A zero mark will be given if you copy someone else's work or you let someone copy your work.

COMP122/22-14 Data Structures and Algorithms — 0 – 40 points

Heaps

2022-03-14

Due Date — 2022-03-21

Class Code

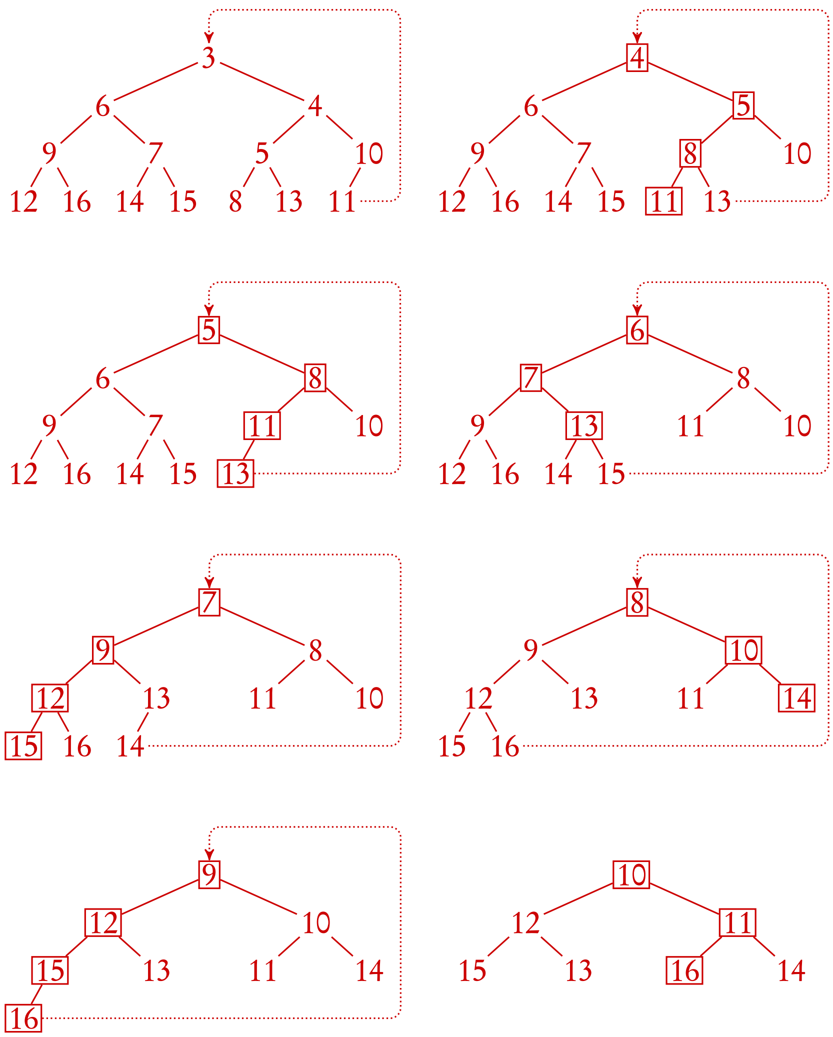
Student No. DO NOT WRITE YOUR NAME

1. For an array-based list a of 14 integers, initially the items of a, from $a[0]$ to $a[13]$ are:

$$3, 6, 4, 9, 7, 5, 10, 12, 16, 14, 15, 8, 13, 11.$$

It is obvious that a represents a complete binary tree which is also a heap, where every parent is less than its children.

We remove the minimum items 7 times successively from the heap, then a has 7 items left, draw the initial heap and the heaps after each removal. (8 points)



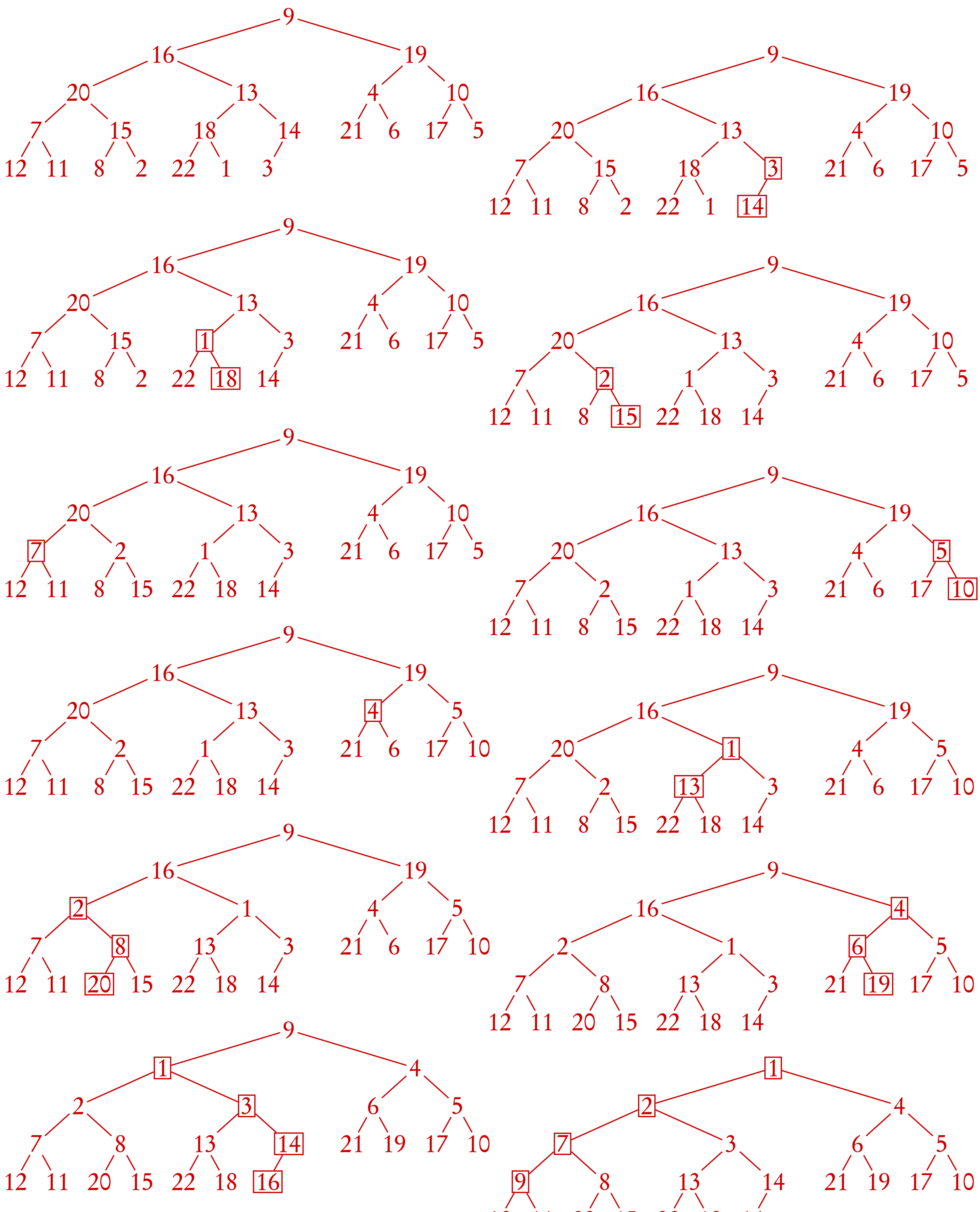
(1)

2. For an array-based list a of 22 integers, the initial items of a, from $a[0]$ to $a[21]$ are:

$$9, 16, 19, 20, 13, 4, 10, 7, 15, 18, 14, 21, 6, 17, 5, 12, 11, 8, 2, 22, 1, 3.$$

We require in the heap that every parent is less than its children.

If a is heapified by the sift-down algorithm, draw the initial tree and the trees after the sifting-down of each element. (12 points)



(cont'd on next page)

(2)

3. By converting the tail-recursion on Page 8 of Lesson 14 to a loop, write a non-recursive *sift_down_i*(*a*, *x*) function to sift-down element x to a proper location in the complete binary tree stored as an array-based list a. (10 points)

```
def sift_down_i(a, x):
```

```
    n = len(a)
    i = 0                                   1
    while True:                             1
        j = 2*i+1                           1
        if j >= n:
            break                           2
        if j+1 < n and not a[j] <= a[j+1]:
            j += 1                          1
        if x <= a[j]:
            break                           2
        a[i] = a[j]
        i = j                               1
    a[i] = x                                1
```

(3)

4. Starting from an arbitrary integer i, if you set the element of a node p in a binary tree T to the preorder *rank* of p, you always get a heap. (The rank of a node is the relative position of the node in the traversal sequence, that is, the first node visited has rank i, the second has rank $i+1$, and so on.)

   Prove by mathematical induction that the above statement is correct.

   Hint: you induct on the size (number of nodes) of T. (10 points)

Let n be the number of nodes in T, n_l the number of nodes in the left subtree and n_r the number of nodes in the right subtree.

Base case: when $n = 0$, T is empty, and is obviously a heap.

Induction step: when $n \geqslant 1$, there is the root node. By the preorder traversal, the rank of the root node is i, the ranks of the left subtree start from $i+1$ and the ranks of the right subtree start from $i+1+n_l$. Thus, the root node satisfies the heap property. Obviously, we have $n_l, n_r < n$, by induction hypothesis, the left and right subtrees are heaps, therefore, all the nodes in T, including the root node, satisfy the heap property. Thus, T is a heap.

(4)