**Ganpat University**

**Faculty of Engineering & Technology**

**Computer Science & Engineering**

**Practical\_4**

***Name:- Dwij Vatsal Desai***

***Sem:- 3***

***Sub: - DS(Data Steucture)***

***Enrollment No.:- 23162121027***

**4) Use of Stack**

Parishram is a 7th semester, who is studying at GUNI-ICT. During his “Compiler Design”

course, his course faculty explained him that compiler work differently while it does evaluation

of an expression due to below reasons:

Infix expressions are readable and solvable by humans because of easily distinguishable order of

operators, but compiler doesn&#39;t have integrated order of operators.

To avoid this traversing, Infix expressions are converted to postfix expression before evaluation.

Make a program to convert infix expression into postfix using stack.

Code:

#include <stdio.h>

#include <ctype.h>

#define Dwij 100

char stack[Dwij];

int top = -1;

char expression[Dwij];

char \*e;

void push(char);

char pop();

int priority(char);

void infixToPostfix();

int main()

{

    printf("Enter the expression: ");

    scanf("%s", expression);

    printf("\n");

    e = expression;

    infixToPostfix();

    return 0;

}

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;

}

void infixToPostfix()

{

    char x;

    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 (top != -1 && priority(stack[top]) >= priority(\*e))

                printf("%c ", pop());

            push(\*e);

        }

        e++;

    }

    while (top != -1)

    {

        printf("%c ", pop());

    }

}

Image(Output):

