=='-') return 1;
 return -1;
}

void infixToPostfix(char* expr)
{
 struct Stack* s=createStack(strlen(expr));
 int i,k=-1;
```

```
char *res=(char*)malloc(strlen(expr)+1);
for(i=0;expr[i];i++)
{
 if(isalnum(expr[i]))
 {
  res[++k]=expr[i];
 }
 else if(expr[i]=='(')
 {
  push(s,expr[i]);
 }
 else if(expr[i]==')')
 {
  while(!isEmpty(s)&&peek(s)!='(')
  {
   res[++k]=pop(s);
  }
  pop(s);
 }
 else
 {
  while(!isEmpty(s)&&prec(peek(s))>=prec(expr[i]))
  {
   res[++k]=pop(s);
  }
  push(s,expr[i]);
 }
}
while(!isEmpty(s))
```

```
 {
  res[++k]=pop(s);
 }
 res[++k]='\0';
 printf("%s\n",res);
}

int main()
{
 char expr[100];
 printf("Enter single digit postfix expression=");
 scanf("%s",expr);
 infixToPostfix(expr);
 return 0;
}
```

Output:



```
PS C:\Users\breez\OneDrive - pdpu.ac.in\PDEU STUDY\Sem 3\DSA
 STUDY\Sem 3\DSA Lab\Practise-6\" ; if ($?) { gcc tempCodeRu
rFile }
Enter single digit postfix expression=a+b*(c^d-e)^(f+g*h)-i
abcd^e-fgh*+^*+i-
```

**Link for all codes:**

https://github.com/PanavPatel06/DSA-Lab/tree/main/Practise-6