



Green University of Bangladesh
Department of Computer Science and Engineering(CSE)

Faculty of Sciences and Engineering
Semester: (Summer , Year:2022), B.Sc. in CSE (Day)

LAB REPORT NO : 4

Course Title: Structured Programming Lab
Course Code: CSE 106 Section:PC-213DA

Lab Experiment Name: Infix to Postfix & Circular Queue

Student Details

Name	ID
MD Dulal Hossain	213902116

Lab Date : 27-07-2022

Submission Date : 07-08-2022

Course Teacher's Name : Farhana Akther Sunny

[For Teachers use only: Don't Write Anything inside this box]

Lab Report Status

Marks:

Signature:.....

Comments:.....

Date:.....

//Infix to Postfix

Algorithm :

Step 1 : Scan the Infix Expression from left to right.

Step 2 : If the scanned character is an operand, append it with final Infix to Postfix string.

Step 3 : Else,

Step 3.1 : If the precedence order of the scanned(incoming) operator is greater than the precedence order of the operator in the stack (or the stack is empty or the stack contains a '(' push it on stack.

Step 3.2 : Else, Pop all the operators from the stack which are greater than or equal to in precedence than that of the scanned operator. After doing that Push the scanned operator to the stack. (If you encounter parenthesis while popping then stop there and push the scanned operator in the stack.)

Step 4 : If the scanned character is an '(' push it to the stack.

Step 5 : If the scanned character is an ')' pop the stack and output it until a '(' respectively is encountered, and discard both the parenthesis.

Step 6 : Repeat steps 2-6 until infix expression is scanned.

Step 7 : Print the output

Step 8 : Pop and output from the stack until it is not empty.

Source Code :

```
#include<stdio.h>
```

```
#include<ctype.h>
```

```
Char stack[100];
```

```
Int top = -1;
```

```

Void push(char x)
{
    Stack[++top] = x;
}

```

```

Char pop()
{
    If(top == -1)
        Return -1;
    Else
        Return stack[top--];
}

```

```

Int priority(char x)
{
    If(x == '(')
        Return 0;
    If(x == '+' || x == '-')
        Return 1;
    If(x == '*' || x == '/')
        Return 2;
    Return 0;
}

```

```

Int main()
{
    Char exp[100];
    Char *e, x;
    Printf("-----Enter the infix expression -----\\n");
    Scanf("%s",exp);
    Printf("\\n");
    E = exp;
    Printf("\\n -----The Postfix expression -----\\n");
}

```

```

While(*e != '\0')
{
    If(isalnum(*e))
        Printf("%c ",*e);
    Else if(*e == '(')
        Push(*e);
    Else if(*e == ')')
    {
        While((x = pop()) != '(')
            Printf("%c ", x);
    }
    Else
    {
        While(priority(stack[top]) >= priority(*e))
            Printf("%c ",pop());
        Push(*e);
    }
    E++;
}

While(top != -1)
{
    Printf("%c ",pop());
}return 0;
}

```

[My Code & Output](#)

main.c



Run

Output

Clear

```
1 #include<stdio.h>
2 #include<ctype.h>
3
4 char stack[100];
5 int top = -1;
6
7 void push(char x)
8 {
9     stack[++top] = x;
10 }
11
12 char pop()
13 {
14     if(top == -1)
15         return -1;
16     else
17         return stack[top--];
18 }
19
20 int priority(char x)
21 {
22     if(x == '(')
23         return 0;
24     if(x == '+' || x == '-')
25         return 1;
26     if(x == '*' || x == '/')
27         return 2;
28     return 0;
29 }
30
31 int main()
32 {
33     char exp[100];
34     char *e, x;
35     printf("-----Enter the infix expression\n");
36     scanf("%s",exp);
37     printf("\n");
38     e = exp;
39     printf("\n -----The Postfix expression\n");
40     while(*e != '\0')
41     {
42         if(isalnum(*e))
43             printf("%c ",*e);
44         else if(*e == '(')
45             push(*e);
46         else if(*e == ')')
47         {
48             while((x = pop()) != '(')
49                 printf("%c ", x);
50         }
51         else
52         {
53             while(priority(stack[top]) >=
54                 priority(*e))
55                 printf("%c ",pop());
56             push(*e);
57         }
58         e++;
59     }
60     while(top != -1)
61     {
62         printf("%c ",pop());
63     }return 0;
64 }
65
```

```
/tmp/Cf8iaRVRGJ.o
-----Enter the infix expression -----
(a+b)*c
-----The Postfix expression -----
a b + c *
```

 Introducing the most
interactive C Course.
Learn practically and get certified.

[Get Started!](#)

//Circular Queue

Algorithm :

Insert

Step 1: if $(\text{rear}+1)\% \text{max} = \text{front}$

write " overflow "

goto step 4

[end of if]

Step 2: if $\text{front} = -1$ and $\text{rear} = -1$

set $\text{front} = \text{rear} = 0$

else if $\text{rear} = \text{max} - 1$ and $\text{front} \neq 0$

set $\text{rear} = 0$

else

set $\text{rear} = (\text{rear} + 1) \% \text{max}$

[end of if]

Step 3: set $\text{queue}[\text{rear}] = \text{val}$

Step 4: Exit

Delete

Step 1: if $\text{front} = -1$

write " underflow "

goto step 4

[end of if]

Step 2: set $\text{val} = \text{queue}[\text{front}]$

Step 3: if $\text{front} = \text{rear}$

set $\text{front} = \text{rear} = -1$

```
else
if front = max -1
set front = 0
else
set front = front + 1
[end of if]
[end of if]
```

Step 4: exit

Source Code :

```
#include<stdio.h>
# define MAX 5
Int cqueue_arr[MAX];
Int front = -1;
Int rear = -1;
Void insert(int item)
{
If((front == 0 && rear == MAX-1) || (front == rear+1))
{
Printf("Queue Overflow \n");
Return;
}
If(front == -1)
{
Front = 0;
Rear = 0;
}
Else
{
If(rear == MAX-1)
Rear = 0;
Else
Rear = rear+1;
```

```

}
Cqueue_arr[rear] = item ;
}
Void deletion()
{
If(front == -1)
{
Printf("Queue Underflown\n");
Return ;
}
Printf("Element deleted from queue is : %d\n",cqueue_arr[front]);
If(front == rear)
{
Front = -1;
Rear=-1;
}
Else
{
If(front == MAX-1)
Front = 0;
Else
Front = front+1;
}
}
Void display()
{
Int front_pos = front,rear_pos = rear;
If(front == -1)
{
Printf("Queue is empty\n");
Return;
}
Printf("Queue elements :");
If( front_pos <= rear_pos )
While(front_pos <= rear_pos)

```



```

{
Printf("%d ",cqueue_arr[front_pos]);
Front_pos++;
}
Else
{
While(front_pos <= MAX-1){
Printf("%d ",cqueue_arr[front_pos]);
Front_pos++;
}
Front_pos = 0;
While(front_pos <= rear_pos)
{
Printf("%d ",cqueue_arr[front_pos]);
Front_pos++;
}
}
Printf("\n");
}
Int main()
{
Int choice,item;
Do
{
Printf("1.Insert \n");
Printf("2.Delete \n");
Printf("3.Display \n");
Printf("4.Quit \n");
Printf("\n");
Printf("Enter your choice : ");
Scanf("%d",&choice);
Switch(choice)
{
Case 1 :
Printf("Input element for insertion in queue : ");

```

```
Scanf("%d", &item);
Insert(item);
Break;
Case 2 :
Deletion();
Break;
Case 3:
Display();
Break;
Case 4:
Break;
Default:
Printf("Wrong choice\n");
}
}while(choice!=4);
Return 0;
}
```

My code & Output

main.c



Run

Output

Clear

```
1 #include<stdio.h>
2 # define MAX 5
3 int cqueue_arr[MAX];
4 int front = -1;
5 int rear = -1;
6 void insert(int item)
7 {
8     if((front == 0 && rear == MAX-1) || (front == rear
9         +1))
10     {
11         printf("Queue Overflow \n");
12         return;
13     }
14     if(front == -1)
15     {
16         front = 0;
17         rear = 0;
18     }
19     else
20     {
21         if(rear == MAX-1)
22             rear = 0;
23         else
24             rear = rear+1;
25         cqueue_arr[rear] = item ;
26     }
27 void deletion()
28 {
29     if(front == -1)
30     {
31         printf("Queue Underflow\n");
32         return ;
33     }
34     printf("Element deleted from queue is : %d\n"
35         ,cqueue_arr[front]);
36     if(front == rear)
37     {
38         front = -1;
39         rear=-1;
40     }
41     else
42     {
43         if(front == MAX-1)
44             front = 0;
45         else
46             front = front+1;
47     }
48 void display()
49 {
50     int front_pos = front,rear_pos = rear;
51     if(front == -1)
52     {
53         printf("Queue is empty\n");
54         return;
55     }
56     printf("Queue elements :");
57     if( front_pos <= rear_pos )
58         while(front_pos <= rear_pos)
59     {
60         printf("%d ",cqueue_arr[front_pos]);
61         front_pos++;
62     }
63     else
64     {
65         while(front_pos <= MAX-1){
66             printf("%d ",cqueue_arr[front_pos]);
67             front_pos++;
68         }
69         while(rear_pos <= MAX-1){
70             printf("%d ",cqueue_arr[rear_pos]);
71             rear_pos++;
72         }
73     }
74 }
```

```
/tmp/HyUGNJ4yki.o
1.Insert
2.Delete
3.Display
4.Quit
Enter your choice : 1
Input element for insertion in queue : 4
1.Insert
2.Delete
3.Display
4.Quit
Enter your choice : 1
Input element for insertion in queue : 6
1.Insert
2.Delete
3.Display
4.Quit
Enter your choice : 3
Queue elements :4 6
1.Insert
2.Delete
3.Display
4.Quit
Enter your choice : 2
Element deleted from queue is : 4
1.Insert
2.Delete
3.Display
4.Quit
Enter your choice : 2
Element deleted from queue is : 6
1.Insert
2.Delete
3.Display
4.Quit
Enter your choice : 2
Queue Underflow
1.Insert
2.Delete
3.Display
4.Quit
Enter your choice : 4
```

 Introducing the most
interactive C Course.
Learn practically and get certified.

[Get Started!](#)

```
main.c
37 front = -1;
38 rear=-1;
39 }
40 else
41 {
42 if(front == MAX-1)
43 front = 0;
44 else
45 front = front+1;
46 }
47 }
48 void display()
49 {
50 int front_pos = front, rear_pos = rear;
51 if(front == -1)
52 {
53 printf("Queue is empty\n");
54 return;
55 }
56 printf("Queue elements :");
57 if( front_pos <= rear_pos )
58 while(front_pos <= rear_pos)
59 {
60 printf("%d ", cqueue_arr[front_pos]);
61 front_pos++;
62 }
63 else
64 {
65 while(front_pos <= MAX-1){
66 printf("%d ", cqueue_arr[front_pos]);
67 front_pos++;
68 }
69 front_pos = 0;
70 while(front_pos <= rear_pos)
71 {
72 printf("%d ", cqueue_arr[front_pos]);
73 front_pos++;
74 }
75 }
76 printf("\n");
77 }
78 int main()
79 {
80 int choice,item;
81 do
82 {
83 printf("1.Insert \n");
84 printf("2.Delete \n");
85 printf("3.Display \n");
86 printf("4.Quit \n");
87 printf("\n");
88 printf("Enter your choice : ");
89 scanf("%d",&choice);
90 switch(choice)
91 {
92 case 1 :
93 printf("Input element for insertion in queue : ");
94 scanf("%d", &item);
95 insert(item);
96 break;
97 case 2 :
98 deletion();
99 break;
100 case 3:
101 display();
102 break;
103 case 4:
104 break;
105 default:
```

Output

Clear

/tmp/HyUGNJ4yki.o

1.Insert
2.Delete
3.Display
4.Quit

Enter your choice : 1
Input element for insertion in queue : 4
1.Insert
2.Delete
3.Display
4.Quit

Enter your choice : 1
Input element for insertion in queue : 6
1.Insert
2.Delete
3.Display
4.Quit

Enter your choice : 3
Queue elements :4 6
1.Insert
2.Delete
3.Display
4.Quit

Enter your choice : 2
Element deleted from queue is : 4
1.Insert
2.Delete
3.Display
4.Quit

Enter your choice : 2
Element deleted from queue is : 6
1.Insert
2.Delete
3.Display
4.Quit

Enter your choice : 2
Queue Underflown
1.Insert
2.Delete
3.Display
4.Quit

Enter your choice : 4

Introducing the most
interactive C Course.

Learn practically and get certified.

Get Started!