```
/*********************************************************

Name  : Shubham S. Tembhurkar
PRN   : 1641060
Class : L.Y.B.TECH.
Batch : B1
Sub   : CCL
Aim   : Implementation of shift reduce parsing algorithm.

*********************************************************/

#include<stdio.h>      //including header files
#include<stdlib.h>
#include<string.h>
char ip_sym[15],stack[15]; //declaring global variables
int ip_ptr=0,st_ptr=0,len,i;
char temp[2],temp2[2];
char act[15];
void check();    //declaring function
void main()      //defining main
{
        printf("\n\t\t SHIFT REDUCE PARSER\n");
        printf("\n GRAMMER\n");                   //defining grammar
        printf("\n E->E+E\n E->E/E");
        printf("\n E->E*E\n E->a/b");
        printf("\n Enter the input symbol:\t");
        gets(ip_sym);                    //take input string
        printf("\n\t Stack implementation table");
        printf("\n Stack\t\t Input symbol\t\t Action");
        //construct parse table
        printf("\n_____\t\t _____\t\t _____\n");
        printf("\n $\t\t%s$\t\t\t--",ip_sym);
        strcpy(act,"shift ");
        temp[0]=ip_sym[ip_ptr];
        temp[1]='\0';
        strcat(act,temp);
        len=strlen(ip_sym);
        for(i=0;i<=len-1;i++) //using for loop
        {
                stack[st_ptr]=ip_sym[ip_ptr];    //inserting in stack
                stack[st_ptr+1]='\0';
                ip_sym[ip_ptr]=' ';
                ip_ptr++;
                printf("\n $%s\t\t%s$\t\t\t%s",stack,ip_sym,act);
                strcpy(act,"shift ");
                temp[0]=ip_sym[ip_ptr];
                temp[1]='\0';
                strcat(act,temp);
                check();   //checking with grammar
                st_ptr++;
        }
        st_ptr++;
```

```c
        check();
}

void check()            //function definition
{
        int flag=0;
        temp2[0]=stack[st_ptr];
        temp2[1]='\0';
        if((!strcmp(temp2,"a"))||(!strcmp(temp2,"b")))
        {
                stack[st_ptr]='E';
                if(!strcmp(temp2,"a"))            //checking for third
production
                        printf("\n $%s\t\t%s$\t\t\tE->a",stack, ip_sym);
                else
                         printf("\n $%s\t\t%s$\t\t\tE->b\n",stack,ip_sym);
                         flag=1;
        }
        if((!strcmp(temp2,"+"))||(strcmp(temp2,"*"))||(!strcmp(temp2,"/")
))
        {
                 flag=1;
        }
        if((!strcmp(stack,"E+E"))||(!strcmp(stack,"E\E"))||(!strcmp(stack
,"E*E")))
        {
                strcpy(stack,"E");
                st_ptr=0;
                if(!strcmp(stack,"E+E"))    //using if condition
                        printf("\n $%s\t\t%s$\t\t\tE->E+E",stack,ip_sym);
                else if(!strcmp(stack,"E\E"))
                        printf("\n $%s\t\t %s$\t\t\tE->E\E",stack,ip_sym);
                else
                        printf("\n $%s\t\t%s$\t\t\tE->E*E",stack,ip_sym);
                flag=1;
        }
        if(!strcmp(stack,"E")&&ip_ptr==len)
        {
                printf("\n $%s\t\t%s$\t\t\tACCEPT\n",stack,ip_sym);
                exit(0);
        }
        if(flag==0)
        {
                printf("\n%s\t\t\t%s\t\t reject\n",stack,ip_sym);
                exit(0);
        }
        return;
}       //end check function


/*********************OUTPUT*****************************
```

```
shubh@ubuntu:~/CCL$ ./a.out

              SHIFT REDUCE PARSER

 GRAMMER

 E->E+E
 E->E/E
 E->E*E
 E->a/b
 Enter the input symbol:    a+b

      Stack implementation table
 Stack              Input symbol          Action
 _____             _____          _____

   $          a+b$                 --
   $a          +b$                 shift a
   $E          +b$                 E->a
   $E+          b$                 shift +
   $E+b          $                 shift b
   $E+E          $                 E->b

   $E            $                 E->E*E
   $E            $                 ACCEPT
shubh@ubuntu:~/CCL$ ./a.out

              SHIFT REDUCE PARSER

 GRAMMER

 E->E+E
 E->E/E
 E->E*E
 E->a/b
 Enter the input symbol:    a/b

      Stack implementation table
 Stack              Input symbol          Action
 _____             _____          _____

   $          a/b$                 --
   $a          /b$                 shift a
   $E          /b$                 E->a
   $E/          b$                 shift /
   $E/b          $                 shift b
   $E/E          $                 E->b
shubh@ubuntu:~/CCL$

 ************************************************************/
```