1. Is it possible to find a loop in a Linked list ?
a. Possilbe at O(n)
b. Not possible
c. Possible at O(n^2) only
d. Depends on the position of loop

Solution: a. Possible at O(n)
Have two pointers say P1 and P2 pointing to the first node of the list.
Start a loop and Increment P1 once and P2 twice in each iteration. At any point of time if P1==P2 then there is a loop in that linked list. If P2 reaches NULL (end of linked list) then no loop exists.

2. Two linked lists L1 and L2 intersects at a particular node N1 and from there all other nodes till the end are common. The length of the lists are not same. What are the possibilities to find N1?.

a. Solution exist for certain cases only
b. No linear solution exist
c. Linear solution is possible
d Only Non-linear solution exist.
Solution: c. Linear solution is possible
Have two pointers say P1 pointing to the first node of L1 and P2 to that of L2. Traverse through both the lists. If P1 reaches L1's last node, point it to the first node of L2 and continue traversing. Do the same thing for P2 when it reaches L2's last node. (By doing this, we are balancing the difference in the length between the linked lists. The shorter one will get over soon and by redirecting to longer list's head, it will traverse the extra nodes also.) Finally they will Meet at the Intersection node.

3. void PrintTree (Tree T)
{
if (T != NULL)
{
PrintTree (T-> Left);
PrintElement (T-> Element);
PrintTree (T->Right);
}
}
The above method 'PrintTree' results in which of the following traversal
a Inorder
b. Preorder
c. Postorder
d. None of the above
Solution: a. Inorder
Inorder:
void PrintTree (Tree T)
{

```
if (T != NULL)
{
PrintTree (T-> Left);
PrintElement (T-> Element);
PrintTree (T->Right);
}
}
```

For preorder use this order

```
PrintElement (T-> Element);
PrintTree (T-> Left);
PrintTree (T->Right);
```

For postorder use this order

```
PrintTree (T-> Left);
PrintTree (T->Right);
PrintElement (T-> Element);
```

4. Given a Binary Search Tree (BST), print its values in ascending order.

a. Perform Depth first traversal
b. Perform Breadth first traversal
c. Perform Postorder traversal
d. Perform Inorder traversal
Solution: d. Perform Inorder traversal
It is the properfy of BST and Inorder traversal.

5. Is it possible to implement a queue using Linked List ?. Enqueue & Dequeue should be O(1).

a. Not possible to implement.
b Only Enqueue is possible at O(1).
c. Only Dequeue is possible at O(1).
d. Both Enqueue and Dequeue is possible at O(1)
Solution: d. Both Enqueue and Dequeue is possible at O(1)
Have two pointers H pointing to the Head and T pointing to the Tail of the linked list.
Perform enqueue at T and perform dequeue at H. Update the pointers after each operations accordingly.

6. Given a Tree, is it possible to find the greatest and least among leaves in linear time?.

a. Solution depends on the tree structure
b.Linear solution exist
c. Only Non-linear solution exist.
d. No linear solution exist
Solution: b. Linear solution exist

Have two variables Min and Max. Perform any tree traversal.Assign the first traversed leaf element to Min and Max for all other leaf elements check with these variables and update it accordingly. If a current element is < Min then update Min with that element. If it is > Min then check with Max.
Note: If you want to find the greatest and least among all nodes perform the checks for each node traversed.

7. Is it possible to find find the greatest and least value among the nodes in a given BST without using any extra variables?

a. No solution exist.
b. Solution need 2 extra variables
c. Solution exist without any extra variables
d Solution need 1 extra variable
Solution:c Solution exist without any extra variables
As per BST property, the left most node should be the least one and the rightmost node should be the greatest. In other words, the first and last node of an Inorder traversal are the least and greatest among the nodes respectively.

8. Is it possible to implement 2 stack in an array?

Condition: None of the stack should indicate an overflow until every slot of an array is used.
a. Only 1 stack can be implemented for the given condition
b. Stacks can not be implemented in array
c. 2 stacks can be implemented for the given condition.
d. 2 stacks can be implemented if the given condition is applied only for 1 stack.
Solution:c. 2 stacks can be implemented for the given condition
Start 1st stack from left (1st position of an array) and 2nd from right (last position say n).
Move 1st stack towards right(i.e 1,2,3 ...n) and 2nd towards left (i.e n,n-1,n-2...1).

9. Given two keys K1 & K2, write an algorithm to print all the elements between them with K1<=K2 in a BST.

a. Solution need 2 extra spaces
b. Linear solution is possible without using any extra space
c No linear solution exist
d Solution need 1 extra space
Solution:b. Linear solution is possible without using any extra space
Perform an inorder traversal. Once you find K1 print it and continue traversal now, print all other traversed elements until you reach K2.
Note: If K1 == K2 stop once you find K1.

10. How many stacks are required to implement a Queue.

a. One
b. Two
c. Three
d. Two + one extra space.
Solution:b Two
Have two stacks S1 and S2.
For Enqueue, perform push on S1.
For Dequeue, if S2 is empty pop all the elements from S1 and push it to S2. The last element you popped from S1 is an element to be dequeued. If S2 is not empty, then pop the top element in it.