

## **SHIFT REDUCE PARSER**

**AIM :** To write a program in C/C++ to implement the shift reduce parser.

### **PROCEDURE:**

1. Shift Reduce Parser requires two Data Structures
  - ☐ Input Buffer
  - ☐ Stack
2. It uses a stack and an input buffer.
3. Insert \$ at the bottom of the stack and the right end of the input string in Input Buffer.
4. Shift – Parser shifts zero or more input symbols onto the stack until the handle is on top of the stack.
5. Reduce – Parser reduces or replaces the handle on top of the stack to the left side of production, i.e., R.H.S. of production is popped, and L.H.S is pushed.
6. Accept – Step 3 and Step 4 will be repeated until it has detected an error or until the stack includes start symbol (S) and input Buffer is empty, i.e., it contains \$.
7. Handle – Each replacement of the Right side of production by the left side in the process above is known as "Reduction" and each replacement is called "Handle."
8. Error: This is the situation in which the parser can neither perform shift action nor reduce action and not even accept action.

### **PROGRAM:**

```
#include<iostream>
#include<string.h>
using namespace std;
struct prodn
{
    char p1[10];
    char p2[10];
};
int main()
{
    char input[20],stack[50],temp[50],ch[2],*t1,*t2,*t;
    int i,j,s1,s2,s,count=0;
    struct prodn p[10];
    FILE *fp=fopen("sr_input.txt","r");
    stack[0]='\0';
    cout<<"Enter the Input String:\n";
    cin>>input;
    while(!feof(fp))
    {
```

```

        fscanf(fp,"%s\n",temp);
        t1= strtok(temp,"->");
        t2= strtok(NULL,"->");
        strcpy(p[count].p1,t1);
        strcpy(p[count].p2,t2);
        count++;
    }
    i=0;
    while(1)
    {
        if(i<strlen(input))
        {
            ch[0]=input[i];
            ch[1]='\0';
            i++;
            strcat(stack,ch);
            cout<<"\n"<<stack;
        }
        for(j=0;j<count;j++)
        {
            t= strstr(stack,p[j].p2);
            if(t!=NULL)
            {
                s1= strlen(stack);
                s2= strlen(t);
                s=s1-s2;
                stack[s]='\0';
                strcat(stack,p[j].p1);
                cout<<"\n"<<stack;
                j=-1;
            }
        }
        if(strcmp(stack,"E")==0&&i==strlen(input))
        {
            cout<<"\n\nAccepted";
            break;
        }
        if(i==strlen(input))
        {
            cout<<"\n\nNot Accepted";
            break;
        }
    }
    return 0;
}

```

## **INPUT:**

**"sr\_input.txt"**

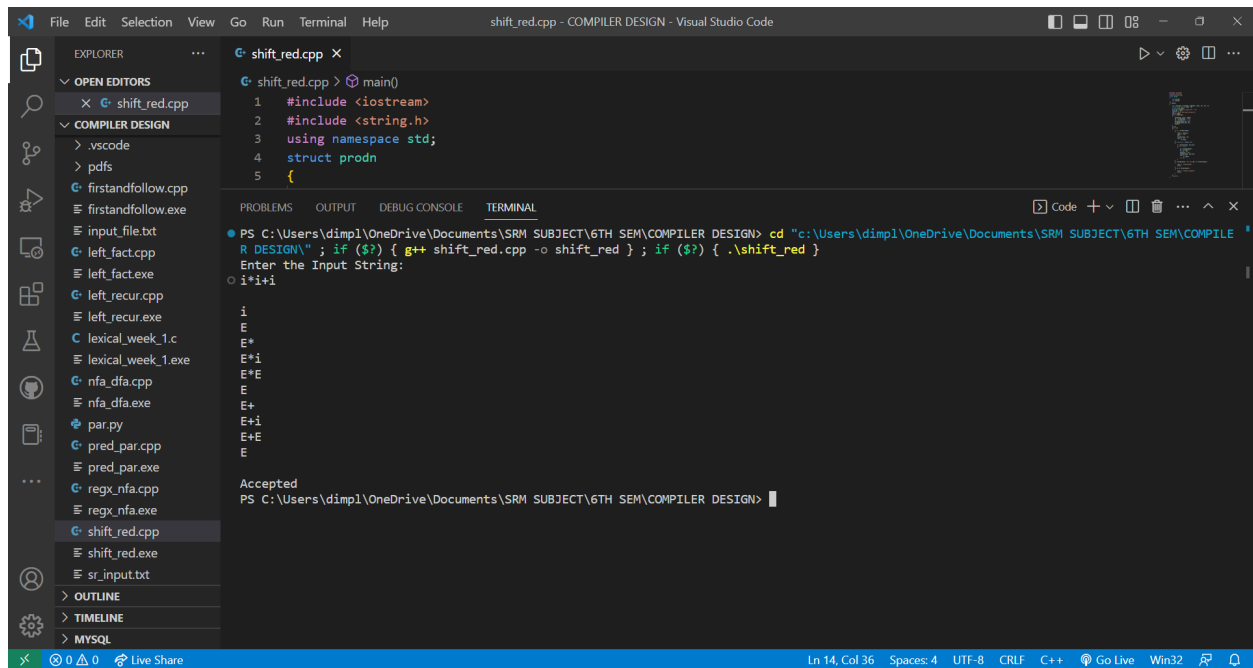
E->E+E

E->E\*E

E->

Enter the input string: i\*i+i

## **OUTPUT:**



```
File Edit Selection View Go Run Terminal Help shift_red.cpp - COMPILER DESIGN - Visual Studio Code

EXPLORER
  OPEN EDITORS
    X shift_red.cpp
  COMPILER DESIGN
    .vscode
    pdfs
    firstandfollow.cpp
    firstandfollow.exe
    input_file.txt
    left_fact.cpp
    left_fact.exe
    left_recur.cpp
    left_recur.exe
    lexical_week_1.c
    lexical_week_1.exe
    nfa_dfa.cpp
    nfa_dfa.exe
    par.py
    pred_par.cpp
    pred_par.exe
    regx_nfa.cpp
    regx_nfa.exe
    shift_red.cpp
    shift_red.exe
    sr_input.txt
    OUTLINE
    TIMELINE
    MYSQL

shift_red.cpp X
  1 #include <iostream>
  2 #include <string.h>
  3 using namespace std;
  4 struct prodn
  5 {

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
  PS C:\Users\dimpl\OneDrive\Documents\SRM SUBJECT\6TH SEM\COMPILER DESIGN> cd "c:\Users\dimpl\OneDrive\Documents\SRM SUBJECT\6TH SEM\COMPILER DESIGN\" ; if ($?) { g++ shift_red.cpp -o shift_red } ; if ($?) { .\shift_red }
  Enter the Input String:
  i*i+i
  i
  E
  E+
  E*i
  E+E
  E=
  E+
  E+i
  E+E
  E
  Accepted
  PS C:\Users\dimpl\OneDrive\Documents\SRM SUBJECT\6TH SEM\COMPILER DESIGN>
```

## **RESULT:**

Thus, we have successfully implemented the shift reduce parser.