```
/* ///*
 * ASSIGNMENT: OPERATOR PRECEDENCE WITH RECOVERY
 * AUTHORS: BIJAY KHATRI & DEEPAK VERMA
 * ROLL: 12/CS/45 AND 12/CS/46
 *//// */
#include<stdio.h>
#include<ctype.h>
#include<string.h>

struct stru1
{
char non_ter[1],pro[25];
}cfg[25];
int n,st=-1,j,i,t=-1,m,in=0;
int v,c,p=1,s=0;
char str[20],stack[20],ch,tmp[10],input[20],symbol[2];
int main()
{
char temp[20];
char prec[6][6]={{' ','-','+','*','/','i'},{'-','=','<','<','<','<'},{'+','>','=','<','<','<'},{'*','>','>','=','<','<'},{'/','>','>','>','=','<'},{'i','>','>','>','>','='}};
printf("\n\n");
for(i=0;i<6;i++)
{
   for(j=0;j<6;j++)
      printf("  %c |",prec[i][j]);
   printf("\n---------------------------------\n");
}
printf("Enter the number of productions:\n\r");
scanf("%d",&n);
printf("\n\r");
printf("Enter the productions:\n\r");
for(i=0;i<n;i++)
{
scanf("%s",cfg[i].non_ter);
//printf("\n\r");
scanf("%s",symbol);
scanf("%s",cfg[i].pro);
printf("\n\r");
}
printf("Enter the input string:\n\r");
scanf("%s",str);
printf("\n\r");
int i,r;
printf("ACTION\t\tSTACK\t\tINPUT\n");
printf("-\t\t-\t\t%s$\n",str);
label1: stack[++st]=str[s++];
int low=0;
      for(j=s;j<strlen(str);j++)
      {
         input[low++]=str[j];
      }m++;
      input[low++]='$';
      input[low]='\0';
label3:     if(str[s-1]=='\0')
      {

         if(stack[st-1]==')')
         {
            printf("unbalanced right paranthesis \n Recovering error : removed right paranthesis\n");
            if(st==2)
            {
            stack[st-1]='\0';
            goto label3;
            }
            st--;
            printf("removed ')'\t%s\t%s\n",stack,input);
            goto label;
         }
         else if(st==1)
         printf("\n\n\tString accepted\n");
         else
         printf("\n\n\tNot accepted by above grammar\n");
         return 0;
      }
      printf("shift %c\t\t%s\t\t%s\n",stack[st],stack,input);
      if((stack[st]=='+'||stack[st]=='-'||stack[st]=='*'||stack[st]=='/')&&(input[0]=='+'||input[0]=='-'||input[0]=='*'||input[0]=='/'))
         printf("\n---MISSING OPERAND---\n");
```

```
    //   scanf("%d",&t);
label: for (i=0;i<=st;i++)
{
   int l=0;
   for (j=i;j<=st;j++)
   {
      temp[l++]=stack[j];
   }
   temp[l]='\0';
   for (r=0;r<n;r++){
   if(strcmp(temp,cfg[r].pro)==0)
   {
      int len = strlen(temp);
      while(len--)
      {
         stack[st--]=NULL;
      }
      stack[++st]=cfg[r].non_ter[0];
      printf ("Reduced %c->%s\t%s\t\t%s\n",cfg[r].non_ter[0],cfg[r].pro,stack,input);

      if((stack[st-1]=='+'||stack[st-1]=='-'||stack[st-1]=='*')&&input[0]=='/')
         goto label1;
      else if((stack[st-1]=='+'||stack[st-1]=='-')&&input[0]=='*')
         goto label1;
      else if((stack[st-1]=='-')&&input[0]=='+')
         goto label1;

      if((isalnum(stack[st]))&&isalnum(stack[st-1]))
         printf("\n---MISSING OPERATOR---\n");
      goto label;
   }
   }
}
goto label1;
return 0;
}

/*
   | -| +| *| /| i|
-----------------------------------
 -| =| <| <| <| <|
-----------------------------------
 +| >| =| <| <| <|
-----------------------------------
 *| >| >| =| <| <|
-----------------------------------
 /| >| >| >| =| <|
-----------------------------------
 i| >| >| >| >| =|
-----------------------------------
Enter the number of productions:
4
Enter the productions:
E
->
a

E
->
E+E

E
->
E*E

E
->
(E)

Enter the input string:
(a+a*a)*a+a

ACTION       STACK      INPUT
-     -      (a+a*a)*a+a$
shift (       (      a+a*a)*a+a$
shift a       (a       +a*a)*a+a$
Reduced E->a  (E       +a*a)*a+a$
```

```
shift +       (E+     a*a)*a+a$
shift a       (E+a      *a)*a+a$
Reduced E->a  (E+E      *a)*a+a$
shift *       (E+E*       a)*a+a$
shift a       (E+E*a      )*a+a$
Reduced E->a  (E+E*E      )*a+a$
Reduced E->E*E (E+E       )*a+a$
Reduced E->E+E    (E      )*a+a$
shift )       (E)     *a+a$
Reduced E->(E)  E     *a+a$
shift *       E*      a+a$
shift a       E*a     +a$
Reduced E->a  E*E     +a$
Reduced E->E*E E      +a$
shift +       E+      a$
shift a       E+a       $
Reduced E->a  E+E       $
Reduced E->E+E    E     $

string accepted

*/
```