Compiler Design Lab

Week 4 & 5

AP20110010143

1. 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.

Code:

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


char input[10];
int i, error;
void S();
void L();
void Lprime();


int main(void){
    i = 0;
    error = 0;
    printf("Enter an arithmetic expression: ");
    gets(input);
    S();
    if(strlen(input) == i && error == 0)
```

```
        printf("\nAccepted...!!\n");
    else
        printf("\nRejected..!!\n");
}
void L(){
    S();
    Lprime();
}
void Lprime(){
    if(input[i] == ','){
        i++;
        S();
        Lprime();
    }
}
void S(){
    if(input[i] == 'a'){
        i++;
    }
    else if(input[i] == '('){
        i++;
        L();
        if(input[i] == ')')
            i++;
        else
            error = 1;
    }
    else{
        error = 1;
```

```
    }
}
```

Output:

```
Enter an arithmetic expression: (a,(a,a))

Accepted...!!
```

```
Enter an arithmetic expression: (a,((a,a),(a,a)))

Rejected..!!
```

2. Implement the computing First and Follow usingCforthefollowing

Code:

```c
#include<stdio.h>
#include<math.h>
#include<string.h>
#include<ctype.h>
#include<stdlib.h>
int n,m=0,p,i=0,j=0;
char a[10][10],f[10];
void follow(char c);
void first(char c);


int main(){
    int i,z;
    char c,ch;
    printf("Enter the no of productions:\n");
    scanf("%d",&n);
    printf("Enter the productions:\n");
    for(i=0;i<n;i++)
```

```
    scanf("%s%c",a[i],&ch);
    do{
        m=0;
        printf("Enter the elements whose first & follow is to be found:");
        scanf("%c",&c);
        first(c);
        printf("First(%c)={",c);
        for(i=0;i<m;i++)
            printf("%c",f[i]);
            printf("}\n");
            strcpy(f," ");
        m=0;
        follow(c);
        printf("Follow(%c)={",c);
        for(i=0;i<m;i++)
            printf("%c",f[i]);
            printf("}\n");
        printf("Continue(0/1)?");
        scanf("%d%c",&z,&ch);
    }while(z==1);
return(0);
}
void first(char c)
{
    int k;
    if(!isupper(c))
        f[m++]=c;
    for(k=0;k<n;k++)
    {
```

```
        if(a[k][0]==c)
        {
            if(islower(a[k][2]))
                f[m++]=a[k][2];
            else
                first(a[k][2]);
        }
    }
}
void follow(char c)
{
    if(a[0][0]==c)
        f[m++]='$';
    for(i=0;i<n;i++)
    {
        for(j=2;j<strlen(a[i]);j++)
        {
            if(a[i][j]==c)
            {
                if(a[i][j+1]!='\0')
                    first(a[i][j+1]);
                if(a[i][j+1]=='\0' && c!=a[i][0])
                    follow(a[i][0]);
            }
        }
    }
}
```

Output:

```
Enter the no of productions:
6
Enter the productions:
E=TA
A=+TA
T=VB
B=*VB
V=i
V=(E)
Enter the elements whose first & follow is to be found:E
First(E)={i(}
Follow(E)={$)}
Continue(0/1)?1
Enter the elements whose first & follow is to be found:A
First(A)={+}
Follow(A)={$)}
Continue(0/1)?1
Enter the elements whose first & follow is to be found:T
First(T)={i(}
Follow(T)={++}
Continue(0/1)?1
Enter the elements whose first & follow is to be found:V
First(V)={i(}
Follow(V)={**}
Continue(0/1)?1
Enter the elements whose first & follow is to be found:B
First(B)={*}
Follow(B)={++}
Continue(0/1)?
```

```
Enter the no of productions:
8
Enter the productions:
S=ABCDE
A=a
A=**
B=b
B=**
C=c
D=d
D=**
Enter the elements whose first & follow is to be found:S
First(S)={a*}
Follow(S)={$}
Continue(0/1)?1
Enter the elements whose first & follow is to be found:A
First(A)={a*}
Follow(A)={b*}
Continue(0/1)?1
Enter the elements whose first & follow is to be found:B
First(B)={b*}
Follow(B)={c}
Continue(0/1)?1
Enter the elements whose first & follow is to be found:C
First(C)={c}
Follow(C)={d*}
Continue(0/1)?1
Enter the elements whose first & follow is to be found:D
First(D)={d*}
Follow(D)={}
Continue(0/1)?0
PS C:\Users\Charan>
```