SRN

PES University, Bangalore
(Established under Karnataka Act No. 16 of 2013)

UE15CS202

Nov - Dec 2016: END SEMESTER ASSESSMENT (ESA) B.TECH. III SEMESTER
UE15CS202- Data Structures

Time: 3 Hrs — Answer All Questions . Neglect syntax errors — Max Marks: 100

1.	a)	i. Consider the following arithmetic expression. Simulate the action of the algorithm to convert the expression into postfix expression using stack. **((A+B)*(C/D)+E-(F+G))** ii. Apply the evaluation algorithm to evaluate the following postfix expression. Assume A=1, B=2, C=3. **ABC+*CBA-+***	06
	b)	A list has **n - 1** nodes whose data elements have values from 1 to n in increasing order - with one value missing. Find the missing value. NODE is a structure of singly linked list . **int missing_element(Node list)** **{ // TODO }** Ex: 1→2→3→5→6. Here missing element is 4.	04
	c)	Write only a function in C to reverse a given input string inputstr = " MYNAME" using **stack**. Store the output string in the variable outputstr = "EMANYM".	04

	d)	Consider the NODE definition as follows:
		typedef struct node {
		int info;
		struct node *next;
		}NODE;
		Write a function to check whether the two lists are same. Check all extreme conditions.
		int Same(Node list1, Node list2)
		{ // TO DO
		}
			06

2.	a)	Consider a doubly linked list node as follows:
		struct node {
		int data;
		struct node *prev, *next;
		};
		A node in the middle of a doubly linked list is connected to its next node and its previous node, but the **links from those two nodes to the current node** have gone corrupt.
		Write only the statements to correct these links in the function correct.
		void correct(Node temp)
		{ // TO DO }
			04

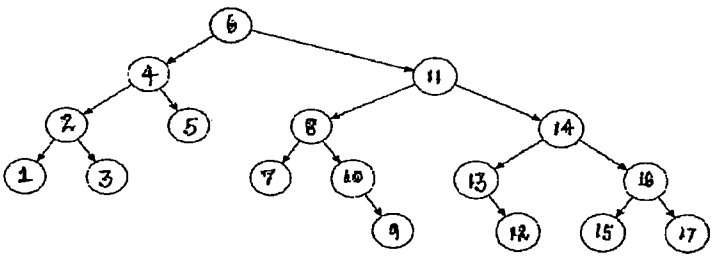
	b)	Write a function ***prefix*** to accept an infix string and create the prefix form of that string. Assume that the string is read from right to left and the prefix string is created from right to left. Ex: Input string is **A*B+C**. Traverse from right that is from B towards A. Output is **CBA*+.**	06

SRN

	c)	Consider a doubly ended queue **[deque]**. The following functions are performed for an **input restricted doubly ended queue** as a deque where only these operations removefirst, removelast and insertfirst are valid and an **output restricted doubly ended queue** as a deque where only the functions removefirst, insertlast, insertfirst are valid. **Show** how each of these can be used to represent both **a stack and a queue**.	04

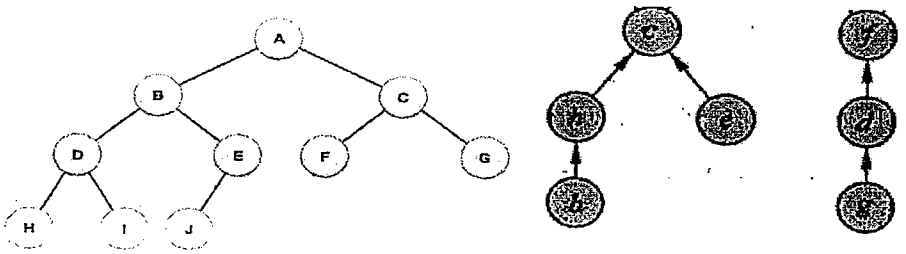
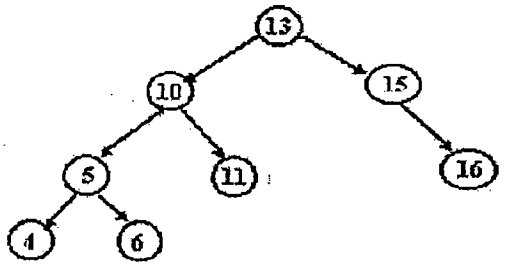
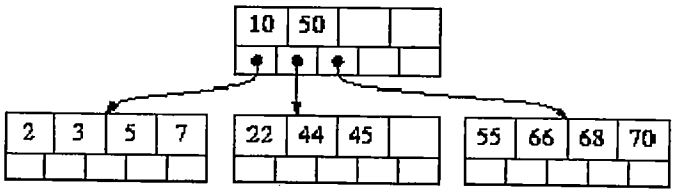
	d)	What does the following code snippet do? Assume structure definition of a SLL.
		node_t* **whatt**(node_t* list, int key)
		{ node_t* temp = list;
		while(temp && ! (temp->key == key))
		{ temp = temp->link; }
		return temp;
		}
		Modify the function **whatt( )** if the list supports a **tail node.**
		Indicate why this would be **more efficient.**
			04
			+
			02

3.	a)	What does the following mean with reference to trees? Write a tree to explain the same. i. Internal node ii. Forest iii. Balancing factor iv. Full binary tree v. Sibling	05
	b)	Consider the following elements. Write the binary search tree structure and delete the element **Ganga.** Construct the tree again. Will the tree still be a BST? Saraswathi , Ganga, Yamuna, Sindhu, Kaveri, Godavari, Narmada, Sutlej, Cauvery, Zodiac, Amazon. Now, traverse the tree in postorder and preorder.	05

	c)	Consider the following binary [redacted] tree. Write the output of the following function.
		 
		`void disp(Node root)`
		`{ Node root1 = root;`
		`while(root != NULL)`
		`{ printf("%d", root→data);`
		`root = root→left; }`
		`while(root1 != null)`
		`{ printf("%d",root1→data);`
		`root1 = root1→right;`
		`}`
		`}`
			05

	d)	What are the basic operations of the abstract data type queue? State these as C function declarations.	05
4.	a)	Write a function to insert a key element into a ordered circular singly linked list. Check for all cases.	06
	b)	Implement a function *pqmindelete()* from the *priority queue* implemented using heap and adjust the priority queue to form a heap again.	06

SRN

	c)	Construct a expression tree for the following infix expression. 3 + 4 * (6 – 8) / 7 +5. Traverse the expression tree in postorder and preorder.	04
	d)	Consider the following trees in the forest. Transform the forest into a corresponding binary tree. 	04
5.	a)	Consider the following binary search tree. Determine the balancing factors for the nodes in the tree. Is the tree balanced?  Now, insert 14 and 17. Will the tree still be balanced? Can the tree now be called a heap? Also, now insert 3. Is it balanced? If not balance the same.	06
	b)	Consider the tree given below. Consider a B Tree of order 5.  Perform insert & delete operations of the following elements on the B tree. Indicate when split or merge occurs and write the tree structure accordingly after every operation. **Insert 17, insert 6, Delete 7**	05
	c)	Consider a hash table of tablesize 16. Calculate the corresponding index value and store the given elements. Use the hash function (keyvalue % tablesize). Use open addressing. How are collisions avoided using quadratic rehash? Compute the index using rehashing done for 3rd time. **987042, 984016, 985043, 986055, 966155, 908128, 999032, 786516, 567092, 777872** Show the computations and the hash table.	06
	d)	Differentiate between dynamic and static memory allocation methods.	03