Deepa P

PUZZLE On ATCD

PUZZLE 1

Tasks:

- Draw the DFA.
- Define the states, transitions, start state, and accept state(s).

PUZZLE 1

Tasks:

- Draw the DFA.
- Define the states, transitions, start state, and accept state(s).

Solution Outline:

- States: S0 (even number of 1s), S1 (odd number of 1s)
- Transitions: S0 --1--> S1, S1 --1--> S0
- Regular expression: (0*10*1)*0*

PUZZLE 2

Scenario: Given an NFA with the following transitions, convert it to a DFA.

- NFA Transitions:
  - q0 --a--> q0
  - q0 --a--> q1
  - q1 --b--> q2
  - q2 --a--> q0
- **Tasks:**
  - Draw the DFA.
  - Provide the transition table.

PUZZLE 2

Scenario: Given an NFA with the following transitions, convert it to a DFA.

- NFA Transitions:
  - q0 --a--> q0
  - q0 --a--> q1
  - q1 --b--> q2
  - q2 --a--> q0
- **Tasks:**
  - Draw the DFA.
  - Provide the transition table.

Solution Outline:

- Use the subset construction method to convert NFA to DFA.
- Create states representing sets of NFA states.

PUZZLE 3

- **Scenario:** Minimize the following DFA:
- **DFA States and Transitions:**
  - States: A, B, C, D
  - Transitions: A --0--> B, A --1--> C
  - B --0--> A, B --1--> D
  - C --0--> D, C --1--> A
  - D --0--> C, D --1--> B
- **Tasks:**

Draw the minimized DFA.

PUZZLE 3

- **Scenario:** Minimize the following DFA:
- **DFA States and Transitions:**
  - States: A, B, C, D
  - Transitions: A --0--> B, A --1--> C
  - B --0--> A, B --1--> D
  - C --0--> D, C --1--> A
  - D --0--> C, D --1--> B
- **Tasks:**

Draw the minimized DFA.

- **Solution Outline:**
- Apply the state minimization algorithm, such as partition refinement.

PUZZLE 4

- **Scenario:** Convert the regular expression a(b|c)*d into an NFA.
- **Tasks:**
- Draw the NFA.
- Show the ε-transitions if any.

PUZZLE 4

- **Scenario:** Convert the regular expression a(b|c)*d into an NFA.
- **Tasks:**
- Draw the NFA.
- Show the ε-transitions if any.

- **Solution Outline:**
- Create an NFA using Thompson's construction method.

PUZZLE 5

- **Scenario:** Convert the regular expression a(b|c)*d into an NFA.
- **Tasks:**
- Draw the NFA.
- Show the ε-transitions if any.

PUZZLE 5

- **Scenario:** Convert the regular expression a(b|c)*d into an NFA.
- **Tasks:**
- Draw the NFA.
- Show the ε-transitions if any.

- **Solution Outline:**

Use the CFG rules to define the PDA transitions.

PUZZLE 6

- **Scenario:** Construct the LL(1) parsing table for the following grammar

  Grammar:

  S → aAB
  A → b | ε
  B → c

- **Tasks:**

  Show the parsing table with appropriate entries.

PUZZLE 6

- **Scenario:** Construct the LL(1) parsing table for the following grammar

  Grammar:

  S → aAB
  A → b | ε
  B → c

- **Tasks:**

  Show the parsing table with appropriate entries.

- **Solution Outline:**

  Compute FIRST and FOLLOW sets.
  Fill the LL(1) table using the grammar rules.

PUZZLE 7

- **Scenario:** Given two regular languages represented by the following DFAs, find the intersection of the two languages.

DFA1: Accepts strings with an even number of 0s.

DFA2: Accepts strings ending in 1.

- **Tasks:**

Construct the DFA for the intersection.

PUZZLE 7

- **Scenario:** Given two regular languages represented by the following DFAs, find the intersection of the two languages.

DFA1: Accepts strings with an even number of 0s.

DFA2: Accepts strings ending in 1.

- **Tasks:**

Construct the DFA for the intersection.

- **Solution Outline:**

Use the product construction method to find the intersection DFA.

PUZZLE 8

- **Scenario:** Given the following input string: int x = 10;, identify the tokens and their types.
- **Tasks:**

Tokenize the input string and classify each token (e.g., keywords, identifiers, operators).

PUZZLE 8

- **Scenario:** Given the following input string: int x = 10;, identify the tokens and their types.
- **Tasks:**

Tokenize the input string and classify each token (e.g., keywords, identifiers, operators).

- **Solution Outline:**

Token list: int (keyword), x (identifier), = (operator), 10 (integer literal), ; (delimiter).

PUZZLE 9

- **Scenario:** Given the following expression: a + b * c, construct the syntax tree according to operator precedence.
- **Tasks:**
  Draw the syntax tree.
  Show the order of operations.

PUZZLE 9

- **Scenario:** Given the following expression: a + b * c, construct the syntax tree according to operator precedence.
- **Tasks:**
  Draw the syntax tree.
  Show the order of operations.

- **Solution Outline:**
  Build the syntax tree with correct precedence (multiplication before addition).

THANK YOU