Department of Computer Science and Engineering

S.G.Shivanirudh , 185001146, Semester VI

26 February 2021

UCS1602 - Compiler Design

Exercise 4: Recursive Descent Parser using C

Objective:

Write a program in C to construct Recursive Descent Parser for the following grammar which is for arithmetic expression involving + and *. Check the Grammar for left recursion and convert into suitable for this parser. Write recursive functions for every non-terminal. Call the function for start symbol of the Grammar in main(). Extend this parser to include division, subtraction and parenthesis operators

Code:

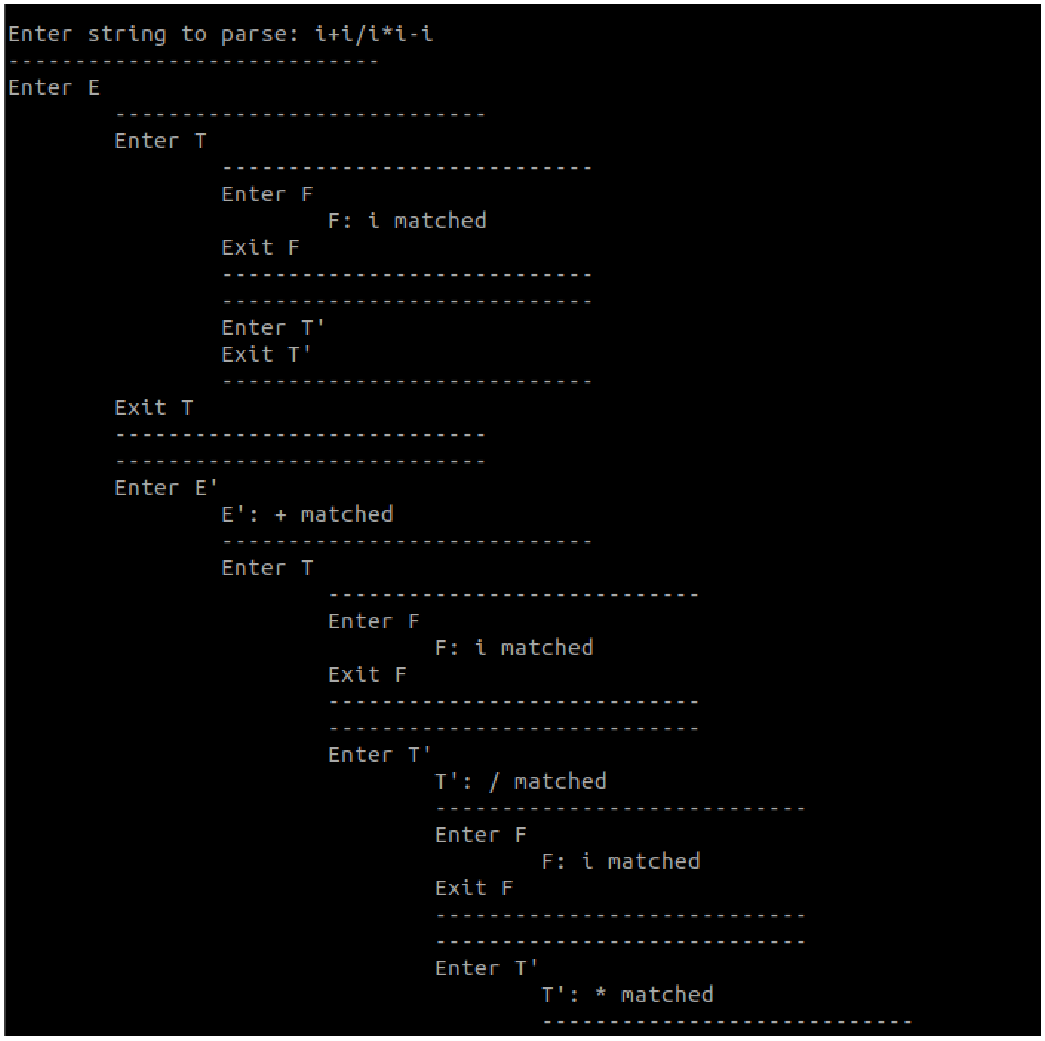
```c
#include<stdio.h>
#include<string.h>
#include<stdlib.h>

```

```c
void tab(int val){
    while(val--)
        printf("\t");
}
int F(char *, int *, int);
int Tprime(char *, int *, int);
int T(char *, int *, int);
int Eprime(char *, int *, int);
int E(char *, int *, int);

int main(){

    char *str = (char*)calloc(100, sizeof(char));
    printf("\nEnter string to parse: ");
    scanf(" %s", str);
    strcat(str, "$");
    int look_ahead = 0;

    printf("----------------------------\n");
    printf("Enter E\n");
    E(str, &look_ahead, 1);
    printf("Exit E\n");
    printf("----------------------------\n");

    if (str[look_ahead] == '$')
        printf("\nSuccess");
    else
        printf("\nFailure: %c at position %d not expected. \n
   ", str[look_ahead], look_ahead);
}

int F(char *str, int *look_ahead, int level)
{
    if (str[*look_ahead] == 'i')
    {
        tab(level);
        printf("F: i matched\n");
        (*look_ahead)++;
    }
    else if (str[*look_ahead] == '(')
    {
        tab(level);
        printf("F: ( matched\n");
        (*look_ahead)++;
        E(str, look_ahead, level + 1);
        if (str[*look_ahead] == ')')
        {
            tab(level);
            printf("F: ) matched\n");
            (*look_ahead)++;
        }
    }
}

int Tprime(char *str, int *look_ahead, int level){
    if(str[*look_ahead] == '*'){
        tab(level);
        printf("T': * matched\n");
        (*look_ahead)++;

        tab(level);
        printf("----------------------------\n");
        tab(level);
        printf("Enter F\n");
        F(str, look_ahead, level + 1);
        tab(level);
        printf("Exit F\n");
        tab(level);
        printf("----------------------------\n");

        tab(level);
        printf("----------------------------\n");
        tab(level);
        printf("Enter T'\n");
        Tprime(str, look_ahead, level + 1);
        tab(level);
        printf("Exit T'\n");
        tab(level);
        printf("----------------------------\n");

    }
    else if(str[*look_ahead] == '/'){
        tab(level);
        printf("T': / matched\n");
        (*look_ahead)++;

        tab(level);
        printf("----------------------------\n");
        tab(level);
        printf("Enter F\n");
        F(str, look_ahead, level + 1);
        tab(level);
        printf("Exit F\n");
        tab(level);
        printf("----------------------------\n");

        tab(level);
        printf("----------------------------\n");
        tab(level);
        printf("Enter T'\n");
        Tprime(str, look_ahead, level + 1);
        tab(level);
        printf("Exit T'\n");
        tab(level);
        printf("----------------------------\n");

    }
}

int T(char *str, int *look_ahead, int level){
    tab(level);
    printf("----------------------------\n");
    tab(level);
    printf("Enter F\n");
    F(str, look_ahead, level + 1);
    tab(level);
    printf("Exit F\n");
    tab(level);
    printf("----------------------------\n");

    tab(level);
    printf("----------------------------\n");
    tab(level);
    printf("Enter T'\n");
    Tprime(str, look_ahead, level + 1);
    tab(level);
    printf("Exit T'\n");
    tab(level);
    printf("----------------------------\n");
}
int Eprime(char *str, int *look_ahead, int level){

    if(str[*look_ahead] == '+'){
        tab(level);
        printf("E': + matched\n");
```

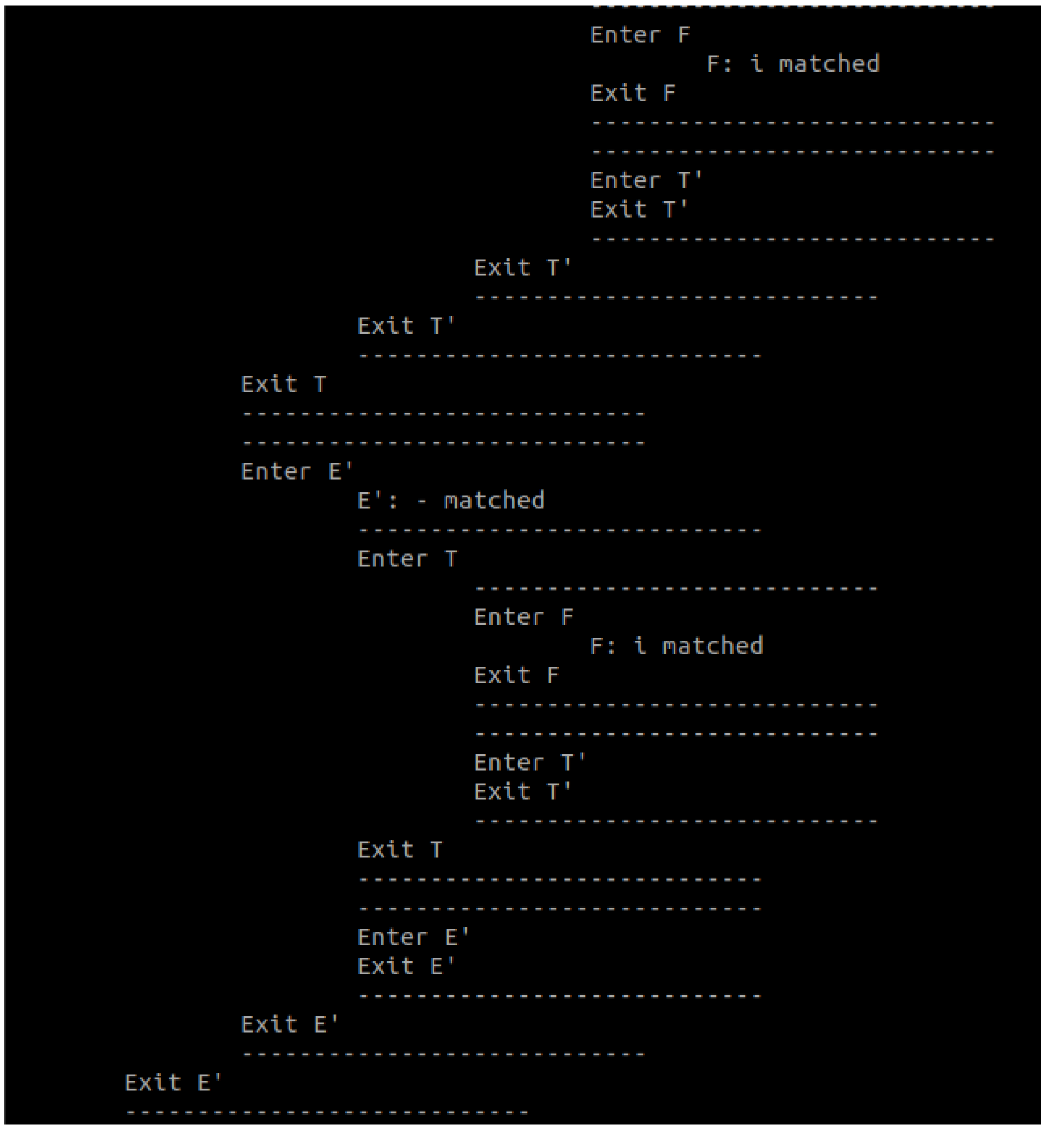
```c
        (*look_ahead)++;

        tab(level);
        printf("----------------------------\n");
        tab(level);
        printf("Enter T\n");
        T(str, look_ahead, level + 1);
        tab(level);
        printf("Exit T\n");
        tab(level);
        printf("----------------------------\n");

        tab(level);
        printf("----------------------------\n");
        tab(level);
        printf("Enter E'\n");
        Eprime(str, look_ahead, level + 1);
        tab(level);
        printf("Exit E'\n");
        tab(level);
        printf("----------------------------\n");
    }
    else if(str[*look_ahead] == '-'){
        tab(level);
        printf("E': - matched\n");
        (*look_ahead)++;

        tab(level);
        printf("----------------------------\n");
        tab(level);
        printf("Enter T\n");
        T(str, look_ahead, level + 1);
        tab(level);
        printf("Exit T\n");
        tab(level);
        printf("----------------------------\n");

        tab(level);
        printf("----------------------------\n");
        tab(level);
        printf("Enter E'\n");
        Eprime(str, look_ahead, level + 1);
        tab(level);
        printf("Exit E'\n");
        tab(level);
        printf("----------------------------\n");
    }
}

int E(char *str, int *look_ahead, int level){
    tab(level);
    printf("----------------------------\n");
    tab(level);
    printf("Enter T\n");
    T(str, look_ahead, level+1);
    tab(level);
    printf("Exit T\n");
    tab(level);
    printf("----------------------------\n");

    tab(level);
    printf("----------------------------\n");
    tab(level);
    printf("Enter E'\n");
    Eprime(str, look_ahead, level+1);
    tab(level);
    printf("Exit E'\n");
    tab(level);
    printf("----------------------------\n");
}
```

Output:

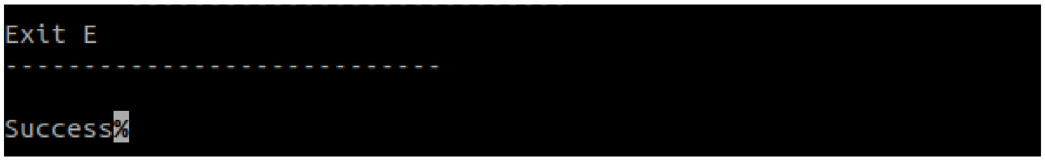
Success scenario:



```
Enter string to parse: i+i/i*i-i
----------------------------
Enter E
        ----------------------------
        Enter T
                ----------------------------
                Enter F
                        F: i matched
                Exit F
                ----------------------------
                ----------------------------
                Enter T'
                Exit T'
                ----------------------------
        Exit T
        ----------------------------
        ----------------------------
        Enter E'
                E': + matched
                ----------------------------
                Enter T
                        ----------------------------
                        Enter F
                                F: i matched
                        Exit F
                        ----------------------------
                        ----------------------------
                        Enter T'
                                T': / matched
                                ----------------------------
                                Enter F
                                        F: i matched
                                Exit F
                                ----------------------------
                                ----------------------------
                                Enter T'
                                        T': * matched
                                        ----------------------------
```

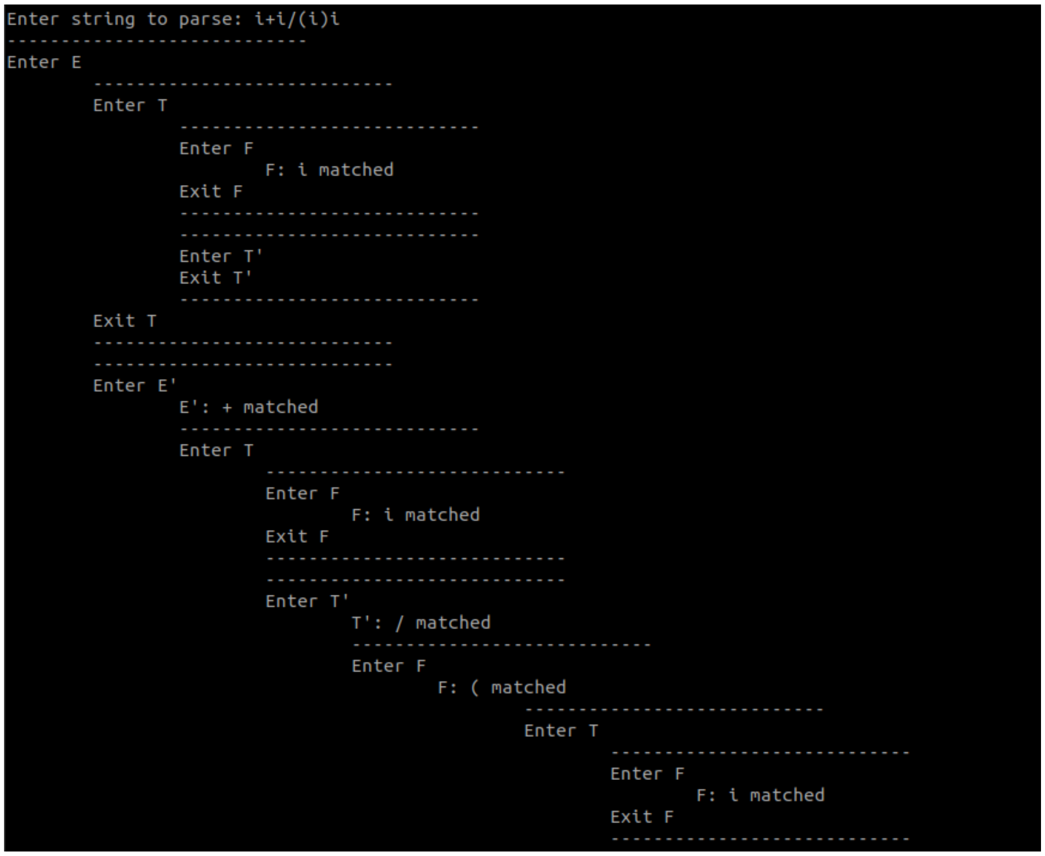


```
                                        ----------------------------
                                        Enter F
                                                F: i matched
                                        Exit F
                                        ----------------------------
                                        ----------------------------
                                        Enter T'
                                        Exit T'
                                        ----------------------------
                                Exit T'
                                ----------------------------
                        Exit T'
                        ----------------------------
                Exit T
                ----------------------------
                ----------------------------
                Enter E'
                        E': - matched
                        ----------------------------
                        Enter T
                                ----------------------------
                                Enter F
                                        F: i matched
                                Exit F
                                ----------------------------
                                ----------------------------
                                Enter T'
                                Exit T'
                                ----------------------------
                        Exit T
                        ----------------------------
                        ----------------------------
                        Enter E'
                        Exit E'
                        ----------------------------
                Exit E'
                ----------------------------
        Exit E'
        ----------------------------
```

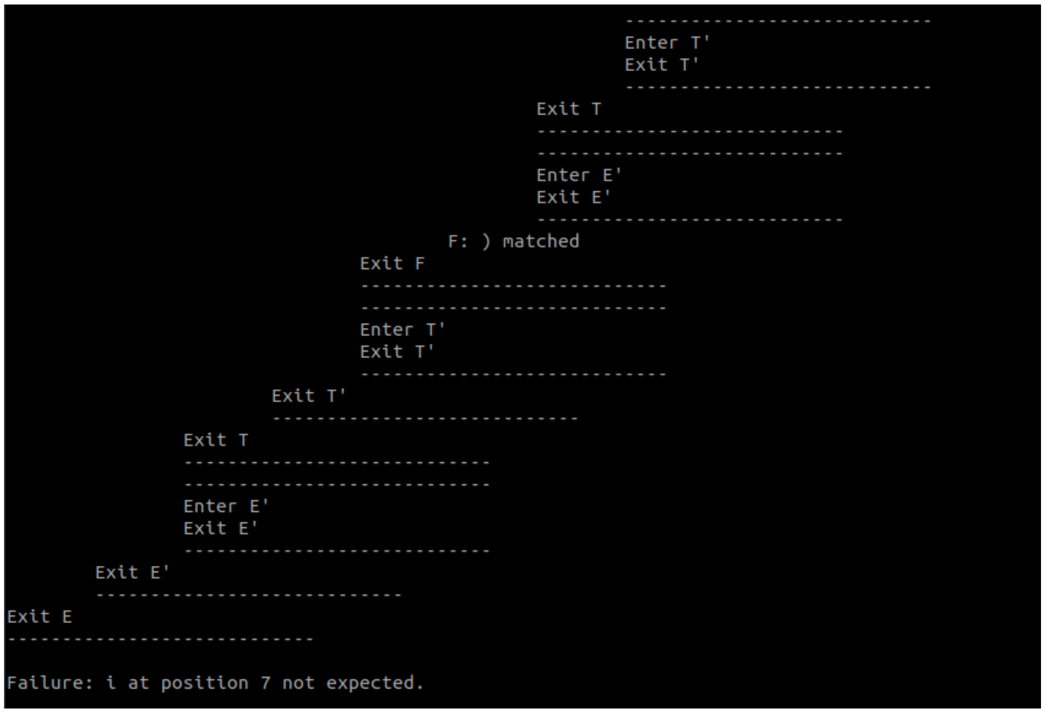


```
Exit E
------------------------------

Success%
```

Failure scenario:



```
Enter string to parse: i+i/(i)i
----------------------------
Enter E
        ----------------------------
        Enter T
                ----------------------------
                Enter F
                        F: i matched
                Exit F
                ----------------------------
                ----------------------------
                Enter T'
                Exit T'
                ----------------------------
        Exit T
        ----------------------------
        ----------------------------
        Enter E'
                E': + matched
                ----------------------------
                Enter T
                        ----------------------------
                        Enter F
                                F: i matched
                        Exit F
                        ----------------------------
                        ----------------------------
                        Enter T'
                                T': / matched
                                ----------------------------
                                Enter F
                                        F: ( matched
                                                ----------------------------
                                                Enter T
                                                        ----------------------------
                                                        Enter F
                                                                F: i matched
                                                        Exit F
                                                        ----------------------------
```



```
                                                        ----------------------------
                                                        Enter T'
                                                        Exit T'
                                                        ----------------------------
                                                Exit T
                                                ----------------------------
                                                ----------------------------
                                                Enter E'
                                                Exit E'
                                                ----------------------------
                                        F: ) matched
                                Exit F
                                ----------------------------
                                ----------------------------
                                Enter T'
                                Exit T'
                                ----------------------------
                        Exit T'
                        ----------------------------
                Exit T
                ----------------------------
                ----------------------------
                Enter E'
                Exit E'
                ----------------------------
        Exit E'
        ----------------------------
Exit E
----------------------------

Failure: i at position 7 not expected.
```

Learning Outcomes:

– Understood the basic working of recursive descent parser.

– Learnt how to use left-recursion-eliminated grammar to write code for recursive descent parser.