Roll Number: *Thapar University Patiala*

Department of Computer Science & Engineering

ME-CS, IS, CA and SE (1st Semester)MST Sep15 PCS-104: Advanced Data Structures and algorithms

Time: 02 Hours; MM: 20 Name of Faculty: Deepak Garg

Note: Solve any four out of the six questions. Only Four Questions will be checked. Cross all other answers, which you don't want to get checked, if you have tried to attempt more than four.

1	Give the Best, Average and Worst case examples and complexities of the following algorithms: a) Insertion Sort b) Quick Sort c) Bucket Sort Solve the following Recurrence Relations: a) $T(n) = 16\,T(n/4) + n^2$ b) $T(n) = T(9n/10) + n$	3 2
2	a) What makes an algorithm Randomized?Is Randomized Quicksort better from normal Quicksort? Give appropriate examples to prove your points. *b) "For any choice of hash functions, there exists a bad set of keys that assign all hash to the same slot."* Justify the above statement by giving appropriate examples.Suggest a solution with examples, in order to remove the above weakness of hashing.	2.5 2.5
3	a) Show the insertion of following elements in an empty BST, along with algorithm: **5, 2, 1, 3, 12, 9, 21, 19, 25** Now, delete the following elements, in order, from the above BST. Also give the algorithm for different deletion cases in BST: **9, 12, 21, 5** b) Justify the following statement with appropriate examples or algorithms: *"Divide and conquer is not suitable where the solution of size n depends upon n sub-solutions, each of size (n-1) and Overlapping Sub Problems."*	3 2
4	Given a decimal number n, convert the number to its binary representation. a) Suggest the data structure that will be used to solve this problem. b) Write the pseudo-code of the algorithm to solve the above problem using the suggested data structure.	5

5	Write the recurrence relation for the given 2 algorithms.
	Solve the above recurrences to find out the actual Big O complexity of the following algorithms using any of the 3 methods.
	
	1)
	``
	int recSearch1(int arr[], int l, int r, int x)
	{
	  if (r < l)
	  return -1;
	  if (arr[l] == x)
	  return l;
	  return recSearch1(arr, l+1, r, x);
	}
	``
	2)
	``
	int recSearch2(int a[], int x, int l, int h)
	{
	  int m;
	  if (l > h) { return -1; }
	  m = (l + h) / 2;
	  if (x == a[m]) { return (m); }
	  else if (x < a[m]) { recSearch2 (a, x, l, m - 1); }
	  else { recSearch2 (a, x, m + 1, h);}
	}
	``
		2.5
		+
		2.5

6	Given a binary tree T (not a binary search tree) and two values say n1 and n2. The lowest common ancestor between two nodes n1 and n2 is defined as the lowest node in T that has both n1 and n2 as descendants (where we allow a node to be a descendant of itself). a) Write the algorithm to find the lowest common ancestor. b) Give the complexity of your algorithm.	5