Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

Stack	Symbol	Input	Action
0	$	id * id$	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Shift Action : push)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Shift Action : push 5 and id in the stack and Symbol)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : pop and push)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : id is in the production body, pop id from the Symbol and 5 from the Stack)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0	$	* id $	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : push Production head F on the Symbol)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0	$ F	* id$	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : Find the go to state from the perspective of Stack and symbol and push it on the stack)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : Pop and Push)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : F is in the production body, pop F from the Symbol and 3 from the Stack)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0	$	* id$	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : push Production head T on the Symbol)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0	$ T	* id$	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : Find the go to state from the perspective of Stack and symbol and push it on the stack)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0 2	$ T	* id$	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Shift Action : Push)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0 2	$ T	* id$	Shift to 7

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Shift Action : Push 7 and * in the Stack and Symbol)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0 2	$ T	* id$	Shift to 7
0 2 7	$ T *	id$	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Shift Action : Push)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0 2	$ T	* id$	Shift to 7
0 2 7	$ T *	id$	Shift to 5

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Shift Action : Push 5 and id in the Stack and Symbol)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0 2	$ T	* id$	Shift to 7
0 2 7	$ T *	id$	Shift to 5
0 2 7 5	$ T * id	$	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : Pop and Push)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0 2	$ T	* id$	Shift to 7
0 2 7	$ T *	id$	Shift to 5
0 2 7 5	$ T * id	$	Reduce by F -> id

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : id is in the production body, pop id from the Symbol and 5 from the Stack)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0 2	$ T	* id$	Shift to 7
0 2 7	$ T *	id$	Shift to 5
0 2 7 5	$ T * id	$	Reduce by F -> id
0 2 7	$ T *	$	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : Push Production head F on the Symbol)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0 2	$ T	* id$	Shift to 7
0 2 7	$ T *	id$	Shift to 5
0 2 7 5	$ T * id	$	Reduce by F -> id
0 2 7	$ T * F	$	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : Find the go to state from the perspective of Stack and symbol and push it on the stack)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0 2	$ T	* id$	Shift to 7
0 2 7	$ T *	id$	Shift to 5
0 2 7 5	$ T * id	$	Reduce by F -> id
0 2 7 10	$ T * F	$	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : Pop and Push)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0 2	$ T	* id$	Shift to 7
0 2 7	$ T *	id$	Shift to 5
0 2 7 5	$ T * id	$	Reduce by F -> id
0 2 7 10	$ T * F	$	Reduce by T -> T * F

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : T * F is in the production body, pop T * F from the Symbol and 10, 7, 2 from the Stack)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0 2	$ T	* id$	Shift to 7
0 2 7	$ T *	id$	Shift to 5
0 2 7 5	$ T * id	$	Reduce by F -> id
0 2 7 10	$ T * F	$	Reduce by T -> T * F
0	$	$	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : push Production head T on the Symbol)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0 2	$ T	* id$	Shift to 7
0 2 7	$ T *	id$	Shift to 5
0 2 7 5	$ T * id	$	Reduce by F -> id
0 2 7 10	$ T * F	$	Reduce by T -> T * F
0	$ T	$	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : Find the go to state from the perspective of Stack and symbol and push it on the stack)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0 2	$ T	* id$	Shift to 7
0 2 7	$ T *	id$	Shift to 5
0 2 7 5	$ T * id	$	Reduce by F -> id
0 2 7 10	$ T * F	$	Reduce by T -> T * F
0 2	$ T	$	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action :Pop and push)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0 2	$ T	* id$	Shift to 7
0 2 7	$ T *	id$	Shift to 5
0 2 7 5	$ T * id	$	Reduce by F -> id
0 2 7 10	$ T * F	$	Reduce by T -> T * F
0 2	$ T	$	Reduce by E -> T

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : T is in the production body, pop T from the Symbol and 2 from the Stack)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0 2	$ T	* id$	Shift to 7
0 2 7	$ T *	id$	Shift to 5
0 2 7 5	$ T * id	$	Reduce by F -> id
0 2 7 10	$ T * F	$	Reduce by T -> T * F
0 2	$ T	$	Reduce by E -> T
0	$	$	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : push Production head E on the Symbol)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0 2	$ T	* id$	Shift to 7
0 2 7	$ T *	id$	Shift to 5
0 2 7 5	$ T * id	$	Reduce by F -> id
0 2 7 10	$ T * F	$	Reduce by T -> T * F
0 2	$ T	$	Reduce by E -> T
0	$E	$	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Reduce Action : Find the go to state from the perspective of Stack and symbol and push it on the stack)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0 2	$ T	* id$	Shift to 7
0 2 7	$ T *	id$	Shift to 5
0 2 7 5	$ T * id	$	Reduce by F -> id
0 2 7 10	$ T * F	$	Reduce by T -> T * F
0 2	$ T	$	Reduce by E -> T
0 1	$E	$	

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

1. E -> E + T
2. E -> T
3. T -> T * F
4. T -> F
5. F -> (E)
6. F -> id

(Accept Action)

Stack	Symbol	Input	Action
0	$	id * id$	Shift to 5
0 5	$ id	* id$	Reduce by F -> id
0 3	$ F	* id$	Reduce by T -> F
0 2	$ T	* id$	Shift to 7
0 2 7	$ T *	id$	Shift to 5
0 2 7 5	$ T * id	$	Reduce by F -> id
0 2 7 10	$ T * F	$	Reduce by T -> T * F
0 2	$ T	$	Reduce by E -> T
0 1	$E	$	Accept

State	Action						Go To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker