# ***SHIFT REDUCE PARSING***

***NAME:*** **RISHAL RAMESH** ***EXP NO:*** **7**

***REG NO:*** **RA1911030010084**

***DATE:*** **10/03/2022**

## ***AIM:***

To study and implement a shift reduce parser.

## ***LANGUAGE USED:***

C++

## ***ALGORITHM:***

1. Start the program
2. Input the number of productions
3. Input the string to be parsed
4. Perform the following basic operations on the string:
   - SHIFT: involves moving the symbols from the input buffer onto the stack
   - REDUCE: if the handle appears on top of the stack then, its reduction by using production rule is done i.e. RHS 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 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
5. Start matching the character with the given productions - If only start symbol is left, string is accepted - Else string is rejected
6. Stop the execution

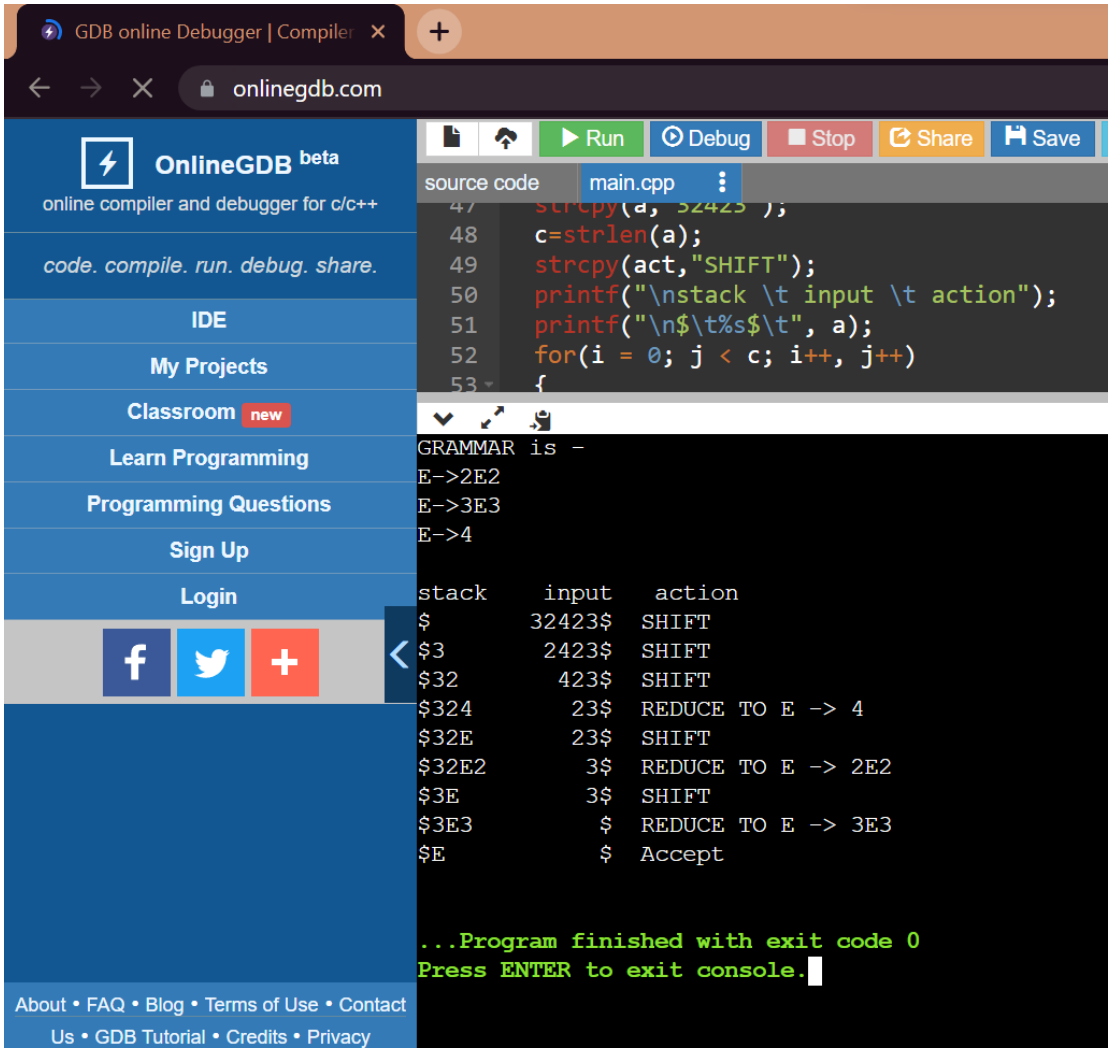
***CODE:***

```
#include <bits/stdc++.h>
using namespace std;
int 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] == '4')
    {
      printf("%s4", ac);
      stk[z] = 'E';
      stk[z + 1] = '\0';
      printf("\n$%s\t%s$\t", stk, a);
    }
  }
  for(z = 0; z < c - 2; z++)
  {
    if(stk[z] == '2' && stk[z + 1] == 'E' && stk[z + 2] == '2')
    {
      printf("%s2E2", ac);
      stk[z] = 'E';
      stk[z + 1] = '\0';
      stk[z + 2] = '\0';
      printf("\n$%s\t%s$\t", stk, a);
      i = i - 2;
    }
  }
```

```
  for(z = 0; z < c - 2; z++)
  {
    if(stk[z] == '3' && stk[z + 1] == 'E' && stk[z + 2] == '3')
    {
      printf("%s3E3", ac);
      stk[z]='E';
      stk[z + 1]='\0';
      stk[z + 1]='\0';
      printf("\n$%s\t%s$\t", stk, a);
      i = i - 2;
    }
  }
  return ;
}
int main()
{
  printf("GRAMMAR is -\nE->2E2 \nE->3E3 \nE->4\n");
  strcpy(a,"32423");
  c=strlen(a);
  strcpy(act,"SHIFT");
  printf("\nstack \t input \t action");
  printf("\n$\t%s$\t", a);
  for(i = 0; j < c; i++, j++)
  {
    printf("%s", act);
    stk[i] = a[j];
    stk[i + 1] = '\0';
    a[j]=' ';
    printf("\n$%s\t%s$\t", stk, a);
    check();
```

```
  }
  check();
  if(stk[0] == 'E' && stk[1] == '\0')
    printf("Accept\n");
  else //else reject
    printf("Reject\n");
}
```

## ***OUTPUT:***




## ***RESULT:***

The shift reduce parser was studied and executed successfully in Python.