**Name**: Darshan Deepak Bhokare

**Roll No.** 26 **PRN:** 12210089

**Branch**: AI&DS-A

**Subject:** Advanced Data Structure

**Infix to Postfix**

#include <stdio.h>

#define MAX 30

char stack[30];

int top = -1;

int isEmpty()

{

    return top == -1;

}

int isFull()

{

    return top == MAX - 1;

}

char peek()

{

    return stack[top];

}

char pop()

{

    if (isEmpty())

        return -1;

    char ch = stack[top];

    top--;

    return (ch);

}

void push(char oper)

{

    if (isFull())

        printf("Stack Full!!!!");

    else

    {

        top++;

        stack[top] = oper;

    }

}

int checkIfOperand(char ch)

{

    return (ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z');

}

int precedence(char ch)

{

    switch (ch)

    {

    case '+':

    case '-':

        return 1;

    case '\*':

    case '/':

        return 2;

    case '^':

        return 3;

    }

    return -1;

}

void Infix\_to\_Postfix\_Conversion(char \*infix, char \*postfix)

{

    int i, j = 0;

    for (i = 0; infix[i]; ++i)

    {

        if (checkIfOperand(infix[i]))

            postfix[j++] = infix[i];

        else if (infix[i] == '(')

            push(infix[i]);

        else if (infix[i] == ')')

        {

            while (!isEmpty() && peek() != '(')

                postfix[j++] = pop();

            if (!isEmpty() && peek() != '(')

                return;

            else

                pop();

        }

        else

        {

            while (!isEmpty() && precedence(infix[i]) <= precedence(peek()))

                postfix[j++] = pop();

            push(infix[i]);

        }

    }

    while (!isEmpty())

        postfix[j++] = pop();

    postfix[j] = '\0';

}

int main()

{

    char infix[MAX];

    char postfix[MAX];

    int choice;

    do

    {

        printf("\nMenu:\n");

        printf("1. Enter infix\n");

        printf("2. Infix to postfix\n");

        printf("4. Exit\n");

        printf("Enter your choice: ");

        scanf("%d", &choice);

        switch (choice)

        {

        case 1:

            printf("Enter the infix : ");

            scanf("%s", infix);

            break;

        case 2:

            printf("\nInfix Expression: %s\n", infix);

            Infix\_to\_Postfix\_Conversion(infix, postfix);

            printf("Postfix Expression: %s\n", postfix);

            break;

        case 4:

            printf("Exit\n");

            break;

        default:

            printf("Invalid choice!\n");

            break;

        }

    } while (choice != 3);

    return 0;

}