Parshvanath Charitable Trust's
A. P. SHAH INSTITUTE OF TECHNOLOGY
(Approved by AICTE New Delhi & Govt. of Maharashtra, Affiliated to University of Mumbai)
(Religious Jain Minority)

Academic Year: 2024-25 **Name of Student:Richa Thanekar**
Semester: VI **Student ID:22102167**
Class : TE (Div-C) **Date of Performance:11/3/25**
Subject: SPCCL **Date of Submission: 11/3/25**
Name of Instructor: Prof. Vandana V. **Batch: C3**

Experiment 4

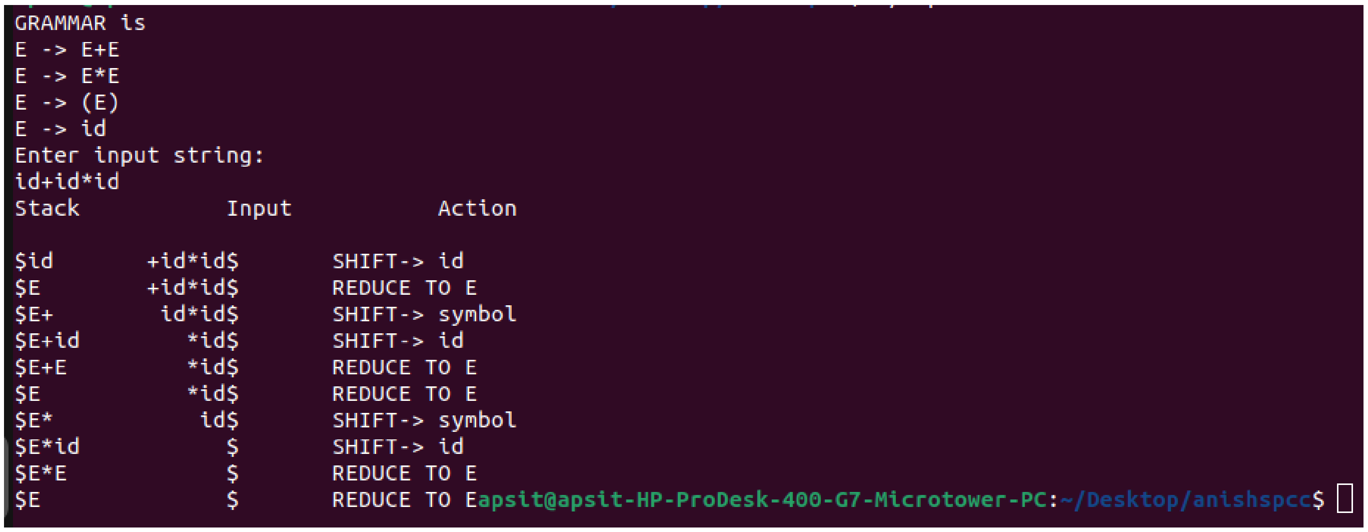
Aim: **Design and implementation of Operator Precedence Parser**

Theory: -

```
#include <stdio.h>
#include <string.h>

int k = 0, z = 0, i = 0, j = 0, c = 0;
char a[20], ac[20], stk[20], act[10];

void check();

int main() {


    puts("GRAMMAR is");
    puts("E -> E+E");
    puts("E -> E*E");
    puts("E -> (E)");
    puts("E -> id");


    puts("Enter input string: ");
    fgets(a, sizeof(a), stdin);
```

```
   a[strcspn(a, "\n")] = '\0';

    c = strlen(a);
    strcpy(act, "SHIFT->");

    puts("Stack\t\tInput\t\tAction");


    for (k = 0, i = 0; j < c;) {
        if (a[j] == 'i' && a[j + 1] == 'd') {

            stk[i] = a[j];
            stk[i + 1] = a[j + 1];
            stk[i + 2] = '\0';
            a[j] = ' ';
            a[j + 1] = ' ';
            printf("\n$%s\t%s$\t%s id", stk, a, act);
            check();
            i += 2;
            j += 2;
        } else if (a[j] != ' ') {

            stk[i] = a[j];
            stk[i + 1] = '\0';
            a[j] = ' ';
            printf("\n$%s\t%s$\t%s symbol", stk, a, act);
            check();
            i++;
            j++;
        } else {
            j++;
        }
    }
    return 0;
}

void check() {

    strcpy(ac, "REDUCE TO E");
```

```
for (z = 0; z < c; z++) {
    if (stk[z] == 'i' && stk[z + 1] == 'd') {
        stk[z] = 'E';
        stk[z + 1] = '\0';
        printf("\n$%s\t%s$\t%s", stk, a, ac);
        i--;
    }
}


for (z = 0; z < c; z++) {
    if (stk[z] == 'E' && stk[z + 1] == '+' && stk[z + 2] == 'E') {
        stk[z] = 'E';
        stk[z + 1] = '\0';
        stk[z + 2] = '\0';
        printf("\n$%s\t%s$\t%s", stk, a, ac);
        i -= 2;
    }
}


for (z = 0; z < c; z++) {
    if (stk[z] == 'E' && stk[z + 1] == '*' && stk[z + 2] == 'E') {
        stk[z] = 'E';
        stk[z + 1] = '\0';
        stk[z + 2] = '\0';
        printf("\n$%s\t%s$\t%s", stk, a, ac);
        i -= 2;
    }
}


for (z = 0; z < c; z++) {
    if (stk[z] == '(' && stk[z + 1] == 'E' && stk[z + 2] == ')') {
        stk[z] = 'E';
        stk[z + 1] = '\0';
        stk[z + 2] = '\0';
        printf("\n$%s\t%s$\t%s", stk, a, ac);
        i -= 2;
        }
    }
}
```



```
GRAMMAR is
E -> E+E
E -> E*E
E -> (E)
E -> id
Enter input string:
id+id*id
Stack            Input            Action

$id       +id*id$       SHIFT-> id
$E        +id*id$       REDUCE TO E
$E+        id*id$       SHIFT-> symbol
$E+id        *id$       SHIFT-> id
$E+E         *id$       REDUCE TO E
$E           *id$       REDUCE TO E
$E*           id$       SHIFT-> symbol
$E*id           $       SHIFT-> id
$E*E            $       REDUCE TO E
$E              $       REDUCE TO Eapsit@apsit-HP-ProDesk-400-G7-Microtower-PC:~/Desktop/anishspcc$
```

Conclusion: