K. K. Wagh Institute of Engineering Education and Research, Nashik
(An Autonomous Institute from A.Y. 2022-23)

Model Answer
End-Sem Examination-I, Winter 2025

Academic Year: 2024-2025	Semester: II
Class: F.Y.	Program: B. Tech
Branch Code: COM/ADS/CSD	Pattern: 2023
Name of Course: Computational Thinking and Problem Solving	Course Code: 2300118A

Q. No	Answer Details	Max Marks
Q1	Computational Thinking is a problem-solving approach that involves formulating problems in a way that enables their solutions to be represented and executed by a computer or by a human using systematic and logical steps. It focuses on breaking down complex problems and designing efficient, logical solutions. The core components of Computational Thinking are: 1. **Decomposition** – Breaking a complex problem into smaller, manageable sub-problems. 2. **Pattern Recognition** – Identifying similarities or patterns among problems or data to simplify solutions. 3. **Abstraction** – Focusing on the essential details while ignoring irrelevant information. 4. **Algorithmic Thinking** – Developing a step-by-step procedure or set of rules to solve the problem.	6
Q2	**Procedural Programming:** ● Focuses on procedures/functions. ● Program is divided into functions. ● Data and functions are separate. ● Example languages: C, Pascal. *Example:* A function to calculate factorial using steps.	6

	Object-Oriented Programming (OOP): ● Based on objects and classes. ● Combines data and methods. ● Supports encapsulation, inheritance, polymorphism. ● Example languages: Java, Python, C++. *Example:* A `Student` class with attributes and methods.	
Q3 a)	**Algorithm:** 1. Start 2. Read N 3. Set count = 1, num = 2 4. While count ≤ N ○ Print num ○ num = num + 2 ○ count = count + 1 5. Stop	6
	OR	
b)	Start Read n fact = 1 For i = 1 to n ● fact = fact × i Print fact Stop	6
c)	Start Read base, exponent result = 1 While exponent > 0 ● If exponent is odd, result = result × base ● base = base × base ● exponent = exponent // 2 Print result Stop	5
	OR	

d)	Start Read n If n ≤ 1, print Not Prime For i = 2 to √n ● If n % i == 0, print Not Prime and Stop Print Prime Stop	5
e)	Start Read N a = 0, b = 1 Print a, b For i = 3 to N ● c = a + b ● Print c ● a = b, b = c Stop	5
	OR	
f)	Start Read a, b While b ≠ 0 ● temp = b ● b = a % b ● a = temp Print a Stop	5
Q.4 a)	Linear search checks each element sequentially until the key is found or list ends. **Algorithm:** 1. Start 2. Read array and key 3. For i = 0 to n-1 ○ If arr[i] == key, print Found and Stop 4. Print Not Found 5. Stop	6

	OR	
b)	Repeatedly swaps adjacent elements if they are in wrong order. **Steps:** Compare adjacent elements and swap; largest element bubbles to the end.	6
c)	Start Read sorted array and key low = 0, high = n-1 While low ≤ high • mid = (low + high) // 2 • If arr[mid] == key, print Found • Else if key < arr[mid], high = mid - 1 • Else low = mid + 1 Stop	5
	OR	
d)	Start For i = 0 to n-1 • min = i • For j = i+1 to n-1 ○ If arr[j] < arr[min], min = j • Swap arr[i] and arr[min] Stop	5
e)	• Start from the first element. • Compare each element with 44. • The search stops when 44 is matched at position 4. **Outcome:** Linear Search successfully identifies the position of the target element through step-by-step comparison.	5
	OR	

f)	• Compare 40 with the middle element 30. • Since 40 > 30, search the right half. • Compare with the next middle element 40 → match found. Outcome: Binary Search efficiently locates the target element by applying the divide-and-conquer approach.	5
Q.5 a)	Explanation: Finding occurrence of a pattern in a given text.	6
	OR	
b)	**Explanation:** Text is aligned to the right margin. **Advantages:** Improves readability, used in reports and tables.	6
c)	Text justification improves the alignment of text in documents. Paragraph justification means adjusting the spaces between words so that each line has equal width. For a width of 30 characters, extra spaces are distributed evenly between words in a line, ensuring both left and right margins are aligned. This helps in improving the visual appearance and readability of the text.	5
	OR	
d)	**Text Wrapping** is a text processing technique used to arrange text within a specified line width so that words automatically move to the next line when the current line exceeds the given width. This improves readability and ensures uniform formatting.	5

	Explanation ● A maximum line width is defined (e.g., 20 or 30 characters). ● Words are placed sequentially on a line. ● If adding a word exceeds the given width, the word is shifted to the next line. ● Words are **not split**; wrapping occurs at word boundaries.	
e)	Start Read document and keyword count = 0 For each word in document ● If word == keyword, count++ Print count Stop	5
	OR	
f)	Truncation is a text processing technique in which a string or paragraph is shortened to a specified length by cutting off extra characters or words, without rounding or reformatting the remaining content. Explanation: When the length of text exceeds a predefined limit (such as display width, character limit, or storage constraint), truncation is applied to retain only the initial portion of the text. It is commonly used in: ● Text editors ● User interfaces (labels, previews) ● Search results and summaries	5