Dept. of Computer Science & Engineering

Course Code: CSE-3528

Course Title: Compiler Lab

Spring-22

Submitted By:

Name: Farida Nusrat

ID: C201242

Semester: 5th

Section: 5AF

Email: c201242@ugrad.iiuc.ac.bd

Submitted To:

Mrs. Israt Binteh Habib

Dept. Of CSE,IIUC.

Signature: Farida Nusrat

Submission Date: 19.11.22

OBJECTIVE:

Implement shift-reduce parsing.

RESOURCE:

Code-blocks

PROGRAM LOGIC:

Shift Reduce parser attempts the construction of parse in a similar manner as done in

bottom-up parsing i.e. the parse tree is constructed from leaves(bottom) to the root(up). A more

general form of the shift-reduce parser is the LR parser.

This parser requires some data structures i.e.

- An input buffer for storing the input string.
- A stack for storing and accessing the production rules.

Basic Operations –

- **Shift**: This involves moving symbols from the input buffer onto the stack.
- **Reduce**: If the handle appears on top of the stack then, its reduction by using the appropriate production rule is done i.e. RHS of a production rule is popped out of a stack and the LHS of a production rule is pushed onto the stack.
- **Accept**: If only the start symbol is present in the stack and the input buffer is empty then, the parsing action is called accept. When accepted action is obtained, it is means successful parsing is done.
- **Error**: This is the situation in which the parser can neither perform shift action nor reduce action and not even accept action.

PROCEDURE:

Go to debug -> run or press CTRL + F9 to run the program.

PROGRAM:

```
#include <bits/stdc++.h>
using namespace std;

int k = 0, z = 0, i = 0, j = 0, c = 0;
char a[16], ac[20], stk[15], act[10];

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';
            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;
        }
}

int main()
{
    cout << "GRAMMAR is E->E+E \n E->E*E \n E->(E) \n E->id \n";
    cout << "Enter input string ";
    cin >> a;
    c = strlen(a);
    strcpy(act, "SHIFT->");
    cout << "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();
        }
    }
    return 0;
}
```

INPUT & OUTPUT:

```
GRAMMAR is E->E+E
 E->E*E
 E->(E)
 E->id
Enter input string id+id*id+id
stack    input    action
$id       +id*id+id$    SHIFT->id
$E        +id*id+id$    REDUCE TO E
$E+        id*id+id$    SHIFT->symbols
$E+id        *id+id$    SHIFT->id
$E+E         *id+id$    REDUCE TO E
$E           *id+id$    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
```

OUTCOMES:

The program is successfully written and created in the C++ language.