Program for implementing Shift Reduce Parsing using C

ALGORITHM:

1. Get the input expression and store it in the input buffer.
2. Read the data from the input buffer one at the time.
3. Using stack and push & pop operation shift and reduce symbols with respect to production rules available.
4. Continue the process till symbol shift and production rule reduce reaches the start symbol.
5. Display the Stack Implementation table with corresponding Stack actions with input symbols.

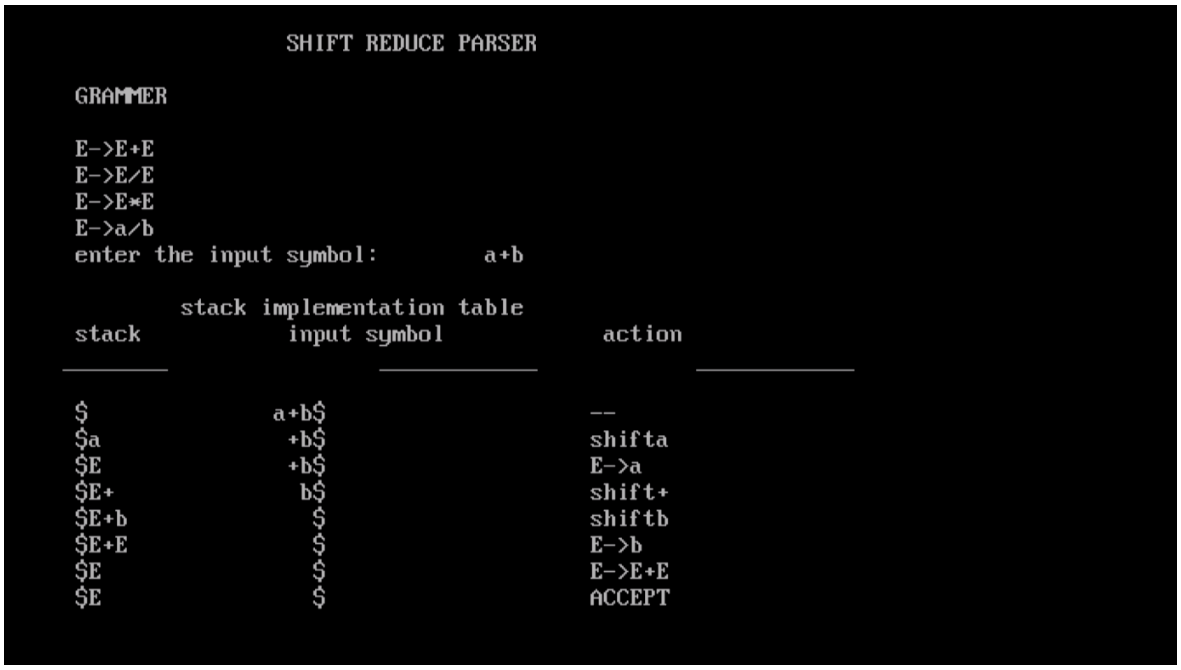
PROGRAM:

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

char ip_sym[15],stack[15];

int ip_ptr=0,st_ptr=0,len,i;

char temp[2],temp2[2];

char act[15];

void check();

void main()

{

clrscr();

printf("\n\t\t SHIFT REDUCE PARSER\n");

printf("\n GRAMMER\n");

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);
```

```
printf("\n\t stack implementation table");

printf("\n stack \t\t input symbol\t\t action");

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++)

{

stack[st_ptr]=ip_sym[ip_ptr];

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();

st_ptr++;

}

st_ptr++;

check();

}

void check()

{
```

```
int flag=0;

temp2[0]=stack[st_ptr];

temp2[1]='\0';

if((!strcmpi(temp2,"a"))||(!strcmpi(temp2,"b")))

{

stack[st_ptr]='E';

if(!strcmpi(temp2,"a"))

printf("\n $%s\t\t%s$\t\t\tE->a",stack,ip_sym);

else

printf("\n $%s\t\t%s$\t\t\tE->b",stack,ip_sym);

flag=1;

}

if((!strcmpi(temp2,"+"))||(strcmpi(temp2,"*"))||(!strcmpi(temp2,"/")
))

{

flag=1;

}

if((!strcmpi(stack,"E+E"))||(!strcmpi(stack,"E\E"))||(!strcmpi(stack
,"E*E")))

{

strcpy(stack,"E");

st_ptr=0;

if(!strcmpi(stack,"E+E"))

printf("\n $%s\t\t%s$\t\t\tE->E+E",stack,ip_sym);

else

if(!strcmpi(stack,"E\E"))

printf("\n $%s\t\t%s$\t\t\tE->E\E",stack,ip_sym);

else

if(!strcmpi(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(!strcmpi(stack,"E")&&ip_ptr==len)

{

printf("\n $%s\t\t%s$\t\t\tACCEPT",stack,ip_sym);

getch();

}

if(flag==0)

{

printf("\n%s\t\t\t%s\t\t reject",stack,ip_sym);

}

return;

}
```

OUTPUT: SHIFT REDUCE PARSER



```
                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$                  --
    $a              +b$                  shifta
    $E              +b$                  E->a
    $E+              b$                  shift+
    $E+b              $                  shiftb
    $E+E              $                  E->b
    $E                $                  E->E+E
    $E                $                  ACCEPT
```