Data Structures and Algorithm

1. Can we solve this using the sorting technique?  
 We can combine two techniques to get a solution to the problem. This can be done as:-  
 Create an array and store all the addresses of the nodes.  
 Now sort this array.  
 For each element in the second list, from the beginning search for the address in the array.  
 We can use a very efficient search algorithm like Binary Search which gives us the result in O(log n).  
 If we find a same memory address, that means that is the merging point of the 2 lists.  
   
2. Can we solve it using hash tables?  
 Yes we can solve it using hash table. This method is similar to the brute force method. The following steps are involved in it:-  
 • Select a List, that has fewer number of elements. We can get the number of elements, by a single scan on both the lists. If both the lists have same number of elements, select any list at random.  
 • Create a hash table using the list with fewer elements. Creating a hash table means storing the address of each of the nodes of the smaller list in a separate data structure such as an array.  
 • Now, traverse the other list and compare the address of each of the node with the values in the hash table.  
 • If there exists an intersection point, certainly we will find a match in the hash table and we will obtain the intersection point  
   
3. Can we use stacks for solving?  
 Yes, we can solve this problem using the concept of stacks. A stack is a data structure which follows the principle of LAST IN FIRST OUT (LIFO).  
   
4. Is there any other way of solving this ?  
 Yes, we can use Brute force method to solve this  
   
5. Can we improve the complexity for?  
 Yes,we can improve complexity.  
 The steps involved are:-  
 • Find the length of both the lists. Let ‘m’ be the length of List 1 and ‘n’ be the length of List 2.  
 • Find the difference in length of both the lists. d = m – n  
 • Move ahead ‘d’ steps in the longer list.  
 • This means that we have reached a point after which, both of the lists have same number of nodes till the end.  
 • Now move ahead in both the lists in parallel, till the ‘NEXT’ of both the lists is not the same.  
 • The NEXT at which both the lists are same is the merging point of both the lists.

Time Complexity: O(max(m,n))  
 Space Complexity: O(1)

a)How will you solve it without recursion?  
 By using iterative method we can find size of binary tree

Veluganes V S

211099