LAB-3

Write a program to convert a given valid parenthesized infix arithmetic expression to postfix expression. The expression consist of single character operands and binary operators.

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#define MAX 100
char st[MAX];
int top=-1;
void push(char st[], char);
char pop(char st[]);
void InfixtoPostfix(char source[], char target[]);
int getPriority(char);
int main()
{
    char infix[100], postfix[100];
    Printf("\n Enter any infix expression : ");
    gets(infix);
    strcpy(postfix,"");
    InfixtoPostfix(infix, postfix);
    Printf("\n The corresponding postfix expression
    puts(postfix);
    return 0;
}
void InfixtoPostfix(char source[], char target[]
{
    int i=0, j=0;
    char temp;
    strcpy(target,"");
```

```
while ( source[i] != '\0' )
{
    if ( source[i] == '(' )
    {
        push ( st, source[i] );
        i++;
    }
    else if ( source[i] == ')' )
    {
        while ( ( top != -1 ) && ( st[top] != '(' ))
        {
            target[j] = pop(st);
            j++;
        }
        temp = pop(st);
        i++;
    }
    else if ( isdigit ( source[i] ) || isalpha(source[i]))
    {
        target[j] = source[i];
        j++;
        i++;
    }
    else if ( source[i] == '+' || source[i] == '-' ||
              source[i] == '*' || source[i] == '/' ||
              source[i] == '^' )
    {
        while ( ( top != -1 ) && ( st[top] != '(' ) &&
                ( getpriority ( st[top] ) > getpriority(source[i])))
        {
            target[j] = pop(st);
            j++;
        }
        push ( st, source[i] );
        i++;
    }
```

```
else
    {
        printf("Incorrect element in expression");
        exit(2);
    }
while((top != -1) && (st[top] != '('))
    {
        target[j] = pop(st);
        j++;
    }
target[j] = '\0';
int getpriority(char op)
{
    if(op == '^')
        return 2;
    else if(op == '/' || op == '*' || op == '%')
        return 1;
    else if(op == '+' || op == '-')
        return 0;
}
void push(char st[], char val)
{
    if(top == MAX-1)
        printf("Stack overflow\n");
    else
    {   top++;
        st[top] = val;
    }
}
```

```
char pop(char st[])
{
    char val = ' ';
    if(top == -1)
        printf("\n Stack overflow");
    else
    {   val = st[top];
        top--;
    }
    return val;
}
```

output :-

Enter any infix expression : A+B-C*D

The corresponding postfix expression is: AB+CD*-

2) WAP to simulate the working of a queue of integers using an array.

```
#include<stdio.h>
#define N 5
int q[N];
int front = -1, rear = -1;
void insert(int);
int delete();
void display();
void main()
{   int n, choice;
    do
    {   printf("\n1) Insertion 2) Delete 3) Display");
        Exit \n");
```

```
printf("Enter your option : \n");
scanf("%d", &choice);
switch(choice)
{
  case 2:
    printf("Enter the number to be
                inserted into queue :\n");
    scanf("%d", &n);
    insert(n);
    break;

case 2:
    n = delete();
    if(n != -1)
    printf("\n the number deleted
                is : %d\n", n);
    break;

case 3:
    display();
    break;

case 4:
    exit(0);
    break;

default : printf("Invalid option\n");
    exit(0);
    break;
}
} while(choice != 4);
}
```

```
void insert (int num) {
    if (rear == MAX-1) {
        printf (" Stack is full \n");
    }
    else if (rear == -1 && front == -1) {
        rear++;
        front++;
        q[rear] = value;
    }
    else {  rear++;
        q[rear] = value;
    }
}

int delete()
{
    int value;
    if (front == -1 && rear == -1) {
        printf (" Stack is empty underflow \n");
    }
    else if (front == rear) {
        value = q[front];
        front = -1;
        rear = -1;
    }
    else {
        value = qu[front];
        front++;
    }
    return value;
}
```

```
int main()
{
    int choice=1, x;
    while(choice==1 && choice!=0)
    {
        printf("Press 1: Insert an element");
        printf("Press 2: Delete an element");
        printf("Press 3: Display the Element");
        printf("Enter your choice");
        scanf("%d", &choice);
    switch(choice)
    { case 1: printf("Enter the element
                which is to be inserted");
              scanf("%d", &x);
              enqueue(x);
              break;
      case 2: dequeue();
              break;
      case 3: display();
    }
  }
  return 0;
}
```

Output :-

1) Insert

2) Display

3) Delete

Enter your choice 1

Enter the element to be inserted 5

1) Insert

2) Display

3) Delete

Enter your choice 1

Enter the element to be inserted 6

1) Insert
2) Delet
3) Display
4) Exit

Enter your option
2
The number deleted is 3

1 Insert
2 Delete
3 Display
4 Exit

Enter your optio
4

03 WAP to stimulate the working of a circular queue:

```
#include<stdio.h>
#define max 6
int queue[max];
int front=-1;
int rear=-1;
void enqueue(int element)
{
    if(front==-1 && rear==-1)
    {   front=0;
        rear=0;
        queue[rear]=element;
    }
else if((rear+1)%max==front)
    {   printf("Queue is overflow");
    }
else {  rear=(rear+1)%max;
        queue[rear]=element;
}   }
```

```
int dequeue()
{
    if ((front == -1) && (rear == -1))
    {
        printf("\n Queue is underflow");
    }
    else if (front == rear)
    {
        printf("The dequeued element is %d",
                                    queue[front]);
        front = -1;
        rear = -1;
    }
    else
    {
        printf("\n The dequeued element is %d",
                                    queue[front]);
        front = (front+1) % max;
    }
}

void display()
{
    int i = front;
    if (front == -1 && rear == -1)
    {
        printf("\n Queue is empty");
    }
    else
    {
        printf("\n Element in a queue are:");
        while (i <= rear)
        {
            printf("%d", queue[i]);
            i = (i+1) % max;
        }
    }
}
```

```
void display()
{
    int i;
    printf("\n");
    if (front==-1 && rear==-1){
        printf("\n Queue is empty");
    }
    else
    {
        for(i=front;i<=rear;i++){
            printf("%d",q[i]);
        }
    }
}
```

Output :-

1) Insert
2) Delete
3) Display
4) Exit

Enter your option : 1

Enter the number to be inserted in the queue

3

1) Insert
2) Delete
3) Display
4) Exit

Enter your option
3

1) Insert
2) Display
3) Delete
Enter your choice 2
Elements in queue are 5, 6.