**HOME WORK**

**Home Work based on Lecture 2.1**

**Home Work based on Lecture 2.1**

1. What data structure can be used to check if a syntax has balanced parenthesis?
   1. Queue
   2. Tree
   3. List
   4. **Stack**
2. To evaluate an expression without any embedded function calls:
   1. **One stack is enough**
   2. Two stacks are needed
   3. As many stacks as the height of the expression tree are needed
   4. A Turing machine is needed in the general case
3. The condition when user tries to remove from an empty stack is called …….
   1. Overflow of Stack
   2. Garbage Collection
   3. **Underflow of Stack**
   4. Empty Collection

**Home Work based on Lecture 2.2**

1. Suppose implementation supports an instruction REVERSE, which reverses the order of elements on the stack, in addition to the PUSH and POP instructions. Which one of the following statements is TRUE with respect to this modified stack?
   1. A queue cannot be implemented using this stack.
   2. A queue can be implemented where ENQUEUE takes a single instruction and DEQUEUE takes a sequence of two instructions.
   3. **A queue can be implemented where ENQUEUE takes a sequence of three instructions and DEQUEUE takes a single instruction.**
   4. A queue can be implemented where both ENQUEUE and DEQUEUE take a single instruction each.
2. How many queues are needed to implement a stack?
   1. 1
   2. **2**
   3. 3
   4. 4

**Home Work based on Lecture 2.3**

1. The time required to search an element in a linked list of length n is
   1. O(log n)
   2. **O(n)**
   3. O(1)
   4. O(n^2)
2. Let P be a singly linked list. Let Q be the pointer to an intermediate node x in the list. What is the worst-case time complexity of the best known algorithm to delete the node x from the list?
   1. **O(n)**
   2. O(log2 n)
   3. O(logn)
   4. O(1)

**Home Work based on Lecture 2.4**

1. Which of the following is true?
   1. B + tree allows only the rapid random access
   2. B + tree allows only the rapid sequential access
   3. **B + tree allows rapid random access as well as rapid sequential access**
   4. B + tree allows rapid random access and slower sequential access
2. Statement 1: When a node is split during insertion, the middle key is promoted to the parent as well as retained in right half-node.

Statement 2: When a key is deleted from the leaf, it is also deleted from the non-leaf

nodes of the tree.

* 1. **Statement 1 is true but statement 2 is false**
  2. Statement 2 is true but statement 1 is false
  3. Both the statements are true
  4. Both the statements are false

1. Visiting root node after visiting left and right sub-trees is called
   1. In-order traversal
   2. Pre-order traversal
   3. **Post-order traversal**
   4. None of these

**Home Work based on Lecture 2.5**

1. Why to prefer red-black trees over AVL trees?
   1. Because red-black is more rigidly balanced
   2. **AVL tree store balance factor in every node which costs space**
   3. AVL tree fails at scale
   4. Red black is more efficient
2. To restore the AVL property after inserting a element, we start at the insertion point and move towards root of that tree. is this statement true?  
   **a) true**  
   b) false
3. Given an empty AVL tree, how would you construct AVL tree when a set of numbers are given without performing any rotations?  
   a) just build the tree with the given input  
   **b) find the median of the set of elements given, make it as root and construct the tree**c) use trial and error  
   d) use dynamic programming to build the tree

**Home Work based on Lecture 2.6**

1. Given the following input (4322, 1334, 1471, 9679, 1989, 6171, 6173, 4199) and the hash function x mod 10, which of the following statements are true? (GATE CS 2004)  
   i. 9679, 1989, 4199 hash to the same value  
   ii. 1471, 6171 has to the same value  
   iii. All elements hash to the same value  
   iv. Each element hashes to a different value  
   (A) i only  
   (B) ii only  
   **(C) i and ii only**  
   (D) iii or iv
2. Which of the following is not a technique to avoid a collision?  
   a) Make the hash function appear random  
   b) Use the chaining method  
   c) Use uniform hashing  
   d) **Increasing hash table size**

**Home Work based on Lecture 2.7**

1. In a binary min heap containing n elements, the largest element can be found in time
   1. **O(n)**
   2. O(nlogn)
   3. O(logn)
   4. O(1)
2. Given an unsorted array. The array has this property that every element in array is at most k distance from its position in sorted array where k is a positive integer smaller than size of array. Which sorting algorithm can be easily modified for sorting this array and what is the obtainable time complexity?
   1. Insertion sort with time complexity O(kn)
   2. **Heap sort with time complexity O(nLogk)**
   3. Quick sort with time complexity O(kLogk)
   4. Merge sort with time complexity O(kLogk)

**Home Work based on Lecture 2.8**

1. What is the number of edges present in a complete graph having n vertices?  
   a) (n\*(n+1))/2  
   **b) (n\*(n-1))/2**  
   c) n  
   d) Information given is insufficient.
2. A connected planar graph having 6 vertices, 7 edges contains \_\_\_\_\_\_\_\_\_\_\_\_regions.  
   a) 15  
   **b) 3**  
   c) 1  
   d) 11
3. For a given graph G having v vertices and e edges which is connected and has no cycles, which of the following statements is true?  
   a) v=e  
   **b) v = e+1**  
   c) v + 1 = e  
   d) v = e-1