Computer Science 12th — Chapter 2: Computational Thinking & Algorithms

Complete Short & Long Question Bank • Easy English (Textbook Exercise + Additional Questions) • istudy.com

PART 1: OFFICIAL TEXTBOOK EXERCISE QUESTIONS (SHORT & LONG)

A. TEXTBOOK SHORT QUESTIONS (ALL 10 QUESTIONS)

Q1: What is meant by computational thinking?

Answer: Computational thinking is a problem-solving process where a problem is understood, broken down into smaller parts, and solved step-by-step using clear logic.

Q2: Why is computational thinking important in problem solving?

Answer: It organizes human thinking, prevents errors, saves time, and helps create structured solutions that both humans and computers can follow easily.

Q3: What is decomposition in problem solving?

Answer: Decomposition is the process of breaking a large, complex problem into smaller, simpler, and more manageable parts.

Q4: What is an algorithm?

Answer: An algorithm is a step-by-step set of clear instructions written in the correct order to solve a specific problem.

Q5: What is logic in computer science?

Answer: Logic is a formal system of rules used by computers to test true/false conditions, reason with facts, and make accurate decisions.

Q6: What is a proposition?

Answer: A proposition is a clear statement that is either strictly True or False, but it cannot be both at the same time.

Q7: What are truth values?

Answer: Truth values are the values given to a logical proposition, which are simply **True (T)** or **False (F)**.

Q8: What is a truth table?

Answer: A truth table is a mathematical table listing all possible inputs (True/False) and showing the final output of a logical expression.

Q9: What is propositional satisfiability?

Answer: A proposition is satisfiable if there is at least one combination of truth values that makes the overall statement True.

Q10: What is a predicate in predicate logic?

Answer: A predicate is an expression containing variables that describes a property or relationship of a subject (e.g., in `Tall(Ali)`, 'Tall' is the predicate).

B. TEXTBOOK LONG QUESTIONS (ALL 8 QUESTIONS)

Long Q1: Explain computational thinking and discuss its importance in solving problems with suitable examples.

Explanation: Computational thinking is a logical thinking framework used to solve complex problems in an organized way.

Importance & 4 Pillars with Examples:

- **Decomposition:** Breaking a large problem into smaller tasks (e.g., planning a party into food, guests, and music).
- **Pattern Recognition:** Spotting trends and similarities (e.g., sorting books by subject or size).
- **Abstraction:** Focusing on vital details and ignoring useless clutter.
- **Algorithm Design:** Writing step-by-step instructions (e.g., cooking recipes or GPS navigation).

Long Q2: Describe the steps involved in problem solving using computational thinking. Explain decomposition with an example.

Problem Solving Steps: 1. Identifying the problem clearly. 2. Breaking the problem into smaller parts (Decomposition). 3. Creating step-by-step algorithms. 4. Evaluating and improving the solution.

Decomposition Example: Developing a school grading software by dividing it into: 1. Input marks, 2. Calculate percentage, 3. Assign grades, 4. Print report card.

Long Q3: What is logic as a knowledge representation framework? Explain its role in reasoning and decision making.

Framework: Logic provides computers with a structured way to store facts, conditions, and rules as knowledge.

Role in Decision Making: Computers use logical statements (like "If-Then" rules) to evaluate true/false conditions, eliminating guesswork, controlling program execution, and powering AI systems.

Long Q4: Explain propositional logic. Discuss simple and compound propositions with examples.

- **Propositional Logic:** Deals with statements that evaluate to either True or False.
- **Simple Proposition:** Contains only one statement that cannot be divided further (e.g., P = "The sun rises in the east").
- **Compound Proposition:** Formed by joining two or more simple statements using logical connectives (e.g., "It is raining AND it is cold").
- **Logical Connectives: AND (∧)** (both must be true), **OR (∨)** (at least one true), **NOT (¬)** (reverses truth).

Long Q5: What are truth tables? Explain how truth tables are created and used to evaluate logical expressions.

Truth Table: Displays the output of a logical statement for all possible true/false input combinations (2^n rows for n variables).

A	B	A AND B (∧)	A OR B (∨)	NOT A (¬A)
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

Long Q6: Explain propositional equivalence. Describe how equivalent logical expressions are identified and why equivalence is important.

Equivalence: Two logical expressions are logically equivalent (≡) if they produce the exact same truth values in every row of a truth table.

Identification & Importance: Identified by comparing truth table columns or applying logical laws. It helps programmers simplify long, complex conditional expressions into shorter, faster code without changing the result.

Long Q7: What is propositional satisfiability? Explain satisfiable and unsatisfiable propositions with examples.

- **Satisfiable Proposition:** An expression that has at least one combination of inputs that makes it True (e.g., `P OR Q`). A valid solution exists.
- **Unsatisfiable Proposition:** An expression that is False under every single input combination (e.g., `P AND NOT P`). No solution exists (contradiction).

Long Q8: Explain predicate logic and quantifiers. Discuss how predicate logic is applied to real-world problems.

- **Predicate Logic:** Extends propositional logic by breaking statements into subjects and predicates with variables (e.g., `Student(x)`).
- **Universal Quantifier (∀):** Means "for all" items in a domain (e.g., ∀x: Student(x) → AttendsClass(x)).
- **Existential Quantifier (∃):** Means "there exists at least one" item (e.g., ∃x: Book(x) ∧ Available(x)).
- **Real-World Application:** Used in modeling database queries, library loan systems, family trees, and rule-based AI reasoning.

PART 2: ADDITIONAL HIGH-YIELD EXAM QUESTIONS (TOPIC-BY-TOPIC)

A. ADDITIONAL SHORT QUESTIONS

Topic 2.1: Computational Thinking

Q1: What is Abstraction in computational thinking?

Answer: Abstraction is the technique of focusing only on the most important information while ignoring unnecessary or distracting details.

Topic 2.1: Computational Thinking

Q2: What is Pattern Recognition?

Answer: Pattern recognition is finding similarities or repeating trends between different problems to solve them more quickly.

Topic 2.2: Problem Solving

Q3: Why is evaluating and testing an algorithm important?

Answer: Evaluating checks whether the algorithm gives correct outputs, catches errors, and helps optimize steps to make them simpler and faster.

Topic 2.4: Propositional Logic

Q4: Are questions or commands considered propositions? Give reason.

Answer: No. Questions (e.g., "Where are you?") and commands (e.g., "Close the door") cannot be evaluated as True or False, so they are not propositions.

Topic 2.4: Logical Connectives

Q5: What is the rule of the logical AND (∧) operator?

Answer: The AND operator produces True only when all combined statements are True; if any statement is False, the result is False.

Topic 2.4: Logical Connectives

Q6: What is the rule of the logical OR (∨) operator?

Answer: The OR operator produces True if at least one statement is True. It is only False when both statements are False.

Topic 2.6: Propositional Equivalence

Q7: How can you prove that two logical statements are equivalent?

Answer: Construct a truth table for both statements. If their final output columns match row-for-row in all cases, they are equivalent.

Topic 2.7: Satisfiability

Q8: Give an everyday example of a satisfiable and an unsatisfiable expression.

Answer:

- **Satisfiable:** "Bring pizza OR burger" (Possible to satisfy).
- **Unsatisfiable:** "You must bring pizza AND you must NOT bring pizza" (Impossible contradiction).

Topic 2.8: Predicate Logic

Q9: Why is predicate logic more powerful than propositional logic?

Answer: Propositional logic only looks at whole sentences. Predicate logic can look inside statements to examine individual subjects, variables, and properties.

Topic 2.9: Quantifiers

Q10: Differentiate between Universal (∀) and Existential (∃) Quantifiers.

Answer:

- **Universal Quantifier (∀):** Asserts that a condition is true for every item in a domain.
- **Existential Quantifier (∃):** Asserts that a condition is true for at least one item in a domain.

Topic 2.11: Reasoning & Inference

Q11: What is Logical Inference?

Answer: Logical inference is the process of deriving and concluding new true facts from existing known facts.

Topic 2.11: Reasoning & Inference

Q12: What is Deduction using logical rules?

Answer: Deduction is a reasoning process that applies general rules to a specific case to guarantee a true conclusion (e.g., "All birds have feathers; a sparrow is a bird → a sparrow has feathers").

B. ADDITIONAL LONG QUESTIONS

Additional Long Q1: Explain Logical Inference and Deduction with Examples and their Role in AI Systems.

- **Logical Inference:** The step-by-step reasoning technique where computers derive new conclusions from known premises.
- **Deductive Reasoning:**
  - *Premise 1 (General Rule):* If it rains, the grass gets wet.
  - *Premise 2 (Specific Fact):* It is raining outside.
  - *Deduction:* Therefore, the grass is wet.
- **Role in Artificial Intelligence (AI):** Medical diagnosis expert systems, automated fraud checkers, and intelligent chatbots use deductive rules to evaluate facts and provide expert decisions automatically.

Additional Long Q2: Compare Propositional Logic vs Predicate Logic in Detail with Structured Examples.

Feature	Propositional Logic	Predicate Logic
Basic Element	Whole statements (P, Q).	Predicates and variables (`P(x)`).
Internal Details	Cannot look inside the statement.	Expresses subjects and their properties clearly.
Quantifiers	Does not support quantifiers.	Uses Universal (∀) and Existential (∃) quantifiers.
Expressive Power	Limited and basic.	Highly expressive and models complex real-world logic.
Example	`P = "Ali is tall"`	`Tall(Ali)` or `∀x: Student(x) → Intelligent(x)`