SRM UNIVERSITY AP
Andhra Pradesh

Department of Computer Science and Engineering
Compiler Design Lab (CS 306)

Name - NIKITHA GODAVARTHI
ID Number - AP19110010460
Branch/ Section - CSE-C

Week 8: Implementation of Shift Reduce Parser

Week 8 Programs

1. Implementation of Shift Reduce parser using C for the following grammar and illustrate the parser's actions for a valid and an invalid string.

   E → E+E
   E → E*E
   E → (E)
   E → d

2. Implementation of Shift Reduce parser using C for the following grammar and illustrate the parser's actions for a valid and an invalid string.

   S –> 0S0 | 1S1 | 2

Programs:

1. LEX Program for identifying the below and print the identified token along with information.

Keywords: int,char,double,void,main
Identifier: letter(letter|digit)*
Integer, Float and Relational operators

Code:

```
#include<stdio.h>
#include<stdlib.h>
void pop(),push(char),display();
char stack[100]="\0", input[100], *ip;
int top=-1;
void push(char c)
{
 top++;
 stack[top]=c;
}
void pop()
{
```

```
 stack[top]='\0';
 top--;
 }
void display()
 {
 printf("\n%s\t%s\t",stack,ip);
 }

void main()
{
        printf("E->E+E\n");
        printf("E->E*E\n");
        printf("E->(E)\n");
        printf("E->d\n");
 printf("Enter the input string followed by $ \n");
 scanf("%s",input);
 ip=input;
 push('$');
 printf("STACK\t BUFFER \t ACTION\n");
 printf("-----\t ------- \t ------\n");
 display();
 if(stack[top]=='$' && *ip=='$'){
 printf("Null Input");
 exit(0);
 }
 do
 {
 if((stack[top]=='E' && stack[top-1]=='$') && (*(ip)=='$'))
 {
 display();
 printf(" Valid\n\n\n");
 break;
 }

 if(stack[top]=='$')
 {
 push(*ip);
 ip++;
 printf("Shift");
 }
 else if(stack[top]=='d')
 {
 display();
 pop();
 push('E');
 printf("Reduce E->d");
 }
 else if(stack[top]=='E' && stack[top-1]=='+' &&
stack[top-2]=='E'&& *ip!='*')  {
 display();
 pop();
 pop();
```

```
 pop();
 push('E');
 printf("Reduce E->E+E");
 }
 else if(stack[top]=='E' && stack[top-1]=='*' &&
stack[top-2]=='E')  {
 display();
 pop();
 pop();
 pop();
 push('E');
 printf("Reduce E->E*E");
 }
 else if(stack[top]==')' && stack[top-1]=='E' &&
stack[top-2]=='(')  {
 display();
 pop();
 pop();
 pop();
 push('E');
 printf("Reduce E->(E)");
 }
 else if(*ip=='$')
 { printf(" Invalid\n\n\n");
 break;
 }
 else
 {
 display();
 push(*ip);
 ip++;
 printf("shift");
 }
 }while(1);

}
```

Testcases:

Input	Expected Output
d+d*d$	Valid
d+*d$	Invalid

2. Implementation of Shift Reduce parser using C for the following grammar and illustrate the parser's actions for a valid and an invalid string.

S –> 0S0 | 1S1 | 2

Source Code -

```
#include<stdio.h>
#include<stdlib.h>
void pop(),push(char),display();
char stack[100]="\0", input[100], *ip;
int top=-1;
void push(char c)
{
   top++;
   stack[top]=c;
}
void pop()
{
   stack[top]='\0';
   top--;
}
void display()
{
   printf("\n%s\t%s\t",stack,ip);
}
void main()
{
   printf("S -> 0S0\n");
   printf("S -> 1S1\n");
   printf("S -> 2\n");
   printf("Enter the input string followed by $ \n");
   scanf("%s",input);
   ip=input;
   push('$');
   printf("STACK\t BUFFER \t ACTION\n");
   printf("-----\t ------- \t ------\n");
   display();
   if(stack[top]=='$' && *ip=='$')
   {
      printf("Null Input");
      exit(0);
   }
   do
   {
      if((stack[top]=='S' && stack[top-1]=='$') && (*(ip)=='$'))
      {
         display();
         printf("\t Valid\n\n\n");
         break;
      }
      if(stack[top]=='$')
      {
         push(*ip);
```

```
            ip++;
            printf("Shift");
        }
        else if(stack[top]=='2')
        {
            display();
            pop();
            push('S');
            printf("\tReduce S->2");
        }
        else if(stack[top]=='0' && stack[top-1]=='S' && stack[top-2]=='0')
        {
            display();
            pop();
            pop();
            pop();
            push('S');
            printf("\tReduce S->0S0");
        }
        else if(stack[top]=='1' && stack[top-1]=='S' && stack[top-2]=='1')
        {
            display();

            pop();
            pop();
            pop();
            push('S');
            printf("\tReduce S->1S1");
        }
        else if(*ip=='$')
        {
            printf("\t Invalid\n\n\n");
            break;
        }
        else
        {
            display();
            push(*ip);
            ip++;
            printf("\tshift");
        }
    }while(1);
}
```

Test Cases -

```
S -> 0S0
S -> 1S1
S -> 2
Enter the input string followed by $
020$
STACK     BUFFER          ACTION
-----     -------         ------

$         020$     Shift
$0        20$              shift
$02       0$               Reduce S->2
$0S       0$               shift
$0S0      $                Reduce S->0S0
$S        $                 Valid



...Program finished with exit code 0
Press ENTER to exit console.
```

```
S -> 0S0
S -> 1S1
S -> 2
Enter the input string followed by $
01210$
STACK     BUFFER          ACTION
-----     -------         ------

$         01210$   Shift
$0        1210$            shift
$01       210$             shift
$012      10$              Reduce S->2
$01S      10$              shift
$01S1     0$               Reduce S->1S1
$0S       0$               shift
$0S0      $                Reduce S->0S0
$S        $                 Valid



...Program finished with exit code 0
Press ENTER to exit console.
```