JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

TEST -1 EXAMINATION- 2026

B.Tech-IV Semester (CSE/IT Backlog)

COURSE CODE (CREDITS):18B11CI412 (3)

MAX. MARKS: 15

COURSE NAME: Design and Analysis of Algorithms

COURSE INSTRUCTOR: Amol Vasudeva MAX. TIME: 1 Hour

Note: (a) All questions are compulsory.
(b) The candidate is allowed to make Suitable numeric assumptions wherever required for solving problems
(c) Use of calculators is not allowed

Q.No	Question	CO	Marks

Q1	Find the time complexity of the following.
	`int value = 0;`
	`for(int i=0; i < n; i++)`
	`    for(int j=0; j< i; j++)`
	`        value += 1;`
		1	2

Q2	a) Write the increasing order of asymptotic complexity of functions f1, f2, f3 and f4. $f1(n) = 2^n$ $f2(n) = n^{(3/2)}$ $f3(n) = n \text{ Log } n$ $f4(n) = n^{(\text{Log } n)}$ b) Is $2^{2n} = O(2^n)$? Prove it.	1	2+2 = 4

Q3	Generate the recurrence relation of the following function.
	`void b_search(int a[], int low, int high, int key)`
	`{`
	`    int mid = (low + high) / 2;`
	`    if (low <= high)`
	`    {`
	`        if (a[mid] == key)`
	`            cout << "Element found at index: " << mid;`
	`        else if(key < a[mid])`
	`            b_search(a, low, mid-1, key);`
	`        else if (a[mid] < key)`
	`            b_search(a, mid+1, high, key);`
	`    }`
	`    else if (low > high)`
	`        cout << " \n Element not found!";`
	`}`
		2	3

Q4	Use master method to solve the following recurrence. $T(n) = 2T(n/4) + (n)^{1/2}$	2	3
Q5	Solve the recurrence $T(n) = n^{1/2}\, T(n^{1/2}) + n$ using substitution method.	2	3

174