ABHINAND K PRASAD

S7 CSA

ROLL NO: 4

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

#define MAX 100

typedef struct {

    int top;

    char items[MAX];

} Stack;

void push(Stack \*s, char c) {

    if (s->top < MAX - 1) {

        s->items[++s->top] = c;

    } else {

        printf("Stack overflow\n");

    }

}

char pop(Stack \*s) {

    if (s->top >= 0) {

        return s->items[s->top--];

    } else {

        printf("Stack underflow\n");

        return '\0';

    }

}

char peek(Stack \*s) {

    if (s->top >= 0) {

        return s->items[s->top];

    } else {

        return '\0';

    }

}

int isEmpty(Stack \*s) {

    return s->top == -1;

}

int precedence(char op) {

    switch (op) {

        case '+':

        case '-':

            return 1;

        case '\*':

        case '/':

            return 2;

        case '^':

            return 3;

        default:

            return 0;

    }

}

void toPostfix(char \*infix, char \*postfix) {

    Stack s;

    s.top = -1;

    int i = 0, k = 0;

    while (infix[i] != '\0') {

        if (isalnum(infix[i])) {

            postfix[k++] = infix[i];

        } else if (infix[i] == '(') {

            push(&s, infix[i]);

        } else if (infix[i] == ')') {

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

                postfix[k++] = pop(&s);

            }

            pop(&s);

        } else {

            while (!isEmpty(&s) && precedence(peek(&s)) >= precedence(infix[i])) {

                postfix[k++] = pop(&s);

            }

            push(&s, infix[i]);

        }

        i++;

    }

    while (!isEmpty(&s)) {

        postfix[k++] = pop(&s);

    }

    postfix[k] = '\0';

}

void toTAC(char \*postfix) {

    Stack s;

    s.top = -1;

    int tempCount = 1;

    char tempVar[10];

    printf("Three Address Code:\n");

    for (int i = 0; postfix[i] != '\0'; i++) {

        if (isalpha(postfix[i])) {

            push(&s, postfix[i]);

        }else {

            char op1 = pop(&s);

            if(isdigit(op1)){

                char op2 = pop(&s);

                sprintf(tempVar, "%d", tempCount++);

                if(postfix[i] == '=')

                    printf("%c %c %c%c\n", op2, postfix[i], op2, op1);

                else

                    if (isdigit(op2))

                        printf("t%s = t%c %c t%c\n", tempVar, op2, postfix[i], op1);

                    else{

                        printf("t%s = %c %c t%c\n", tempVar, op2, postfix[i], op1);

                    }

                push(&s, tempVar[0]);

            }else{

                char op2 = pop(&s);

                sprintf(tempVar, "%d", tempCount++);

                if(postfix[i] == '=')

                    printf("%c %c %c\n", op2, postfix[i], op1);

                else

                    if (isdigit(op2)){

                        printf("t%s = t%c %c %c\n", tempVar, op2, postfix[i], op1);

                    }

                    else

                        printf("t%s = %c %c %c\n", tempVar, op2, postfix[i], op1);

                push(&s, tempVar[0]);

            }

        }

    }

}

int main() {

    char infix[MAX], postfix[MAX];

    printf("Enter infix expression: ");

    scanf("%s", infix);

    toPostfix(infix, postfix);

    printf("\nPostfix expression: %s\n", postfix);

    toTAC(postfix);

    return 0;

}

**OUTPUT**

