Name: Dhruv Rupareliya (DSY)

Roll No: 71

Experiment no: 3

Code:

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>
#define SIZE 100
char stack[SIZE];
int top = -1;
void push(char);
char pop();
int is_op(char);
int precedence(char);
void ItoP(char infix[], char postfix[]);
int main() {
 char infix[SIZE], postfix[SIZE];
 printf("Enter Infix expression: ");
 gets(infix);
 ItoP(infix, postfix);
 printf("Postfix Expression: %s\n", postfix);
 return 0;
}
void push(char a) {
 if (top >= SIZE - 1) {
 printf("Stack Overflow.\n");
 } else {
 top = top + 1;
 stack[top] = a;
```

```
 }
}
char pop() {
 char item;
 if (top < 0) {
 printf("Stack underflow: Invalid infix expression\n");
 return '\0';
 } else {
 item = stack[top];
 top = top - 1;
 return item;
 }
}
int is_op(char s) {
 switch (s) {
 case '^':
 case '*':
 case '/':
 case '+':
 case '-':
 return 1;
 default:
 return 0;
 }
}
int precedence(char s) {
 switch (s) {
 case '^':
 return 3;
 case '*':
 case '/':
```

```
 return 2;
 case '+':
 case '-':
 return 1;
 default:
 return 0;
 }
}
void ItoP(char infix[], char postfix[]) {
 int i, j;
 char item;
 char x;
 push('(');
 strcat(infix, ")");
 int infix_length = strlen(infix);
 if (infix_length >= SIZE) {
 printf("Invalid infix expression: Too long.\n");
 return;
 }
 i = 0;
 j = 0;
 item = infix[i];
 while (item != '\0') {
 if (item == '(') {
 push(item);
 } else if (isdigit(item) || isalpha(item)) {
 postfix[j] = item;
 j++;
 } else if (is_op(item) == 1) {
 x = pop();
 while (is_op(x) == 1 && precedence(x) >= precedence(item)) {
```

```
postfix[j] = x;
j++;
x = pop();
}
push(x);
push(item);
} else if (item == ')') {
x = pop();
while (x != '(') {
postfix[j] = x;
j++;
x = pop();
}
} else {
printf("Invalid infix Expression.\n");
return;
}
i++;
item = infix[i];
}
if (top >= 0) {
printf("Invalid infix Expression.\n");
return;
}
postfix[j] = '\0';
}
```

Output:

```
Enter Infix expression: (A*X+(B*C))
Postfix Expression: AX*BC*+
```

```
Enter Infix expression: ((A-B)*(A+B))
Postfix Expression: AB-AB+*
```