Parsing of string in SLR(1) Parser

SLR parsing Algo:

Input: An Input string w and SLR- parsing table with function ACTION and GOTO for a grammar G

Output: If w is in L(G), the reduction steps of a Bottom up parser for w, otherwise an error indication.

METHOD: Initially S_0 (Initial State) on the Stack and w$ in the I/p Buffer

```
let 'a' be the first symbol of w$ ;
while(1)
{
let S be the state on the top of the Stack;
   if (ACTION[S,a] = shift t)
   {
     push t onto the stack;
     let a be the next I/P symbol;
   }
elseif (ACTION[S,a] = reduce A→β) {
     pop 2|β| symbol from the stack;
     let state t now be on the Top of stack;
     Push GOTO[t,A] onto the stack;
     Output the production A→β;
   }
elseif ( ACTION[S,a] = accept)
     break;
else
   error;
}
```

Parse the Input string aabb$ by SLR(1) parsing Table

	ACTION			GOTO	
	a	b	$	S	A
0	S3	S4		1	2
1			Accept		
2	S3	S4			5
3	S3	S4			6
4	r3	r3	r3		
5			r1		
6	r2	r2	r2		

1. S → AA
2. A → aA
3. A → b

⇐ Table is Build by LR(0) collection of Item

0 → Initial State on the Stack

STACK	Input string	ACTION
0	aabb$	Shift
0a3	abb$	Shift
0a3a3	bb$	Shift
0a3a3b4	b$	reduce(r_3) A → b
0a3a3A6	b$	reduce(r_2) A → aA → Symbol 2 so Four Symbol Pop from Stack
0a3A6	b$	reduce r_2 A → aA
0A2	b$	Shift
0A2b4	$	reduce(r_3) A → b
0A2A5	$	reduce(r_1) S → AA
0S1	$	Accept