Write a C program for implementation of LR parsing algorithm to accept a given input string.

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
int point=0,top;
char inp[20],stack[100];
void push(char);
void pop();
void s0(char);
void s1(char);
void s2(char);
void s3(char);
void s4(char);
void s5(char);
void s8(char);
void push(char k)
{
if(top==99)
{
printf("string not accepted");
}
else
stack[++top]=k;
}
void pop()
{
if(top==-1)
{
printf("string not accepted");
}
else
top--;
}
void s0(char l)
{
if(l=='e')
{
printf("shift3 \n");
push(l);
push('3');
point++;
}
else if(l=='d')
{
printf("shift4 \n");
push(l);
push('4');
point++;
}
else
{
printf("string not accepted");
exit(0);
}
}
void s1(char l)
{
if(l=='$')
{
push('$');
printf("string accepted");
exit(0);
}
}
void s2(char l)
{
if(l=='e')
{
printf("shift3 \n");
push(l);
push('3');
point++;
}
else if(l=='d')
{
printf("shift4 \n");
push(l);
push('4');
point++;
}
else
{
printf("string not accepted");
exit(0);
}
}
void s3(char l)
{
if(l=='e')
{
printf("shift3 \n");
push(l);
push('3');
point++;
}
else if(l=='d')
```

```
{
printf("shift4 \n");
push(l);
push('4');
point++;
}
else
{
printf("string not accepted");
exit(0);
}
}
void s4(char l)
{
if(l=='e'||l=='d'||l=='$')
{
printf("reduce C->d \n");
pop();
pop();
push('C');
}
else
{
printf("string not accepted");
exit(0);
}
}
void s5(char l)
{
if(l=='$')
{
printf("reduce S->CC \n");
pop();
pop();
pop();
pop();
push('S');
}
else
{
printf("string not accepted");
exit(0);
}
}
void s8(char l)
{
if(l=='e'||l=='d'||l=='$')
{
printf("reduce C->eC \n");
```

```
pop();
pop();
pop();
pop();
push('C');
}
else
{
printf("string not accepted");
exit(0);
}
}

main()
{
int i,j=0,s=0,a,x;
top=-1;

printf("Grammer:\n");
printf("S->CC \n C->eC \n C->d \n");
printf("enter the ip $:");
scanf("%s",&inp);
printf("\n");
printf("STACK \t INPUT \t ACTON \n");
Printf("______________________________\n");
push('$');
push('0');
while(1)
{
j=top;
a=0;
printf("\n");
while(a<=j)
{
printf("%c",stack[a]);
a++;
}
printf("\t");
s=point;
while(inp[s]!='\0')
{
printf("%c",inp[s]);
s++;
}
printf("\t");
if(stack[top]=='0') s0(inp[point]);
else if(stack[top]=='1') s1(inp[point]);
else if(stack[top]=='2') s2(inp[point]);
else if(stack[top]=='3') s3(inp[point]);
else if(stack[top]=='4') s4(inp[point]);
else if(stack[top]=='5') s5(inp[point]);
else if(stack[top]=='8') s8(inp[point]);
else if(stack[top]=='S')
{
x=top-1;
if(stack[x]=='0')
{ printf("shift1");
push('1');  }
else
printf("string not accepted");
}
else if(stack[top]=='C')
{
x=top-1;
if(stack[x]=='0') {
printf("shift2");
 push('2');  }
else if(stack[x]=='2')
{ printf("shift5");
 push('5');
 }
else if(stack[x]=='3')
{ printf("shift8");
push('8');
}
else
printf("string not accepted");
}
else
{
printf("string not accepted");
exit(0);
}
}
}
```

OUTPUT:

```
Grammer:

S->CC
```

```
C->eC

C->d

enter the ip $:edd$

STACK       INPUT      ACTON
________________________________________

$0          edd$       shift3

$0e3        dd$        shift4

$0e3d4      d$         reduce C->d

$0e3C       d$         shift8

$0e3C8      d$         reduce C->eC

$0C         d$         shift2

$0C2        d$         shift4

$0C2d4      $          reduce C->d

$0C2C       $          shift5

$0C2C5      $          reduce S->CC

$0S         $          shift1

$0S1        $          string accepted
```