



SOMAIYA
VIDYAVIHAR

K J Somaiya Institute of Technology

An Autonomous Institute Permanently Affiliated to the University of Mumbai

DEPARTMENT OF INFORMATION TECHNOLOGY

Course Name and Code: Data Structures Lab (ITL302)

Semester: III (SYIT)

Academic Year: 2023-24 (Odd Semester)

Name: parnika gangrade

Experiment No. 03

Aim: Implementations of Infix to Postfix Expression for real-world application.

Objectives:

1. To introduce the concepts of data structures and analysis procedure.
2. To conceptualize linear data structures and its implementation for various real-world applications.

Program:

```
#include<stdio.h>
#include<stdlib.h>
#include<ctype.h>
#include<string.h>
```

```
#define SIZE 100
char stack[SIZE];
int top = -1;
```

```
void push(char item)
{
    if(top >= SIZE-1)
    {
        printf("\n Stack Overflow.");
    }
    else
    {
        top = top+1;
        stack[top] = item;
    }
}
```

```
char pop()
```

```

{
    char item ;

    if(top <0)
    {
        printf("stack under flow: invalid infix expression");
        getchar();
        exit(1);
    }
    else
    {
        item = stack[top];
        top = top-1;
        return(item);
    }
}

int is_operator(char symbol)
{
    if(symbol == '^' || symbol == '*' || symbol == '/' || symbol == '+' || symbol == '-')
    {
        return 1;
    }
    else
    {
        return 0;
    }
}

int precedence(char symbol)
{
    if(symbol == '^')
    {
        return(3);
    }
    else if(symbol == '*' || symbol == '/')
    {
        return(2);
    }
    else if(symbol == '+' || symbol == '-')
    {
        return(1);
    }
    else
    {
        return(0);
    }
}

void InfixToPostfix(char infix_exp[], char postfix_exp[])

```

```

{
    int i, j;
    char item;
    char x;

    push('(');
    strcat(infix_exp, "");

    i=0;
    j=0;
    item=infix_exp[i];

    while(item != '\0')
    {
        if(item == '(')
        {
            push(item);
        }
        else if( isdigit(item) || isalpha(item))
        {
            postfix_exp[j] = item;
            j++;
        }
        else if(is_operator(item) == 1)
        {
            x=pop();
            while(is_operator(x) == 1 && precedence(x)>= precedence(item))
            {
                postfix_exp[j] = x;
                j++;
                x = pop();
            }
            push(x);

            push(item);
        }
        else if(item == ')')
        {
            x = pop();
            while(x != '(')
            {
                postfix_exp[j] = x;
                j++;
                x = pop();
            }
        }
        else
        {
            printf("\nInvalid infix Expression.\n");
            getchar();
            exit(1);
        }
    }
}

```

```

        }
        i++;
        item = infix_exp[i];
    }
    if(top>0)
    {
        printf("\nInvalid infix Expression.\n");
        getchar();
        exit(1);
    }
}
postfix_exp[j] = '\0';
}

```

```

int main()
{
    char infix[SIZE], postfix[SIZE];

    printf("\n Enter Infix expression : ");
    gets(infix);

    InfixToPostfix(infix,postfix);
    printf(" Postfix Expression: ");
    puts(postfix);

    return 0;
}

```

Output:

```

● dl406@itadmin:~/Desktop/Manali$ cd "/home/dl406/1
Enter Infix expression: A+B+C+D
Postfix Expression: AB+C+D+
● dl406@itadmin:~/Desktop/Manali$ cd "/home/dl406/1
Enter Infix expression: (A+B)*(C+D)
Postfix Expression: AB+CD+*

```