SRM UNIVERSITY AP
Andhra Pradesh

Department of Computer Science and Engineering
Compiler Design Lab (CS 306)

Week 6: Implementation of Recursive Descent Parser

Week 6 Programs

1. Implement Recursive Descent Parser for the Expression Grammar given below.

   E → TE'
   E'→ +TE' | ϵ
   T → FT'
   T'→ *FT' | ϵ
   F → (E) | i

2. Construct Recursive Descent Parser for the grammar
   G = ({S, L}, {(,), a, ,}, {S → (L) | a ; L→ L, S | S}, S) and verify the acceptability of the following strings:
   i. (a,(a,a))
   ii. (a,((a,a),(a,a)))

   You can manually eliminate Left Recursion if any in the grammar.

Instructions:

- Explanation and code of program 1 is given below.
- You are required to implement the same for program 2
- Upload both programs into your Github accounts under the folder **Week6-Lab-exercise**

Description:

Program:

C implementation of Recursive Descent Parser for the Expression Grammar is given below.

```
#include<stdio.h>
#include<string.h>
int E(),Edash(),T(),Tdash(),F();
char *ip;
char string[50];
int main()
{
   printf("Enter the string\n");
   scanf("%s",string);
   ip=string;
   printf("\n\nInput\tAction\n--------------------------------\n");
```

```
    if(E() && ip==’\0’){
        printf("\n--------------------------------\n");
        printf("\n String is successfully parsed\n");
    }
    else{
        printf("\n--------------------------------\n");
        printf("Error in parsing String\n");
    }
}
int E()
{
   printf("%s\tE->TE' \n",ip);
    if(T())
    {
        if(Edash())
        {
            return 1;
        }
        else
            return 0;
    }
    else
        return 0;
}
int Edash()
{
    if(*ip=='+')
    {
        printf("%s\tE'->+TE' \n",ip);
        ip++;
        if(T())
        {
            if(Edash())
            {
                return 1;
            }
            else
                return 0;
        }
        else
            return 0;
    }
    else
    {
        printf("%s\tE'->^ \n",ip);
        return 1;
    }
}
int T()
{
    printf("%s\tT->FT' \n",ip);
    if(F())
    {
```

```
        if(Tdash())
        {
            return 1;
        }
        else
            return 0;
    }
    else
        return 0;
}
int Tdash()
{
    if(*ip=='*')
    {
        printf("%s\tT'->*FT' \n",ip);
        ip++;
        if(F())
        {
            if(Tdash())
            {
                return 1;
            }
            else
                return 0;
        }
        else
            return 0;
    }
    else
    {
      printf("%s\tT'->^ \n",ip);
      return 1;
    }
}
int F()
{
    if(*ip=='(')
    {
        printf("%s\tF->(E) \n",ip);
        ip++;
        if(E())
        {
            if(*ip==')')
            {
                ip++;
              return 0;
            }
            else
                return 0;
        }
        else
            return 0;
    }
```

```
    else if(*ip=='i')
    {
       ip++;
         printf("%s\tF->id \n",ip);
       return 1;
    }
    else
       return 0;
}
```

Test cases:

i+i*i	String is successfully parsed
i+i	String is successfully parsed
i*i	String is successfully parsed
i*i+i*i+i	String is successfully parsed
i+*+i	Error in parsing String
i+i*	Error in parsing String