

COMPILER DESIGN LAB - WEEK 8

Sagar Sujith Somepalli
AP19110010199
CSE E

Q1. Implementation of Shift Reduce parser using C for the following grammar and illustrate the parser's actions for a valid and an invalid string.

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow (E)$

$E \rightarrow d$

```
#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);
}
int 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);
}
```

Valid String Input->d+d*d

```
Console Shell

> clang-7 -pthread -lm -o main main.c
> ./main
E->E+E
E->E*E
E->(E)
E->d
Enter the input string followed by $
d+d*d$
STACK      BUFFER      ACTION
-----
$      d+d*d$  Shift
$d      +d*d$  Reduce E->d
$E      +d*d$  shift
$E+    d*d$    shift
$E+d      *d$  Reduce E->d
$E+E      *d$  shift
$E+E*     d$   shift
$E+E*d    $    Reduce E->d
$E+E*E    $    Reduce E->E*E
$E+E      $    Reduce E->E+E
$E $      Valid
```

Invalid String Input->d+*d

```
Console Shell
> clang-7 -pthread -lm -o main main.c
> ./main
E->E+E
E->E+E
E->(E)
E->d
Enter the input string followed by $
d+*d$
STACK      BUFFER      ACTION
-----
$   d+*d$   Shift
$d  +*d$   Reduce E->d
$E  +*d$   shift
$E+ *d$ shift
$E+*   d$  shift
$E+*d  $   Reduce E->d Invalid
```

Q2. Implementation of Shift Reduce parser using C for the following grammar and illustrate the parser's actions for a valid and an invalid string.

$S \rightarrow 0S0 \mid 1S1 \mid 2$

```
#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\t\t%s\t\t\t", stack, ip);
}

int 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('$');
    printf("STACK\t BUFFER \t\t ACTION\n");
    printf("-----\t ----- \t \t-----\n");
    display();
    if(stack[top]=='$' && *ip=='$'){
        printf("Null Input");
        exit(0);
    }
    do
    {
        if((stack[top]=='S' && stack[top-1]=='$') && (*ip)=='$')
        {
            display();
            printf(" Valid\n\n\n");
            break;
        }
        if(stack[top]=='$')
        {
            push(*ip);
            ip++;
            printf("Shift");
        }
        else if(stack[top]=='2')
        {
            display();
            pop();
            push('S');
            printf("Reduce S->2");
        }
    }
}

```

```

}
else if(stack[top]=='0' && stack[top-1]=='S' && stack[top-2]=='0'&&
*ip!='*')
{
    display();
    pop();
    pop();
    pop();
    push('S');
    printf("Reduce S->0S0");
}
else if(stack[top]=='1' && stack[top-1]=='S' && stack[top-2]=='1')
{
    display();
    pop();
    pop();
    pop();
    push('S');
    printf("Reduce S->1S1");
}

else if(*ip=='$')
{
    printf(" Invalid\n\n\n");
    break;
}
else
{
    display();
    push(*ip);
    ip++;
    printf("shift");
}
}while(1);
}

```

Valid String Input :- 10201

```
Console Shell

> clang-7 -pthread -lm -o main main.c
> ./main
S->0S0
S->1S1
S->2
Enter the input string followed by $
10201$
STACK      BUFFER      ACTION
-----
$          10201$      Shift
$1         0201$      shift
$10        201$       shift
$102       01$       Reduce S->2
$10S       01$       shift
$10S0      1$       Reduce S->0S0
$1S        1$       shift
$1S1       $       Reduce S->1S1
$$         $       Valid
```

Invalid String Input-> 11012

```
Console Shell

> clang-7 -pthread -lm -o main main.c
> ./main
S->0S0
S->1S1
S->2
Enter the input string followed by $
11012$
STACK      BUFFER      ACTION
-----
$          11012$      Shift
$1         1012$      shift
$11        012$      shift
$110       12$      shift
$1101      2$      shift
$11012     $      Reduce S->2 Invalid
```