



Chapter 2

Computational Thinking & Algorithms

Student Learning Outcomes:

By the end of this chapter, students will be able to :

- apply problem-solving by effectively identifying, breaking down, and creating solutions for both straightforward and complex problems using a variety of computational thinking. They will further evaluate and refine their solutions, demonstrating their understanding of different computational techniques.
- understand practical application of fundamental computational artifacts and algorithms. This includes the ability to explain common computational problem-solving methods and a basic comprehension of primitive computational structures like lists, stacks, queues, trees, and graphs. Moreover, students will be able to use knowledge representation and reasoning, employing propositional and predicate logic in their problem-solving.

Introduction

In this chapter, students learn the basic concepts of logic and computational thinking used in computer science. They understand how to solve problems in a clear and systematic way. Students explore computational thinking techniques such as decomposition and step-by-step problem solving. They also study propositional logic, truth values, truth tables, and logical equivalence.

The chapter introduces satisfiable and unsatisfiable propositions and their role in decision-making. Students then learn predicate logic, variables, and quantifiers to express relationships and conditions. Finally, they develop logical reasoning and inference skills, building a strong foundation for analytical thinking and advanced computer science topics.

2.1 Introduction to Computational Thinking

Computational thinking is a way of thinking to solve problems clearly and logically. It helps us break big problems into smaller and simpler parts. This thinking is not only for computer science students.



TIDBIT

Computational thinking is used by scientists, engineers, and even chefs when planning complex tasks.

Computational Thinking

Computational thinking means thinking like a problem solver. It focuses on understanding a problem step by step. We break a big problem into smaller parts. We look for patterns in problems and solutions. We remove unnecessary details and focus on important points. We then create clear steps to solve the problem. These steps can be followed by humans or computers. Computational thinking helps us think logically and clearly.

correct order. This prevents errors during problem solving. Anyone can understand and use the steps easily. Computers also follow step-by-step instructions (as shown in Figure 2.2). Good planning leads to correct results.



Figure 2.2: Step by Step Algorithm Example

Evaluating and Improving Solutions

After finding a solution, it is important to evaluate it. Evaluation helps us check if the solution is correct. It allows us to find and fix mistakes. Testing the solution shows whether it works properly. We can improve the solution by making it simpler or faster. This process helps us learn from errors. Improving solutions leads to better performance. Evaluation is an important part of problem-solving.

2.3 Logic as a Knowledge Representation Framework

Logic is the science of true reasoning and is an important concept in computer science. It helps computers store, understand, and use information correctly. Using logic, complex information can be represented in a simple way.



TIDBIT

Logic gives computers a way to “think” about facts and make decisions, almost like a brain.

Logic in Computer Science

In computer science, logic means using rules to make correct decisions. Logic helps computers understand true and false conditions. It allows systems to process information correctly. Logical statements are used in programs and algorithms. These statements help control the flow of programs. Logic is the basis of many computer operations. It helps computers think in a structured way.

Role of Logic in Reasoning and Decision Making

Logic plays an important role in reasoning and decision making. It helps computers and humans draw correct conclusions. Using logic, decisions are made based on facts and rules. It removes confusion and guesswork. Logical reasoning helps solve problems step by step. Computers use logic to choose the correct action. Logic ensures that decisions are consistent and accurate. It is used in expert systems and artificial intelligence.



CLASS ACTIVITY

- Discuss how logic helps in making decisions.
- Students make simple “if-then” statements about school life.
- Example: “If it rains, then we will stay indoors.”

2.4 Propositional Logic

Propositional logic is a basic form of logic used in computer science. It deals with statements that can be true or false. These statements are called propositions (as shown in Figure 2.3)

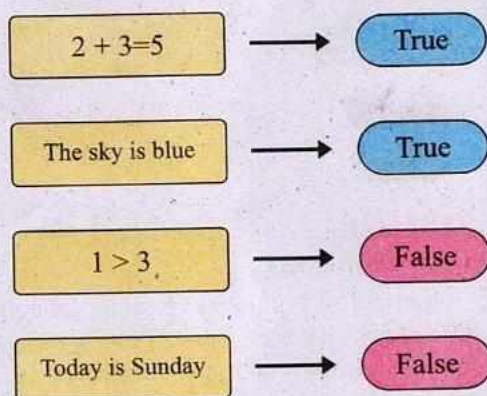


Figure 2.3: Propositional Logic Statements



DO YOU KNOW?

Propositional logic is the simplest type of logic, but it forms the base of all computer reasoning!

Proposition

A proposition is a statement that is either true or false. It cannot be both true and false at the same time. Questions and commands are not propositions. For example, “The sun rises in the east” is a proposition. A proposition must have a clear meaning. It should not be confusing. Propositions are the basic units of logic. They are used to form logical expressions. Understanding propositions is important in logic. A proposition is usually represented by a capital letter, such as P, Q, or R. For example, if p represents the statement “The sky is blue”, then we write:

$$P = \text{“The sky is blue”}$$

Simple and Compound Propositions

A simple proposition contains only one statement. It cannot be broken into smaller parts. For example, “It is raining” is a simple proposition. A compound proposition is formed by combining two or more simple propositions. Logical connectives are used to join them. For example, “It is raining and it is cold” is a compound proposition. Compound propositions can express complex ideas. They are common in logical reasoning. Both types are important in propositional logic.



Logical connectives: AND, OR, NOT

Logical connectives are words used to connect propositions. AND is used when both statements must be true. OR is used when at least one statement is true. NOT is used to reverse the truth of a statement. These connectives help form compound propositions. They are commonly used in computer programs. Logical connectives help in decision-making (as shown in Figure 2.4). They make logical expressions clear. Understanding them is essential in logic.

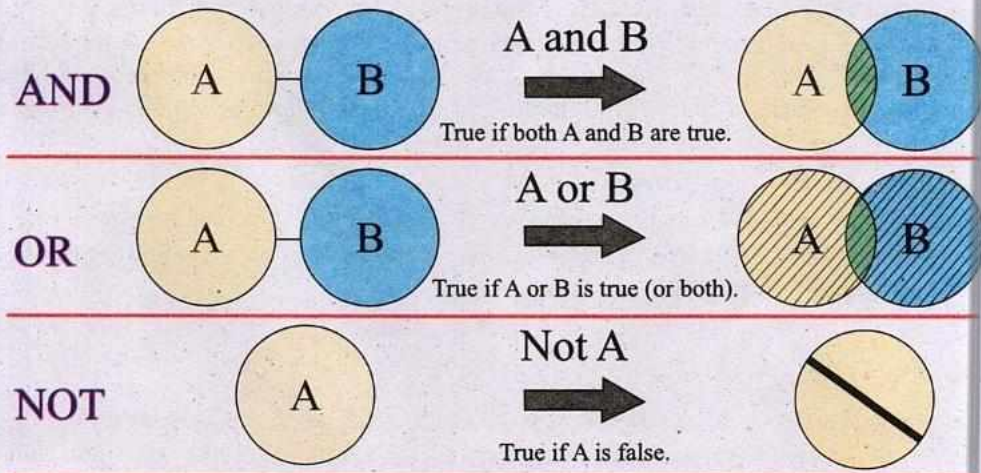


Figure 2.4: Logical Connectives

2.5 Truth Tables

Truth tables are used in logic to show the truth values of statements. They help us understand how logical expressions work. A truth table lists all possible combinations of truth values. It shows the final result for each combination. Truth tables are important in propositional logic. They are widely used in computer science and mathematics. They help in checking the correctness of logical expressions. Truth tables make logical thinking clear and simple (as shown in Figure 2.5).

A	B	A AND B	A OR B	NOT A
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

Figure 2.5: Truth Table for Logical Expressions



Purpose of Truth Tables

The main purpose of truth tables is to show how a logical expression behaves. They help us find when a statement is true or false. Truth tables list all possible input values. They help remove confusion in logical reasoning. Using truth tables, we can check complex expressions easily. They help compare different logical statements. Truth tables are useful in decision-making. They are important tools in logic and computer science.

Creating Truth Tables for Logical Expressions

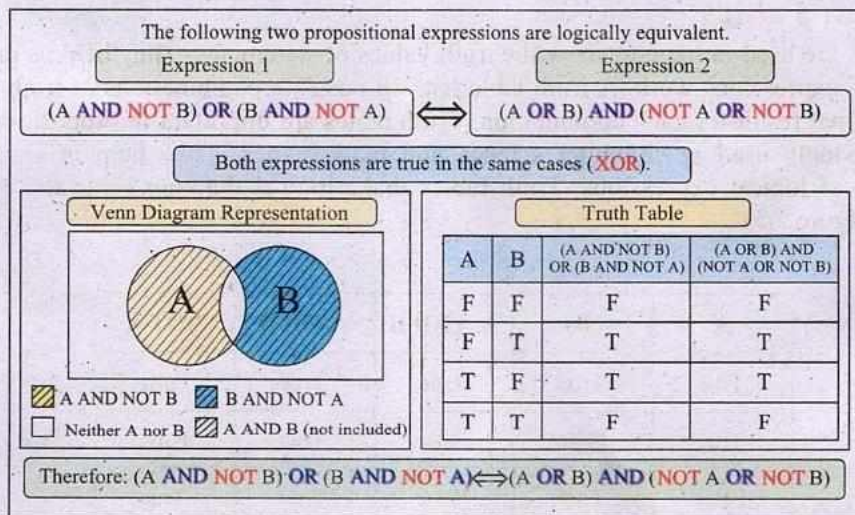
Creating a truth table starts with identifying the propositions. Each proposition is given a truth value of true or false. All possible combinations of truth values are listed. Logical connectives are applied step by step. The final result is written for each row. This process helps to understand the expression clearly. Truth tables must be neat and organized. They are easy to follow and use.

Truth Values of Propositions

Each proposition P or Q can take the value True (T) or False (F). Truth tables enumerate all possible combinations (2^n rows for n propositions). Truth value of a compound proposition is determined by the values of its components.

2.6 Propositional Equivalence

Propositional equivalence is a concept in propositional logic. It explains when two logical expressions have the same meaning. Even if expressions look different, they may give the same result. Propositional equivalence is checked using truth tables. If two expressions have the same truth values, in all cases, they are equivalent. The symbol " $\equiv$ " is used to show equivalence. Propositional equivalence improves clarity in reasoning (as shown in Figure 2.6).



TIDBIT

Equivalent expressions may look different but always give the same result; they help simplify complex logic.



Identifying Equivalent Logical Expressions

Equivalent logical expressions are identified using truth tables. Each expression is tested for all possible truth values. If the results match in every case, they are equivalent. Logical laws can also help identify equivalence. These laws simplify expressions. Identifying equivalence helps reduce complex expressions.

Importance of Equivalence in Logic

Equivalence is important because it helps simplify logic problems. It allows us to replace complex expressions with simpler ones. This saves time and effort. In programming, equivalent expressions improve code efficiency. Equivalence helps avoid logical errors. It ensures correct reasoning and decisions. This concept is useful in mathematics and computer science. Understanding equivalence strengthens logical thinking.

2.7 Propositional Satisfiability

Propositional satisfiability is a concept in logic that checks whether a statement can be true. It helps us know if a logical expression has at least one situation where it is true. This concept is widely used in computer science, especially in problem solving and programming. Satisfiability helps us test logical statements and make decisions. It is important for designing algorithms and verifying programs. Understanding satisfiability improves reasoning skills. It also helps in solving complex problems efficiently.

Example:

“Bring pizza OR (burger AND do NOT bring soda).”

Result: Bring pizza, no soda = satisfiable expression. Solution exists.

“You must bring pizza AND you must NOT bring pizza.”

Result: No solution exists = unsatisfiable expression.

DO YOU KNOW?

A logical expression is satisfiable if there is at least one way to make it true.

Conditions for a Proposition to be Satisfiable

A proposition is satisfiable if there is at least one combination of truth values that makes it true. If all possible combinations make it false, it is unsatisfiable. Satisfiability depends on the structure of the expression and the logical connectives used. Logical connectives like AND, OR, and NOT determine whether a statement can be true. Checking all possible cases using a truth table is one way to test satisfiability. A proposition with at least one “true” outcome is satisfiable. This condition helps in reasoning and decision-making.

Examples of Satisfiable and Unsatisfiable Expressions

An example of a satisfiable expression is “P OR Q.” This is true if either P or Q is true. Another example is “P AND Q,” which is true when both P and Q are true. An unsatisfiable expression is “P AND NOT P.” This is never true because P cannot be both true and false. Satisfiable expressions have at least one true outcome, while unsatisfiable expressions have none. Understanding these examples helps in learning logic and problem-solving.



CLASS ACTIVITY

Provide several expressions and ask students to classify them as satisfiable or unsatisfiable.

2.8 Predicate Logic

Predicate logic is an extension of propositional logic. It allows us to express more detailed statements about objects and their properties. In predicate logic, we can talk about individuals, groups, and relationships between them (as shown in Figure 2.7).

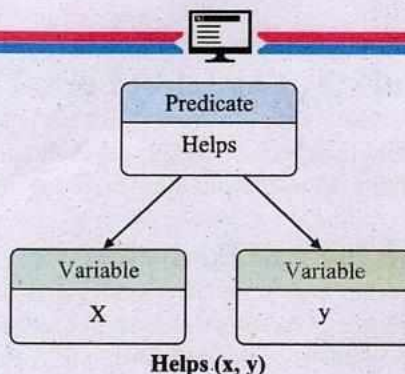


Figure 2.7: Predicate Logic Structure

Introduction to Predicate Logic

Predicate logic studies statements that contain variables and predicates. A predicate is a statement that can be true or false depending on the values of the variables. For example, "x is greater than 5" is a predicate because it depends on x. Predicate logic allows expressing statements like "All students are hardworking" or "Some books are interesting." It is used to represent knowledge formally.

Difference Between Propositional and Predicate Logic

Propositional logic deals with simple statements that are either true or false. It cannot describe properties of objects or relationships between them. Predicate logic, on the other hand, uses predicates and variables to express detailed statements. Predicate logic is more expressive and powerful than propositional logic. Propositional logic is easier to understand but less flexible. Predicate logic allows statements about many objects and conditions.

Predicates and Variables

Predicates describe properties or relationships of objects. For example, "isEven(x)" is true if x is an even number. Variables represent objects or values in the domain of discourse. Using variables, we can write general statements like "For all x, x > 0." Predicates and variables together form predicate logic expressions. They help express complex ideas clearly and formally. Understanding predicates and variables is important for reasoning and problem-solving.

Propositional Logic

A = "Ali is Tall"

- Cannot see inside (A)
- Limited

Predicate Logic

Tall(Ali)

- Can see subject Ali + property Tall
- Much more expressive

2.9 Quantifiers in Predicate Logic

Quantifiers are used in predicate logic to describe how many objects satisfy a condition. They help us make general statements or statements about specific objects. Quantifiers make predicate logic more powerful than propositional logic. Using quantifiers, we can write statements like "All students are hardworking" or "Some books are interesting." Quantifiers help represent knowledge clearly and precisely.



TIDBIT

Quantifiers like “for all” or “there exists” help us express general rules or possibilities clearly.

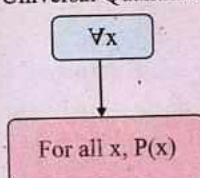
Universal Quantifier (for all)

The universal quantifier is used to indicate that a statement is true for all objects in a given set (as shown in Figure 2.8). It is denoted by the symbol “ $\forall$ ”. For example, “ $\forall x, x > 0$ ” means “For all x , x is greater than 0.” The universal quantifier is used to make general statements about every object in a domain. It is important in mathematics and logic proofs. Using “for all” ensures that no object in the set violates the statement. This quantifier helps express rules and general truths clearly.

Existential Quantifier (there exists)

The existential quantifier is used to indicate that a statement is true for at least one object in a set (as shown in Figure 2.8). It is denoted by the symbol “ $\exists$ ”. For example, “ $\exists x, x$ is even” means “There exists at least one x that is even.” This quantifier is used when a condition is satisfied by some object but not necessarily all. It helps in expressing possibilities and specific examples. The existential quantifier is important for problem solving and reasoning.

Universal Quantifier



Existential Quantifier

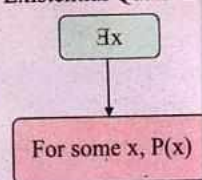


Figure 2.8: Universal and Existential Quantifier

Using Quantifiers in Logical Statements

Quantifiers can be combined with predicates to make clear logical statements. For example, “ $\forall x, P(x)$ ” means the predicate $P(x)$ is true for all x , while “ $\exists x, P(x)$ ” means $P(x)$ is true for at least one x . Quantifiers can also be used together in complex statements. For instance, “ $\forall x \exists y, x < y$ ” means for every x there exists a y that is greater than x . Using quantifiers allows us to express rules, conditions, and relationships clearly.

2.10 Applying Predicate Logic to Real-World Problems

Predicate logic can be used to solve real-world problems by representing knowledge clearly and systematically. It helps us model situations involving objects, their properties, and relationships. Using predicate logic, we can analyze problems, make decisions, and find solutions logically.



DO YOU KNOW?

Predicate logic can model family trees, social networks, and even computer programs!

Representing Relationships Using Predicates

Predicates can represent relationships between objects in the real world. For example, $Borrows(x, y)$ means “student x borrows book y ,” and $Teaches(t, s)$ means “teacher t teaches student s .” These predicates clearly describe how students, books, and teachers are connected.



Using such relationships, we can model the working of the library system. This makes it easier to organize and understand interactions between different entities.

Writing Logical Statements With Quantifiers

Quantifiers help us write general rules about the library. For example, $\forall x, \text{Student}(x) \rightarrow \text{Borrows}(x, \text{Book})$ means “every student borrows a book.” $\exists y, \text{Book}(y) \wedge \text{Available}(y)$ means “there exists at least one book that is available.” These statements help define rules and conditions in a precise way. Quantifiers allow us to describe both general situations and specific cases in the library system.



CLASS ACTIVITY

Ask students to write rules like “All students must attend class” using universal quantifiers.

Reasoning and drawing conclusions

Using predicate logic, we can draw conclusions from given facts. For example, if “ $\forall x, \text{Student}(x) \rightarrow \text{HasLibraryCard}(x)$ ” and “Ali is a student,” then we conclude “Ali has a library card.”

This logical reasoning helps in decision making, such as checking whether a student can borrow a book. It ensures correct and consistent conclusions, making the system reliable and easy to manage.

2.11 Logic-Based Reasoning and Inference

Logic-based reasoning and inference are ways of thinking that help us draw conclusions from known facts. This method uses rules of logic to reach conclusions step by step. Reasoning helps us solve problems, make decisions, and verify information. Logical inference ensures that conclusions are correct and reliable. Understanding this concept improves critical thinking skills. Logic-based reasoning is essential for solving complex real-world problems.

Logical Inference

Logical inference is the process of deriving new facts from known information. It is a way of reasoning that follows logical rules. For example, if we know “All humans are mortal” and “Socrates is a human,” we can infer that “Socrates is mortal.” Logical inference ensures conclusions are true if the premises are true. It is used in mathematics, computer science, and artificial intelligence. Inference helps in making decisions based on evidence. It is a fundamental concept in logic and reasoning.

Deduction Using Logical Rules

Deduction is a type of reasoning that starts from general facts to reach a specific conclusion. Logical rules like “if-then” statements are used in deduction. For example, “If it rains, the ground gets wet” and “It is raining” lead to the conclusion “The ground is wet.” Deduction is reliable because the conclusion is guaranteed to be true if the premises are true. Computers use deduction to solve problems and answer questions. Deduction helps in formal reasoning and problem-solving tasks.



Summary

Computational Thinking

Computational thinking is a problem-solving approach in which a problem is understood, broken into smaller parts, and solved step by step using logical thinking.

Decomposition

Decomposition is the process of breaking a large or complex problem into smaller and manageable parts.

Algorithm

An algorithm is a step-by-step set of instructions used to solve a problem or perform a task.

Logic

Logic is a formal method of representing facts, statements, and rules so that correct reasoning and decision-making can be performed.

Proposition

A proposition is a statement that is either true or false, but not both at the same time.

Compound Proposition

A compound proposition is formed by combining two or more simple propositions using logical operators such as AND, OR, and NOT.

Truth Table

A truth table is a table used to show the truth values of a logical expression for all possible combinations of its inputs.

Propositional Equivalence

Propositional equivalence means that two logical expressions have the same truth values in all possible cases.

Propositional Satisfiability

A proposition is satisfiable if there is at least one condition under which the proposition becomes true.

Predicate Logic

Predicate logic is a type of logic that represents statements using subjects, properties, variables, and relationships.

Predicate

A predicate is a logical expression that describes a property or relationship of a subject, such as (Tall(Ali)).

Quantifier

A quantifier is used in predicate logic to show how many objects a statement applies to, such as for all or there exists.

Logical Inference

Logical inference is the process of drawing a correct conclusion from given facts or statements.

Deduction

Deduction is a reasoning method in which a specific conclusion is derived from general rules or known facts.

EXERCISE

Multiple Choice Questions

- Computational thinking is mainly used for:
(a) Playing computer games (b) Solving problems in a logical way
(c) Writing essays (d) Drawing pictures
- The first step in problem solving is:
(a) Writing code (b) Evaluating the solution
(c) Identifying the problem (d) Breaking the problem
- Breaking a large problem into smaller parts is called:
(a) Abstraction (b) Decomposition
(c) Evaluation (d) Automation
- A step-by-step solution to a problem is known as an:
(a) Program (b) Flowchart
(c) Algorithm (d) Function
- The logic that uses only true or false values is:
(a) Predicate logic (b) Fuzzy logic
(c) Propositional logic (d) Natural logic
- A compound proposition is:
(a) (P) (b) (Q)
(c) $(P \wedge Q)$ (d) True
- The purpose of a truth table is to:
(a) Write programs (b) Store data
(c) Evaluate logical expressions (d) Design algorithms
- A proposition that is true for at least one case is called:
(a) Unsatisfiable (b) Invalid
(c) Satisfiable (d) Equivalent
- The symbol for the universal quantifier is:
(a) $\exists$ (b) $\longrightarrow$
(c) $\wedge$ (d) $\forall$
- Predicate logic is mainly used to:
(a) Draw diagrams (b) Represent relationships using variables
(c) Store numbers (d) Design networks

Short Questions

- What is meant by computational thinking?
- Why is computational thinking important in problem solving?
- What is decomposition in problem solving?
- What is an algorithm?
- What is logic in computer science?



6. What is a proposition?
7. What are truth values?
8. What is a truth table?
9. What is propositional satisfiability?
10. What is a predicate in predicate logic?



Long Questions

1. Explain computational thinking and discuss its importance in solving problems with suitable examples.
2. Describe the steps involved in problem solving using computational thinking. Explain decomposition with an example.
3. What is logic as a knowledge representation framework? Explain its role in reasoning and decision making.
4. Explain propositional logic. Discuss simple and compound propositions with examples.
5. What are truth tables? Explain how truth tables are created and used to evaluate logical expressions.
6. Explain propositional equivalence. Describe how equivalent logical expressions are identified and why equivalence is important.
7. What is propositional satisfiability? Explain satisfiable and unsatisfiable propositions with examples.
8. Explain predicate logic and quantifiers. Discuss how predicate logic is applied to real-world problems.



Answer Key

1. B
2. C
3. B
4. C
5. C
6. C
7. C
8. C
9. D
10. B