CONCORDIA UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE AND SOFTWARE ENGINEERING

COMP 6651: Algorithm Design Techniques Winter 2022

Quiz # 10

First Name	**Last Name**	**ID#**

Question 1

1. Which one(s) of the following array elements represents a binary min heap?
   A. 12 10 8 25 14 17 **B.** 8 10 12 25 14 17 **C.** 8 10 12 14 21 32 19
   D. 25 17 14 12 10 8 **E.** 14 17 25 10 12 8
   Your choice: A ☐ B ☒ C ☒ D ☐ E ☐
   Answer. 1 point. A tree is min heap when data at every node in the tree is smaller than or equal to it's children' s data. So, only 8 10 12 25 14 17 and 8 10 12 25 14 17 generates required tree.

2. In a binary min heap containing n elements, the largest element can be found in ..... time.
   A. $O(n)$ **B.** $O(n \log n)$ **C.** $O(\log n)$ **D.** $O(1)$
   Your choice: A ☒ B ☐ C ☐ D ☐
   Answer. 1 point. In min heap the smallest is located at the root and the largest elements are located at the leaf nodes. So, all leaf nodes need to be checked to find the largest element. Thus, worst case time will be $O(n)$.

3. Describe an $O(n)$ algorithm for building a min-heap.

1 point

```
BUILD-MAX-HEAP(A)
1.        heap-size[A] ← length[A]
2.        for $i$ ← ⌊length[A]/2⌋ downto 1
3.        do MAX-HEAPIFY(A,i)
```

2 points

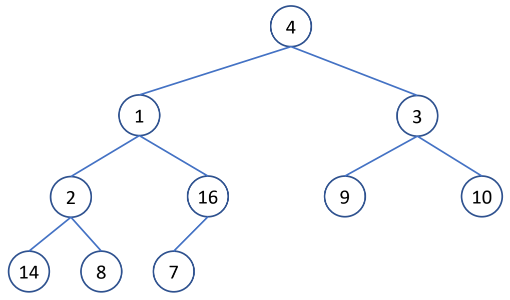
```
MAX-HEAPIFY(A, i)
1.     $\ell \leftarrow$ LEFT($i$)
2.     $r \leftarrow$ RIGHT($i$)
3.     if $\ell \leq$ heap-size[A] and A[$\ell$] >A[$i$]
4.         then largest ← $\ell$
5.         else largest ← $i$
6.     if $r \leq$ heap-size[A] and A[$r$] > $A$[largest]
7.         then largest ← $r$
8.     if largest $\neq i$
9.         then exchange A[$i$] ↔ A[largest]
10.            MAX-HEAPIFY (A, largest).
```

4. Illustrate the first steps of your algorithm on the following array:

4	1	3	2	16	9	10	14	8	7

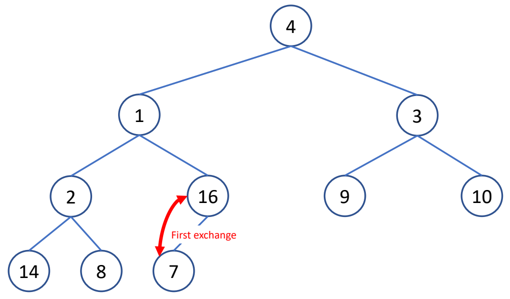
   - Draw the first tree structure that is derived directly from the array

1 point



- Explain clearly the first swap of elements

1 point



Question 2

You need to design a branch-and-bound for the TSP problem. As seen in the lecture, you first need to propose algorithms in order to compute lower and upper bounds. Propose an algorithm for computing a lower bound: it is required to write a pseudo-code (an example on a toy example is not an algorithm)

2 points

For each city $i, 1 \leq i \leq n$, find the sum s_i of the distances from city i to the two nearest cities
Compute the sum s of these n numbers
Divide the result by 2
If all distances are integer values, round up the result to the nearest integer: LB $= \lceil s/2 \rceil$