

## EXPERIMENT 2

### Code:

#### Infix to Postfix Conversion

```
#include <stdio.h>
#include <stdlib.h>
char stack[100], infix[100], postfix[100];
int top=-1;
void push(char p){
    top++;
    stack[top]=p;
}
char pop()
{
    if(top<=-1)
    {
        exit(0);
        return '\0';
    }
    return (stack[top--]);
}
int priority(char op){
    if(op=='+'||op=='-')
        return 0;
    else if(op=='/'||op=='*')
        return 1;
    else if(op=='%') //diff priorityyyyyyyyyyy
        return 2;
    return -1;
}
void infixtopostfix(){
    char x,y;
    int i=0,j=0;
```

```

while(infix[i]!='\0'){
    x=infix[i];
    if(x>='A'&& x<='Z' || x>='a'&& x<='z' || x>='0'&& x<='9'){
        postfix[j]=x;
        j++;
    }
    else if(x=='('){
        push(x);
    }
    else if(x==')'){
        while(stack[top]!='('){
            y=pop();
            postfix[j]=y;
            j++;
        }
        top--;
    }
    else if(x=='+'||x=='-'||x=='*'||x=='/'||x=='%'){
        if(top==-1) //empty stack condition
        {
            push(x);
        }
        else if(priority(x)>priority(stack[top])){ //condition
            push(x);
        }
        else{
            while(top!=-1 && priority(x)<=priority(stack[top])) //top
condition
            {
                y=pop();
                postfix[j]=y;
                j++;
            }
        }
    }
}

```

```

        }
        push(x);
    }
}
else if(x!='\n')
    printf("Invalid expression: %c",x);
    i++;
}
while(top!=-1){
    y=pop();
    postfix[j]=y;
    j++;
}
postfix[j]='\0';
}
void main(){
    printf("Enter expression: ");
    fgets(infix,sizeof(infix),stdin);
    infixtopostfix();
    int cnt;
    printf("\nPostfix expression: ");
    fflush(stdout);
    for(cnt=0;postfix[cnt]!='\0';cnt++){
        printf("%c",postfix[cnt]);
        fflush(stdout);
    }
}

```

**Output:**

```

E:\piyu\Computer Engg\Sem 3\DSA\Stack>.\a
Enter expression: a+b*c+d

Postfix expression: abc*+d+

```

## Postfix Evaluation

### Code:

```
#include <stdio.h>
#include <stdlib.h>
char stack[100], infix[100], postfix[100];
int top=-1;
void push(char p){
    top++;
    stack[top]=p;
}
char pop()
{
    if(top<=-1)
    {
        exit(0);
        return '\0';
    }
    return (stack[top--]);
}
int priority(char op){
    if(op=='+'||op=='-')
        return 0;
    else if(op=='/'||op=='*')
        return 1;
    else if(op=='%') //diff priorityyyyyyyyyyy
        return 2;
    return -1;
}
void infixtopostfix(){
    char x,y;
    int i=0,j=0;
    while(infix[i]!='\0'){
```

```

x=infix[i];
if(x>='A'&& x<='Z' || x>='a'&& x<='z' || x>='0'&& x<='9'){
    postfix[j]=x;
    j++;
}
else if(x=='('){
    push(x);
}
else if(x==')'){
    while(stack[top]!='('){
        y=pop();
        postfix[j]=y;
        j++;
    }
    top--;
}
else if(x=='+'||x=='-'||x=='*'||x=='/'||x=='%'){
    if(top==-1) //empty stack condition
    {
        push(x);
    }
    else if(priority(x)>priority(stack[top])){
        push(x);
    }
    else{
        while(top!=-1 && priority(x)<=priority(stack[top]))
        {
            y=pop();
            postfix[j]=y;
            j++;
        }
        push(x);
    }
}

```

```

        }
    }
    else if(x!='\n')
        printf("Invalid expression: %c",x);
    i++;
}
while(top!=-1){
    y=pop();
    postfix[j]=y;
    j++;
}
postfix[j]='\0';
}
void main(){
    printf("Enter expression: ");
    fgets(infix,sizeof(infix),stdin);
    infixtopostfix();
    int cnt;
    printf("\nPostfix expression: ");
    fflush(stdout);
    for(cnt=0;postfix[cnt]!='\0';cnt++){
        printf("%c",postfix[cnt]);
        fflush(stdout);
    }
}

```

**Output:**

```

E:\piyu\Computer Engg\Sem 3\DSA\Stack>.\a
Enter expression: 43*25*+8-
Result: 14.000000

```