

EXPERIMENT-20

AIM:

Write a C program to compute TRAILING() – operator precedence parser for the given grammar

PROGRAM:

The screenshot shows the Code::Blocks IDE interface. The top menu bar includes File, Edit, Search, View, Project, Execute, Tools, AStyle, Window, and Help. The toolbar contains various icons for file operations. The main window shows the code editor with the file EXP_4.cpp open. The code implements a function findTrailing to calculate trailing symbols for a grammar. The code uses character arrays grammar and trailing, and an integer n. It iterates through the grammar array, checks if the current character is a non-terminal, and then processes the trailing symbols. The code handles terminals like operators and identifiers. The bottom panel shows the compiler output, indicating 0 errors and 0 warnings, with the output filename being C:\Users\Haritha\OneDrive\Documents\EXP_4.exe.

```
#include <stdio.h>
#include <string.h>
char grammar[10][10];
char trailing[10][10];
int n;
void findTrailing(char nonTerminal, int index) {
    for (int i = 0; i < n; i++) {
        if (grammar[i][0] == nonTerminal) {
            int len = strlen(grammar[i]);
            char last = grammar[i][len - 1];
            char secondLast = grammar[i][len - 2];
            if ((last >= 'a' && last <= 'z') || last == ')' || last == '+' || last == '*' || last == '-' || last == '/') {
                trailing[index][strlen(trailing[index])] = last;
            }
            else if (last >= 'A' && last <= 'Z') {
                findTrailing(last, index);
            }
        }
    }
}
int main() {
    // Input grammar rules
    // E->E+T
    // E->T
    // TRAILING(E) = { + }
    // TRAILING(E) = { + }
    // -----
    // Process exited after 27.59 seconds with return value 0
    // Press any key to continue . . .
}
```

OUTPUT:

The screenshot shows a terminal window with a black background and white text. The user enters the number of grammar rules (2) and the grammar rules themselves (E->E+T and E->T). The program then outputs the trailing symbols for each rule, which are both '{ + }'. Finally, it prints a dashed line, the message 'Process exited after 27.59 seconds with return value 0', and a prompt to press any key to continue.

```
C:\Users\Haritha\OneDrive\Documents> Enter number of grammar rules: 2
Enter grammar rules (Format: A->B):
E->E+T
E->T

TRAILING() for given grammar:
TRAILING(E) = { + }
TRAILING(E) = { + }

-----
Process exited after 27.59 seconds with return value 0
Press any key to continue . . .
```