/* SR Parser for Grammar E->E+E / E/E / E*E / E-E / id */

```
#include<stdio.h>
#include<conio.h>
#include<string.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->E-E\n E->id");
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(islower(temp2[0]))
{
stack[st_ptr]='E';
flag=1;
}
if((!strcmp(temp2,"+"))||(!strcmp(temp2,"*"))
||(!strcmp(temp2,"/"))||(!strcmp(temp2,"-")))
{
flag=1;
}
if((!strcmp(stack,"E+E"))||(!strcmp(stack,"E/E"))
||(!strcmp(stack,"E*E"))||(!strcmp(stack,"E-E")))
{
if(!strcmp(stack,"E+E"))
{
strcpy(stack,"E");
printf("\n $%s\t\t%s$\t\t\tE->E+E",stack,ip_sym);
}
else
if(!strcmp(stack,"E/E"))
{
strcpy(stack,"E");
printf("\n $%s\t\t %s$\t\t\tE->E/E",stack,ip_sym);
}
else
if(!strcmp(stack,"E-E"))
{
strcpy(stack,"E");
printf("\n $%s\t\t %s$\t\t\tE->E-E",stack,ip_sym);
}
else
{
strcpy(stack,"E");
printf("\n $%s\t\t%s$\t\t\tE->E*E",stack,ip_sym);
}
```

```
flag=1;
st_ptr=0;
}
if(!strcmp(stack,"E")&&ip_ptr==len)
{
printf("\n $%s\t\t%s$\t\t\tACCEPT",stack,ip_sym);
getch();
exit(0);
}
if(flag==0)
{
printf("\n $%s\t\t\t%s\t\t reject",stack,ip_sym);
getch();
exit(0);
}
return;
}
```