Exercise 4
Recursive Decent Parser

Praveen Kumar R
312217104114

February 1, 2020

Aim

To Write a C program to construct a recursive decent parser for the grammer.

$$
\begin{aligned}
E &\rightarrow TE' \\
E' &\rightarrow +TE' \mid \epsilon \\
T &\rightarrow FT' \\
T' &\rightarrow *FT' \mid \epsilon \\
F &\rightarrow id \mid (E)
\end{aligned}
$$

C Program

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

void E(char[],int*);
void T(char[],int*);
void Tprime(char[],int*);
void Eprime(char[],int*);
void F(char[],int*);


void E(char a[],int* n){
  printf("In E\n");
  T(a,n);
  Eprime(a,n);
}

void Eprime(char a[], int *n){
  printf("In E\'\n");
  if(a[*n]=='+'){
    printf("encountered + symbol : pointer advanced\n");
    (*n)++;
      T(a,n);
      Eprime(a,n);
  }
  else{
    return;
  }

}

void T(char a[],int *n){
  printf("In T\n");
  F(a,n);
  Tprime(a,n);
}

void Tprime(char a[],int *n){
  printf("In T\'\n");
  if(a[*n]=='*'){
    printf("encountered * symbol : pointer advanced\n");
    (*n)++;

      F(a,n);
      Tprime(a,n);
  }
  else{
    return;
  }

}

void F(char a[],int *n){
  printf("In F\n");

  if(a[*n]=='('){
    printf("encountered ( symbol : pointer advanced \n");
    (*n)++;
    printf("\n");
    E(a,n);
    if(a[*n]==')'){
      printf("encountered ) symbol : pointer advanced\n");
      (*n)++;
      if(*n==strlen(a)){
        printf("Parsing Complete string accepted\n");
        exit(0);
      }
    }

  }
  else if(a[*n]=='i' && a[*n+1]=='d'){
    printf("encountered id symbol : pointer advanced\n");
    (*n)+=2;
    if(*n==strlen(a)){
      printf("Parsing Complete string accepted\n");
      exit(0);
    }
  }
  else{
    printf("Parsing error: String not accepted\n");
    exit(0);
  }
}
int main(){
  char a[100];
  int n=0;
  printf("Enter the String to be parsed: ");
  scanf("%s",a);
  E(a,&n);
}
```

Sample Input & Output 1

```
Enter the String to be parsed: (id+id*id)
In E
In T
In F
encountered ( symbol : pointer advanced
In E
In T
In F
encountered id symbol : pointer advanced
In T'
In E'
encountered + symbol : pointer advanced
In T
In F
encountered id symbol : pointer advanced
In T'
encountered * symbol : pointer advanced
In F
encountered id symbol : pointer advanced
In T'
In E'
encountered ) symbol : pointer advanced
Parsing Complete string accepted
```

Sample Input & Output 2

```
Enter the String to be parsed: (idid*id
In E
In T
In F
encountered ( symbol : pointer advanced
In E
In T
In F
encountered id symbol : pointer advanced
In T'
In E'
In T'
In E'
Parser Error : String not accepted
```