# Exp 9 - Implementation of Shift Reduce Parsing

**Aim:** To write a program for Implementation of Shift Reduce Parsing

## Algorithm:

1. Start the program.
2. Initialize the required variables.
3. Enter the input symbol.
4. Perform the following:
   for top-of-stack symbol, s, and next input symbol, a
   Shift x: (x is a
   STATE number)
   Push a, then x on the top of
   the stack
   Advance ip to point to the
   next input symbol.
   Reduce y: (y is a
   PRODUCTION number)
   Assume that the production is of the form A→ß Pop 2 * |ß| symbols of the stack.
   At this point the top of the stack should be a state number, say s'.
   Push A, then goto of T[s',A] (a state number) on the top of the stack.
   Output the production A→ß.
5. Print if string is accepted or not.
6. Stop the program.

Exp 9 -Implementation of Shift Reduce Parsing
RA1911003010143
Abhishek Kumar

# Program:

```
#include <stdio.h>
#include <string.h>
int k = 0, z = 0, i = 0, j = 0, c = 0;
char a[16], ac[20], stk[15], act[10];
void check();
int main()
{
    puts("GRAMMAR is E->E+E \n E->E*E \n E->(E) \n E->id");
    puts("enter input string ");
    gets(a);
    c = strlen(a);
    strcpy(act, "SHIFT->");

    puts("stack \t input \t action");
    for (k = 0, i = 0; j < c; k++, i++, j++)
    {
        if (a[j] == 'i' && a[j + 1] == 'd')
        {
            stk[i] = a[j];
            stk[i + 1] = a[j + 1];
            stk[i + 2] = '\0';
            a[j] = ' ';
            a[j + 1] = ' ';
            printf("\n$%s\t%s$\t%sid", stk, a, act);
            check();
        }
        else
        {
            stk[i] = a[j];
            stk[i + 1] = '\0';
            a[j] = ' ';
            printf("\n$%s\t%s$\t%ssymbols", stk, a, act);
            check();
        }
    }
}
```

```
void check()
{
    strcpy(ac, "REDUCE TO E");
    for (z = 0; z < c; z++)
        if (stk[z] == 'i' && stk[z + 1] == 'd')
        {
            stk[z] = 'E';
            stk[z + 1] = '\0';
            printf("\n$%s\t%s$\t%s", stk, a, ac);
            j++;
        }
    for (z = 0; z < c; z++)
        if (stk[z] == 'E' && stk[z + 1] == '+' && stk[z + 2] == 'E')
        {
            stk[z] = 'E';
            stk[z + 1] = '\0';
```

Exp 9 -Implementation of Shift Reduce Parsing
RA1911003010143
Abhishek Kumar

```
            stk[z + 2] = '\0';
            printf("\n$%s\t%s$\t%s", stk, a, ac);
            i = i - 2;
        }
    for (z = 0; z < c; z++)
        if (stk[z] == 'E' && stk[z + 1] == '*' && stk[z + 2] == 'E')
        {
            stk[z] = 'E';
```

```
            stk[z + 1] = '\0';
            stk[z + 1] = '\0';
            printf("\n$%s\t%s$\t%s", stk, a, ac);
            i = i - 2;
        }
    for (z = 0; z < c; z++)
        if (stk[z] == '(' && stk[z + 1] == 'E' && stk[z + 2] == ')')
        {
            stk[z] = 'E';
            stk[z + 1] = '\0';
            stk[z + 1] = '\0';
            printf("\n$%s\t%s$\t%s", stk, a, ac);
            i = i - 2;
        }
}
```

# Output:

```
PS C:\Users\abhis\Desktop\Study\6th Semester\Compiler Design\Lab\Exp 9> cd "c:\Users\abhis\Desktop\Study\6th Semester\Comp
p9.c -o Exp9 } ; if ($?) { .\Exp9 }
GRAMMAR is E->E+E
 E->E*E
 E->(E)
 E->id
enter input string
id+id
stack    input   action

$id       +id$  SHIFT->id
$E        +id$  REDUCE TO E
$E+        id$  SHIFT->symbols
$E+id        $  SHIFT->id
$E+E         $  REDUCE TO E
$E           $  REDUCE TO E
PS C:\Users\abhis\Desktop\Study\6th Semester\Compiler Design\Lab\Exp 9> cd "c:\Users\abhis\Desktop\Study\6th Semester\Comp
 if ($?) { gcc Exp9.c -o Exp9 } ; if ($?) { .\Exp9 }
GRAMMAR is E->E+E
 E->E*E
 E->(E)
 E->id
enter input string
id+id*id
stack    input   action

$id       +id*id$       SHIFT->id
$E        +id*id$       REDUCE TO E
$E+        id*id$       SHIFT->symbols
$E+id        *id$       SHIFT->id
$E+E         *id$       REDUCE TO E
$E           *id$       REDUCE TO E
$E*           id$       SHIFT->symbols
$E*id           $       SHIFT->id
$E*E            $       REDUCE TO E
$E              $       REDUCE TO E
```

Exp 9 -Implementation of Shift Reduce Parsing
RA1911003010143
Abhishek Kumar

**Result:** Implementation of Shift Reduce Parsing was successfully performed.