الجامعة اللبنانية
كلية الاعلام
الفرع الاول

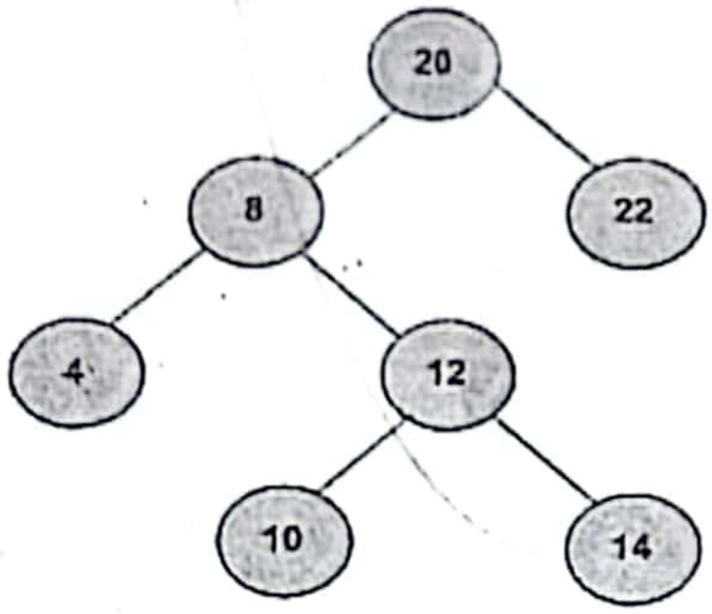
Université Libanaise
Faculté d'Information

امتحان نهائي 2022- 2023

الدورة الاولى

المرحلة: اجازة	المادة: Data Structures and Algorithms
السنة المنهجية: الثانية	المدة: ساعتين
الاختصاص: علم البيانات	أستاذ المادة: Dr. Mageda Sharafeddin

1- Given values of two nodes in a Binary Search Tree, write an algorithm to find the Lowest Common Ancestor (LCA). You may assume that both values exist in the tree.
The function prototype should be as follows: node *LCA(node* root, int n1, int n2) n1 and n2 are two values in present nodes in the tree with given "root" node. For example, consider the tree below, LCA of 10 and 14 is 12 and LCA of 8 and 14 is 8. (30pts)



2- Find contiguous numbers of different sign (35pts): We are given a list A of integers of such that·

i) len(A)>=2

ii) The first and last elements of A have different signs

iii) All the elements of A are non-zero

Given A, we would like to find any two contiguous integers x and y in A of different signs. Note that the existence x and y is guaranteed by (i), (ii), and (iii). (The argument is by contradiction:

If all contiguous integers in A are of the same signs, then the first and second are of the same sign, and the second and the third are of the same sign, etc. Thus, the first and last elements are of the same signs, which contradicts (ii)).

Write a function signChange(A), which given a list of integers satisfying (i), (ii), and (iii), returns a tuple (x,y), where x and y are contiguous integers A of different signs (see the below test cases).

You are not asked to check if Conditions (i), (ii), and (iii) are satisfied; assume that they hold. If there are more than one pairs of contiguous integers of different signs (e.g., last test case below), any one of them is a valid answer. **The solution must run in O(log n)** time. (Hint: use recursion).

Test program/Output:

```
print(signChange([ 2, -1]))                    -> (2,-1)
print(signChange([2, 3, -1]))                  -> (3, -1)
print(signChange([2, 3, -5, 2,-1]))            -> (3, -5)
print(signChange([-2,-3, -5, 2,10]))           -> (-5, 2)
print(signChange([-2,-3,-5, -20, 2,-25,10]))   -> (-20, 2)
```

3- You are given as input an array that comprises a non-increasing sequence of elements in the first quarter of the array, followed by a non-decreasing sequence of elements in the last three quarters of the array. Find the most efficient way to sort the array. Write the code and explain it. Zero points for any sub optimal solution. (35pts)