**Part A) Solution**

**Question 1) Short Question Answers**

1. Which of the following grammars can be parsed by LR(0) but not by SLR(1),Give one line reason also?  
   A) S → a A, A → b  
   B) S → A a | b, A → c  
   C) S → a A a, A → b

Solution : None

1. Which of the following grammars will cause a shift-reduce conflict in an LR(0) parser, Give reason ?  
   A) S → a A, A → b  
   B) S → A a | b, A → c  
   C) S → a A b, A → c  
   D) S → A, A → a A | ε

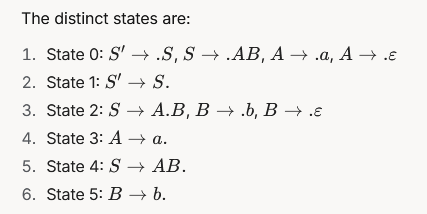
Solution: **D) S → A, A → a A | ε**  
**Reason:** In an LR(0) parser, a shift-reduce conflict occurs because the grammar’s ε-production (A → ε) allows a reduction in the same state where a terminal (‘a’) can be shifted, creating ambiguity in the parsing decision.

1. For the grammar: S → A a | B, A → b B, B → c | ε . What is the DFA state that will exhibit a shift-reduce conflict?

Solution: Start start or state with [A → b . B], [B → . c], [B → .]

1. Consider the grammar:

S → A B, A → a | ε , B → b | ε . With the augmented rule S' → S, how many states are in the LR(0) DFA, accounting for all closures and transitions?

Solution : Total: **6 states**.

1. Consider the Grammar

S → A a | B a

A → c A | d

B → c B | d

In SLR(1), a reduce-reduce conflict occurs in a state with [A → d .] and [B → d .]. Why does this conflict persist despite follow sets?

Solution : FOLLOW(A) and FOLLOW(B) both include a, causing ambiguity.

1. Consider the following grammar: Is it LR(0) or SLR(1) or Both give reason?  
   S → aAc | aBd  
   A → b  
   B → b

Solution: **Not LR(0):** The grammar has a reduce-reduce conflict in State 5 (A→b A \rightarrow b A→b vs. B→b B \rightarrow b B→b) in the LR(0) parser, as it cannot decide between the reductions without lookahead.

**SLR(1):** The conflict is resolved using FOLLOW sets (FOLLOW(A)={c} FOLLOW(A) = \{c\} FOLLOW(A)={c}, FOLLOW(B)={d} FOLLOW(B) = \{d\} FOLLOW(B)={d}), which are disjoint, making the grammar unambiguous with one-symbol lookahead. **Answer:** The grammar is **SLR(1) but not LR(0)**.

Question no 2)

1. Consider the following Grammar

S → A B C | D a

A → b A c | B d | ε

B → e B f | A C | ε

C → g | ε

D → h D | S C b

1. Compute the First sets for all non-terminals (S, A, B, C, D). **[5 marks]**
2. Compute the Follow sets for all non-terminals (S, A, B, C, D). **[5 marks]**

|  |  |  |
| --- | --- | --- |
|  | First Set | Follow Set |
| S | beghd | $bg |
| A | begd epsilon | $begdf |
| B | begh epsilon | $bdgf |
| C | g epsilon | $ bdfg |
| D | hbegd | a |

1. Determine if this grammar is LL(1). Justify your answer by analyzing potential conflicts. **[5 marks]**

**Conflict:** {b, e, g} overlaps between productions. The grammar is **not LL(1)** due to overlapping First sets for S’s productions. Or indirect left recursion

**Question no 3)**

1. Consider the following Grammar

S' → S

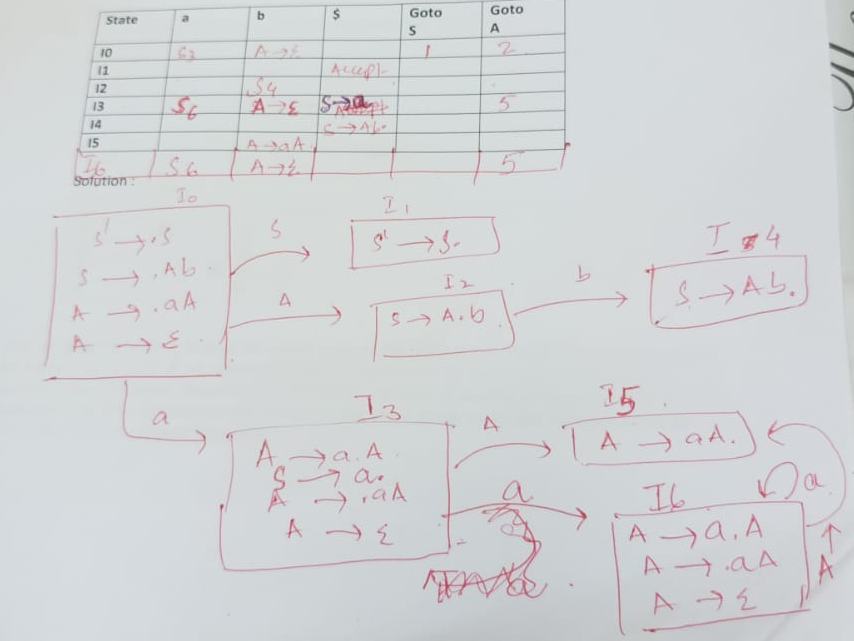
S → A b | a

A → a A | ε

Construct the DFA and SLR (1) table for the above grammar

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **State** | **a** | **b** | **$** | **Goto**  **S** | **Goto**  **A** |
| **I0** |  |  |  |  |  |
| **I1** |  |  |  |  |  |
| **I2** |  |  |  |  |  |
| **I3** |  |  |  |  |  |
| **I4** |  |  |  |  |  |
| **I5** |  |  |  |  |  |

Solution :



**Question no 4)**

4)Consider the following context-free grammar:

S → aSb | ab

Following is the SLR(1) Parsing table for the Grammar, where is

R1 = Reduce S → aSb

R2= S → ab

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **State** | **a** | **b** | **$** | **Goto**  **S** |
| **I0** | **S2** |  |  | **1** |
| **I1** |  |  | **Accept** |  |
| **I2** | **S2** | **S4** |  | **3** |
| **I3** |  | **S5** |  |  |
| **I4** |  | **R2** | **R2** |  |
| **I5** |  | **R1** | **R1** |  |

You are using an SLR(1) bottom-up parser to parse the input string aaabb. The current state of the parser’s stack is [a, S, b], where the stack is represented by the sequence of symbols (ignoring state numbers). Based on this grammar, the SLR(1) parsing mechanism, and the given stack configuration, determine what portion of the input string aaabb has been parsed so far and what remains to be parsed? **[5 marks]**

Fill the following table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Step** | **Stack** | **Input** | **Lookahead** | **Action** |
| 1 | [0] | aaabb$ |  |  |
| 2 |  |  |  |  |
| 3 |  |  |  |  |
| 4 |  |  |  |  |
| 5 |  |  |  |  |
| 6 |  |  |  |  |
| 7 |  |  |  |  |

Solution : 