Université d'Ottawa
Faculté de génie

École de science informatique et de génie électrique (SIGE)

uOttawa
L'Université canadienne
Canada's university

University of Ottawa
Faculty of Engineering

School of Electrical Engineering and Computer Science (EECS)

CSI 2110
Data Structures and Algorithms

Midterm Exam (30%) - Solution

Date: Sunday Oct. 27, 2019

1. This is a closed book exam
2. None of calculators and cellular phone is allowed.
3. You have 2 hours to finish.
4. Write your answers on this questionnaire.
5. Peeking at your neighbors' work may be a cause for expulsion from the exam.
6. At the end of the exam, when time is up:
   - Stop working and turn your exam upside down.
   - Remain silent.
   - Do not move or speak until all exams have been picked up, and a TA or the Professor gives the go-ahead to leave.

-

	Marks (out of 100)
Name : _____ Student's number : _____ Signature : _____ -	

☺ Good Luck! ☺

[Question 1] (6 marks)

Fill the blanks with the best possible big-oh complexity:

(1) $n^4 (1+2n)/n^2 + 1039*n^3 + 1039*n^2$ is O(n^3)

$= n^2(1 + 2n) + 1039n^3 + 1039n^2$
$= 1041n^3 + 1040n^2$
$= O(n^3)$

(2) $-n + \log_2 (10)^n - \log_2(7*n)$ is O(n)

$= -n + nlog_2 10 - log_2 7 - log_2 n$
$= n(log_2 10 - 1) - log_2 n - log_2 7$
$= O(n)$

(3) $n^2 + \sum_{i = 10, 11, .., n} (i^2)$ (for n > 10) is O(n^3)

$= n^2 + 10^2 + 11^2 + \cdots + n^2$
$\leq n^2 + n^2 + \cdots + n^2$
$\leq n * n^2$
$= O(n^3)$

[Question 2] (5 marks)

The given pseudo-code below takes an array A = {2, 6, 3, 5, 9, 1, 34, 92, 9} of n positive integers as input and uses an initially empty stack S as an internal variable:

```
t ← 0
for (i=0; i<n; i++) do    # some questionnaire are indicated by n-1 loops
        if (A[i] mod 2) = 1 then
            S.push(A[i])

while S.size() > 1 do {
    k = S.pop()
    t ← t + k*k }

return t
```

(1) (3 Marks) What is the output of the above algorithm?

Solution:

Case 1 (with n loops): output: 188. After 'for' loop, the odd numbers are pushed into S in the order: {3, 5, 9, 1, 9}. 'while' loop will return the top k-1 elements of S. k is the size of S. t will accumulate the squared value of element (i.e. $t = 9^2 + 1^2 + 9^2 + 5^2 = 188$).

Case 2 (with n-1 loops): output: 107. After 'for' loop, the odd numbers are pushed into S in the order: {3, 5, 9, 1}. 'while' loop will return the top k-1 elements of S. k is the size of S. t will accumulate the squared value of element (i.e. $t = 1^2 + 9^2 + 5^2 = 107$).

(2) (2 Marks) What is the running time of the above algorithm in terms of n, the number of integers in a non-empty array A?

a) $O(n^2)$
b) $O(n \log n)$
c) $O(n)$
d) $O(n^3)$
e) $O(1)$

[Question 3] (2 marks)

What is the worst-case running time of the following algorithm for input being a bi-dimensional n x n array A, for an arbitrary integer?

```
sum = 0;
for (i=0; i<n; i++) do
        for (j=0; j<100; j++) do
                for (k=0; k< i*j ; k++) do
                        sum = sum + A[i][j]*k
return sum
```

Solution: $O(n^2)$.
For each i loop, j will loop for 100 times, and k will loop for n + 2n + … + 99n = 4950n times. Total running time: $O(n^2)$ for i->n, j->100, k->4950n

[Question 4] (3 marks)

Prove that $f(n) = \sum_{i=1,\ldots,n} (n^2 - i)$ is $O(n^3)$ by using the definition of big-Oh.

Solution:

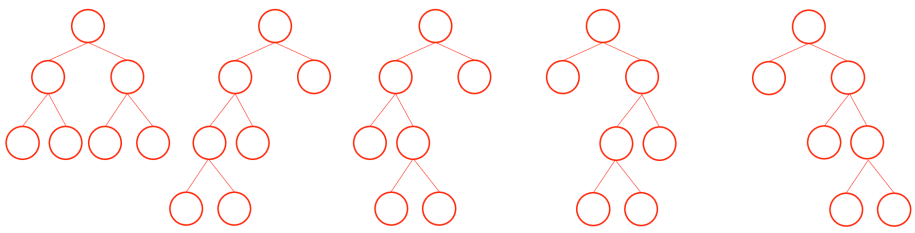
$$
\begin{aligned}
f(n) &= \sum_{i=1}^{n} (n^2 - i) \\
&= \sum_{i=1}^{n} n^2 - \sum_{i=1}^{n} i \\
&= n * n^2 - \frac{(1+n) * n}{2} \\
&= n^3 - \frac{1}{2}n^2 - \frac{1}{2}n \\
&\leq n^3 \\
&\leq Cn^3
\end{aligned}
$$

For all $C \geq 1, and\ n > n_0 \geq 0$. So f(n) is $O(n^3)$

[Question 5] (4 marks)

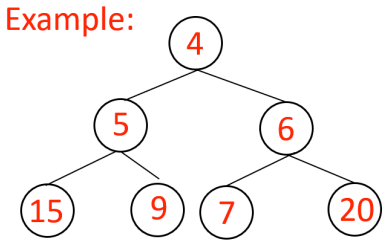
Draw all possible full binary trees with 7 nodes.

Solution:



[Question 6] (6 marks)

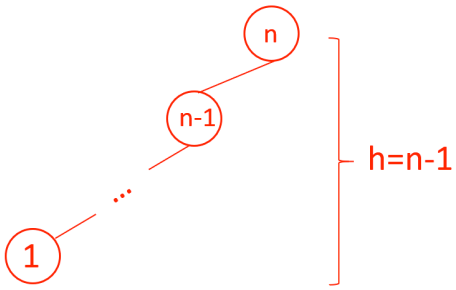
(a) The array that represents a min-heap is always in descending order.
[TRUE / FALSE]



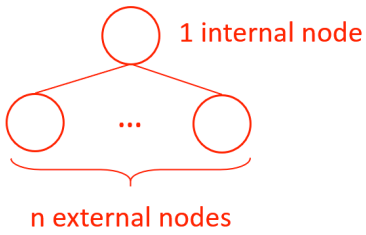
Array: [4,5,6,15,9,7,20]
The array needs to be sorted

(b) An element with a maximum key in a min-heap is always external.
[TRUE / FALSE]
Because in min-heap, key(parent) should always smaller or equal to key(child). The maximum key can only appear on the last layer, which contains leaves.

(c) The height of a binary tree of n nodes is always $O(\log n)$
[TRUE / FALSE]
Height could be between O(log n) and O(n) according to the structure. An extreme example is shown as:

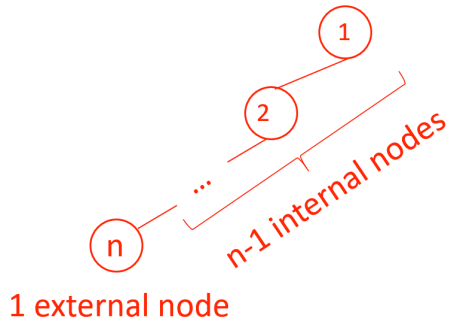


(d) The number of external nodes of a tree is always one more than the number of internal nodes
[TRUE / FALSE]
For an arbitrary tree, the external nodes could far more than internal nodes. See example:

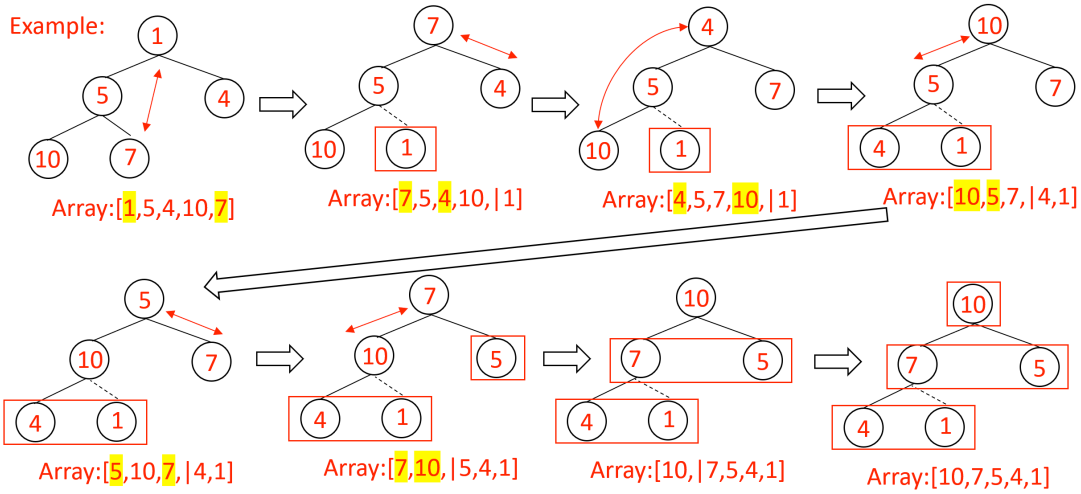


(e) The number of external nodes of a binary tree is always one more than the number of internal nodes
[TRUE / FALSE]

For an arbitrary binary tree, the number of internal nodes can be greater than external nodes. See example:



(f) With a given sequence, which heap must be used to sort it in descending order? [Max-heap / Min-heap]



[Question 7] (6 marks)

What is cost in Big-Oh in worst case to search its minimum element for each data structure below?

(a) Max-heap

Solution: O(n). Need to check all elements, because there is no relation in value between left and right child.

(b) Min-heap

Solution: O(1). The root element.

(c) Sorted sequence in descending order in singly linked list

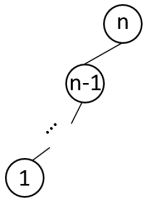
Solution: O(n). Elements in a singly linked list should be visited from the front to the tail.

(d) Unsorted sequence in doubly linked list implementation

Solution: O(n). in unsorted sequence, each element need to be checked.

(e) Binary search tree

Solution: O(n). See the worst case below:

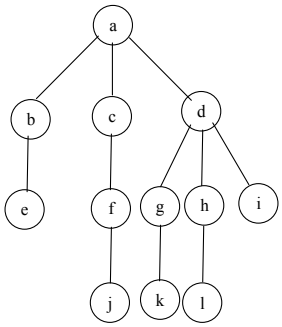


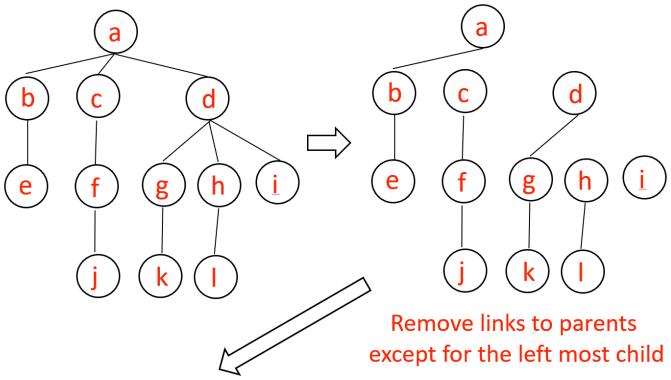
(f) Array sorted in ascending order

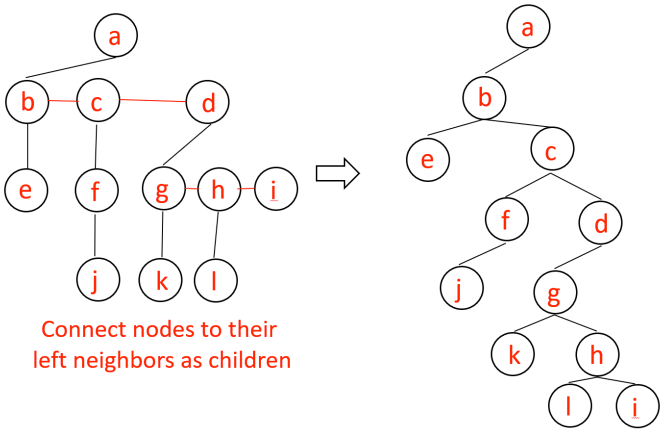
Solution: O(1). The first element.

[Question 8] (4 marks)

Consider a general tree T below. Convert it to be the binary tree T' which represents T. Draw T'.







[Question 9] (2 marks)

How many nodes does a complete binary tree of height h have?

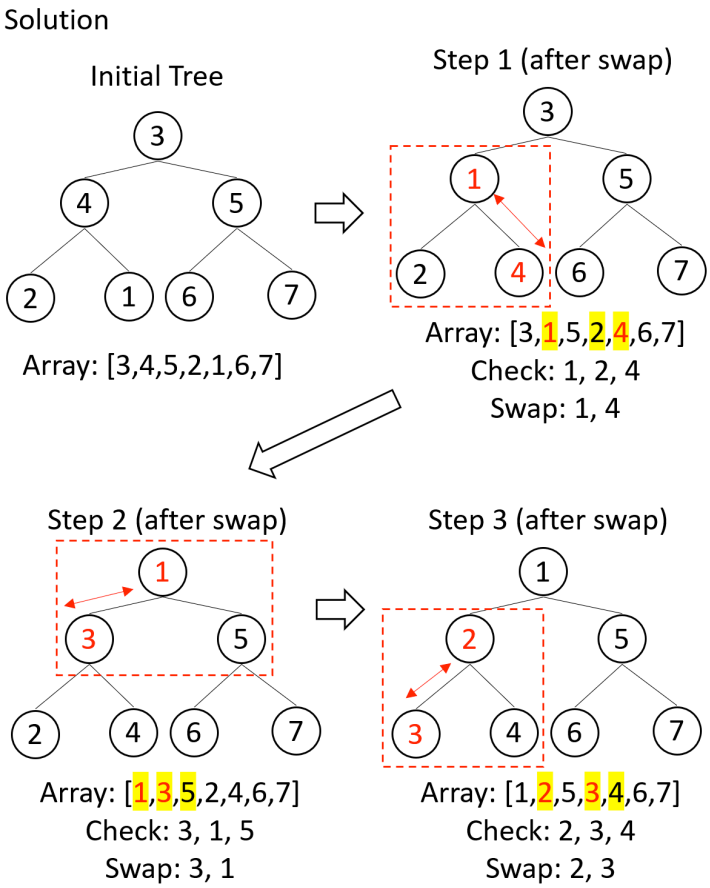
a. 2^h nodes
b. 2^{h+1} nodes
c. 2^{h-1} nodes
d. between 2^{h-1} and 2^h nodes
e. between 2^h and 2^{h+1} nodes

Solution: number of the perfect tree with h-1 height: $2^h - 1$. Min number of a complete tree with h height: $(2^h - 1) + 1$. Max number of a complete tree with h height: $(2^h - 1) + 2^h = 2^{h+1} - 1$. The range of the num: $[2^h, 2^{h+1} - 1]$. So the range is within $[2^h, 2^{h+1}]$.

[Question 10] (6 marks)

Show bottom-up heap construction with an input sequence $(3, 4, 5, 2, 1, 6, 7)$ using a Min-Heap. Construct the initial tree, where '3' is at the root position of the initial tree, i.e. an array implementation of this initial tree is $[3, 4, 5, 2, 1, 6, 7]$. And then use bottom-up heap construction to convert it to be Heap. Illustrate all the steps using trees.

Solution



[Question 11] **(3 marks)**

What is the running time of the following algorithm for input sequence A with n element? Heapify (A, n) is an algorithm to convert a sequence to be a min-Heap.

```
sum = 0;
Heapify (A, n)
for (i=0; i<n-1; i++) do
        a = removeMin(A)
        sum = sum + a;
return sum
```

Solution:

Time cost:

1	sum = 0;
nlog(n)	Heapify (A, n)
n	for (i=0; i<n-1; i++) do
log(n)	a = removeMin(A)
2 ('=', '+')	sum = sum + a;
1	return sum

removeMin return

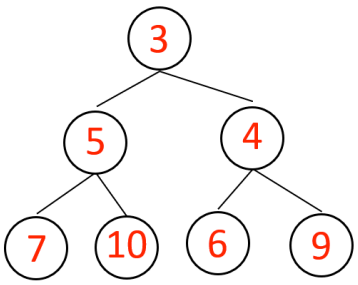
$$T(n) = 1 + nlog(n) + n * (\log(n) + 2) + 1 + 2n = 2n * logn + 4n + 2$$

sum=0 heapify for loop '+' and '=' i<n-1 and i++

[Question 12] **(4 marks)**

There is a complete binary tree in in-place implementation; [3, 5, 4, 7, 10, 6, 9].

Solution: binary tree is shown as below:



(a) What is the value of right child of a node with value 5?

Solution: 10

(b) How many internal nodes and what are values of them?

Solution: 3 with values 3, 5, 4

[Question 13] **(8 marks)**

Assume there is no dummy node.

(a) How many external nodes of a complete binary tree with number of 3567 nodes?

Solution: 1784. Number of external nodes: $n - \left\lfloor \frac{n}{2} \right\rfloor = 3567 - 1783 = 1784$

(b) What is the height of a perfect binary tree with 4095 nodes?

Solution: 11.

$$2^{h+1} - 1 = 4095$$
$$2^{h+1} = 4096$$
$$log_2(2^{h+1}) = log_2(2^{12})$$
$$h + 1 = 12$$
$$h = 11$$

(c) What is the height of a min-heap with 340 nodes?

Solution: 8.

$h = \lfloor log_2(N+1) \rfloor - 1 = 9 - 1 = 8$ or $h = \lfloor log_2 N \rfloor$

(d) What is maximum number of nodes with a max-heap with height 8?

Solution: 511.

$$n = 2^{h+1} - 1$$
$$= 2^9 - 1$$
$$= 511$$

(e) How many internal nodes are there in a max-heap with 500 nodes?

Solution: 250.

$$\left\lfloor \frac{n}{2} \right\rfloor = \frac{500}{2} = 250$$

(f) How many external nodes are there, at least, in a full binary tree with 7 nodes?

Solution: 4.

$$\frac{n+1}{2} = \frac{7+1}{2} = 4$$

(g) What is the maximum height of any binary tree with 500 nodes?

Solution: 499. The tree can be left or right skewed. So the maximum height: h=n-1.

(h) How many internal nodes are there in a perfect binary tree of height 4?

Solution: 15.

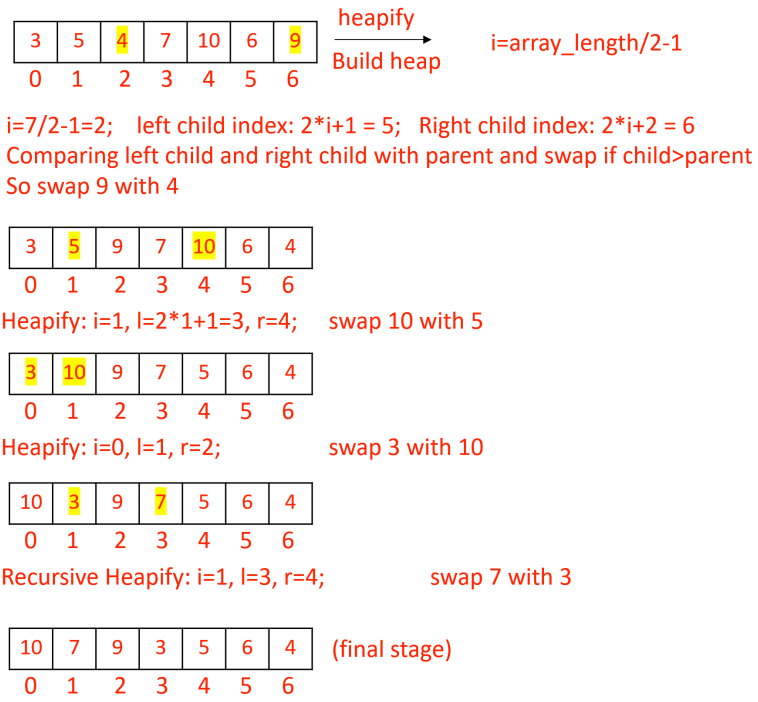
$$n = 2^h - 1 = 2^4 - 1 = 15$$

[Question 14] **(8 marks)**

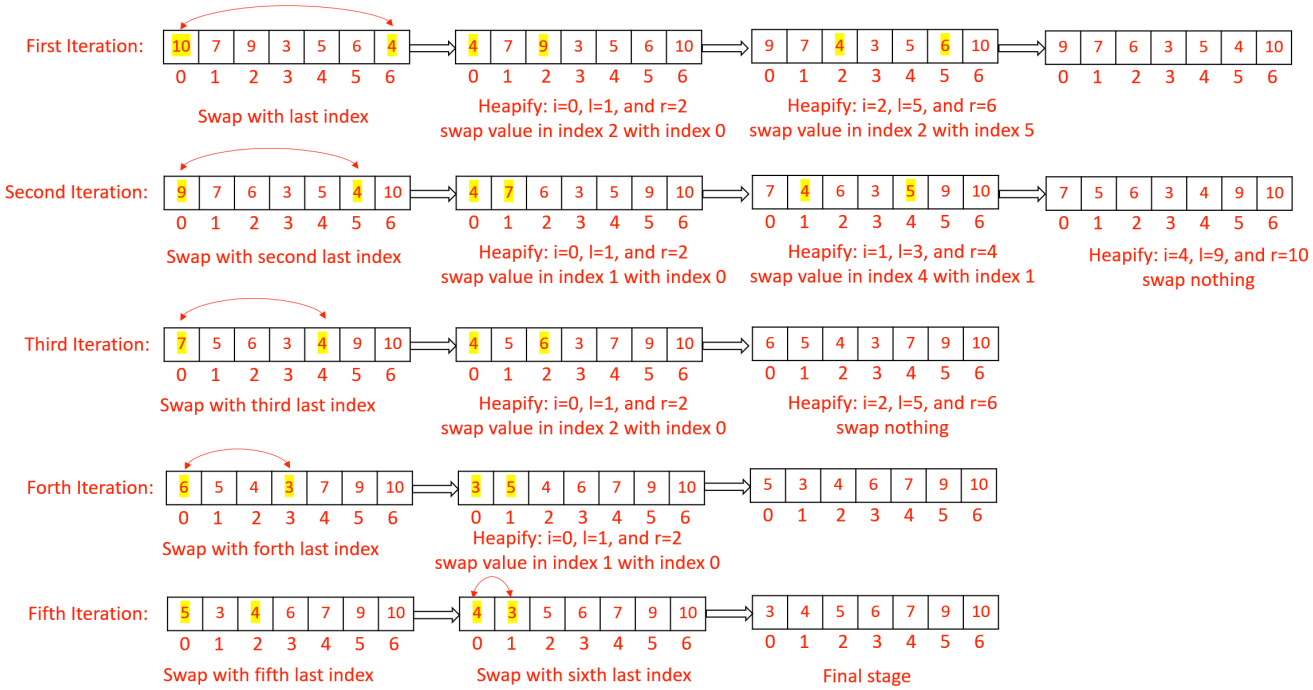
Convert an input sequence [3, 5, 4, 7, 10, 6, 9] to be a max-heap. Illustrate all the steps in-place heapsort algorithm.

Solution:

1) Construct the max-heap



2) Heap sort: we need to go through several iterations which is calculated by getting length of array. We will swap largest element with the last index and do heapify. In second iteration, we will swap with second iteration and swap with second last index and heapify and so on.

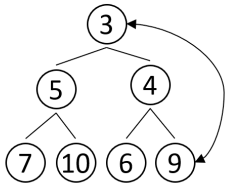


[Question 15] **(6 marks)**

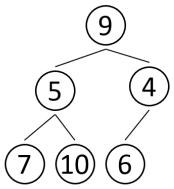
Let the tree min-heap represented with the following array [3, 5, 4, 7, 10, 6, 9]. Show, step by step, what happens if 2 minimum element removal operations (removeMin()) are performed. Give the state of the array after these removals.

Solution 1

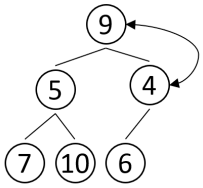
First: removeMin()



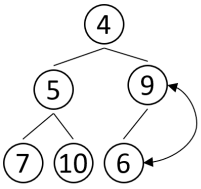
[3, 5, 4, 7, 10, 6, 9]



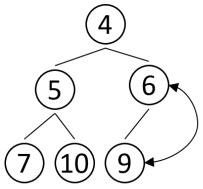
[9, 5, 4, 7, 10, 6]



[9, 5, 4, 7, 10, 6]



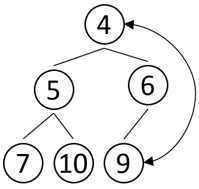
[4, 5, 9, 7, 10, 6]



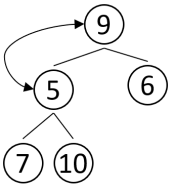
[4, 5, 6, 7, 10, 9]

Elements to be swapped are labeled by yellow color

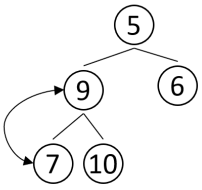
Second: removeMin()



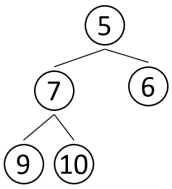
[4, 5, 6, 7, 10, 9]



[9, 5, 6, 7, 10]



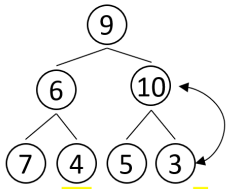
[5, 9, 6, 7, 10]



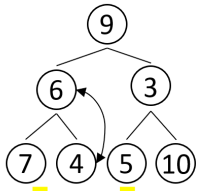
[5, 7, 6, 9, 10]

Solution 2 (insert in reverse order with bottom-up method)

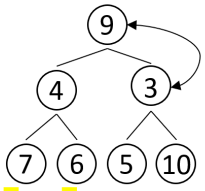
Generate Min-heap: Elements to be swapped are labeled by yellow color



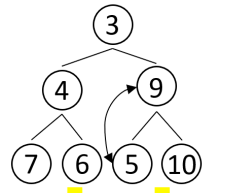
[9, 6, 10, 7, 4, 5, 3]



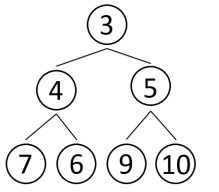
[9, 6, 3, 7, 4, 5, 10]



[9, 4, 3, 7, 6, 5, 10]

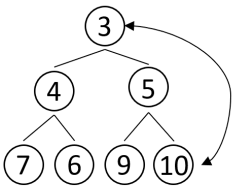


[3, 4, 9, 7, 6, 5, 10]

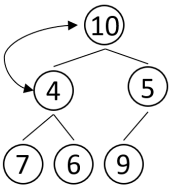


[3, 4, 5, 7, 6, 9, 10]

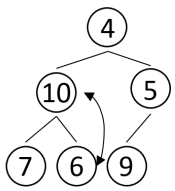
First: removeMin()



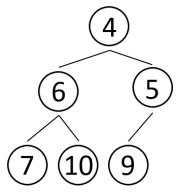
[3, 4, 5, 7, 6, 9, 10]



[10, 4, 5, 7, 6, 9]

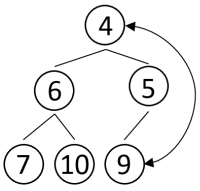


[4, 10, 5, 7, 6, 9]

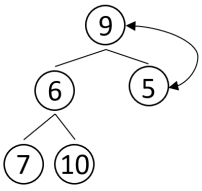


[4, 6, 5, 7, 10, 9]

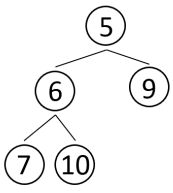
Second: removeMin()



[4, 6, 5, 7, 10, 9]



[9, 6, 5, 7, 10]

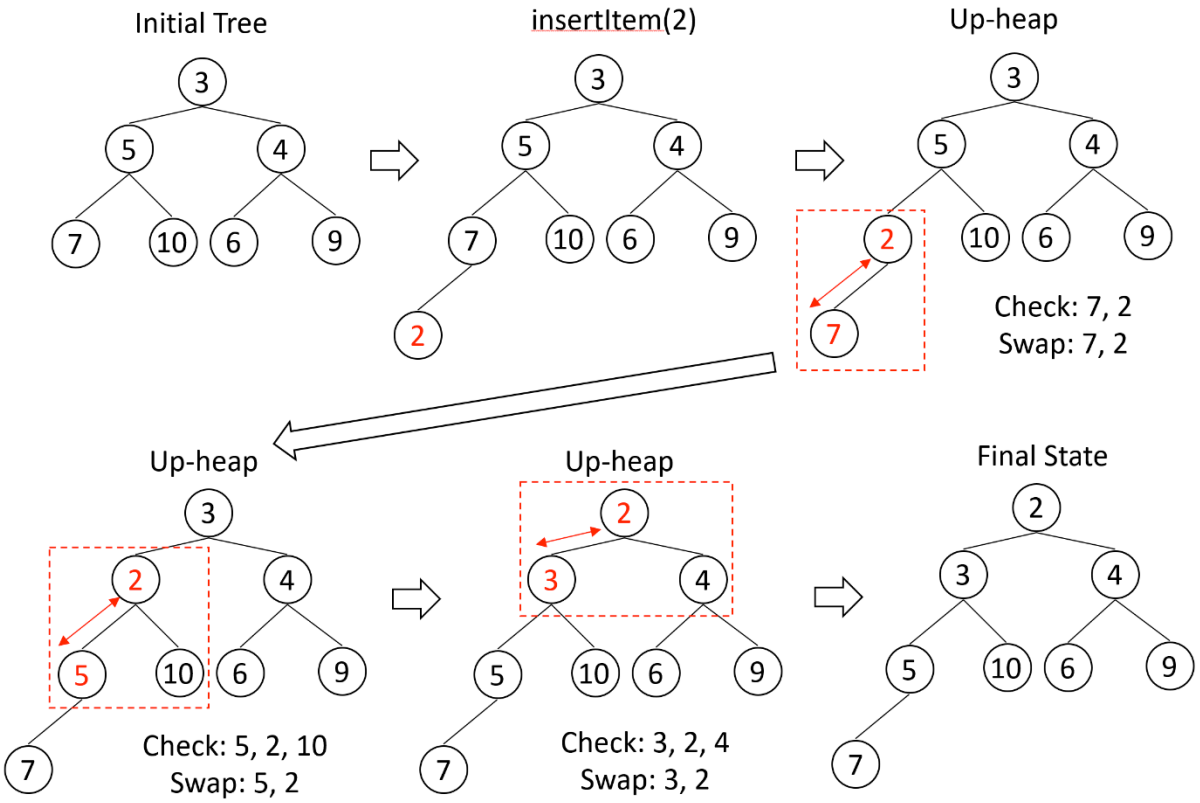


[5, 6, 9, 7, 10]

[Question 16] **(6 marks)**

A min-heap is [3, 5, 4, 7, 10, 6, 9]. Apply insertItem(2). Illustrate all the steps using trees.

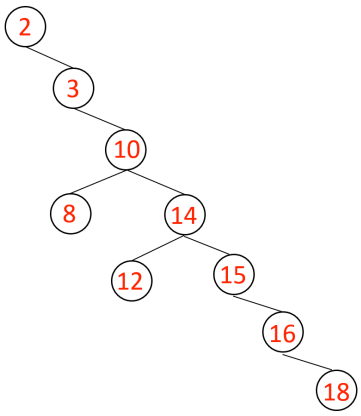
Solution



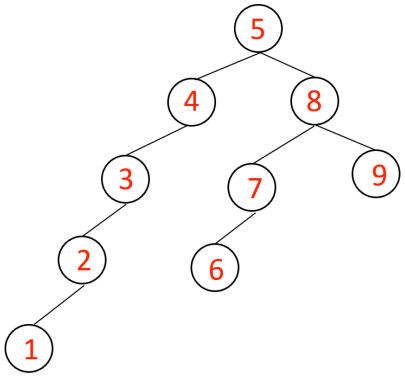
[Question 17] **(8 marks)**

(1) (6 marks) Draw three binary search trees with following sequences.

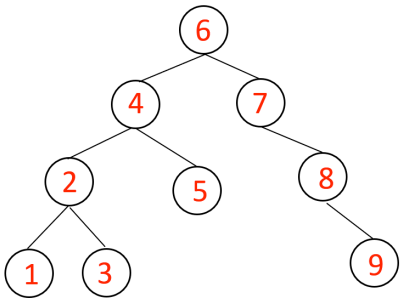
a. 2,3,10,8, 14,12,15, 16, 18



b. 5,8,4,9,3,7,2,6,1



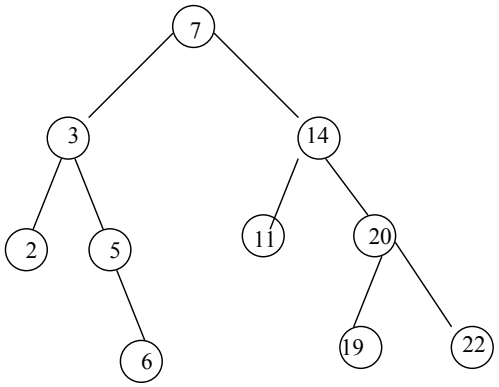
c. 6,4,7,2,5,8,3,1,9



(2) (2 marks) Which tree work best to search, add or delete an element?
Solution: c tree.

[Question 18] (6 marks)

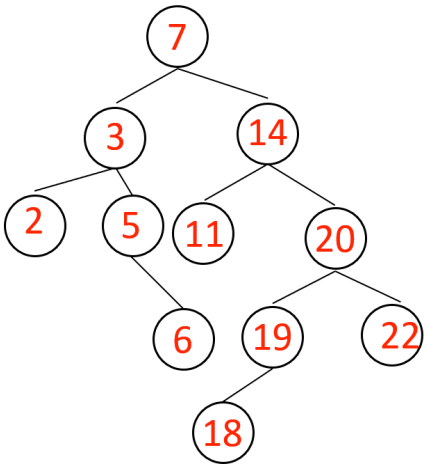
Consider a binary search tree below.



(a) How many comparisons are done when searching for 21?
Solution:
Case 1(if consider 'null' node): 5 comparisons. 7->14->20->22->null
Case 2 (if doesn't consider 'null' node): 4 comparisons. 7->14->20->22

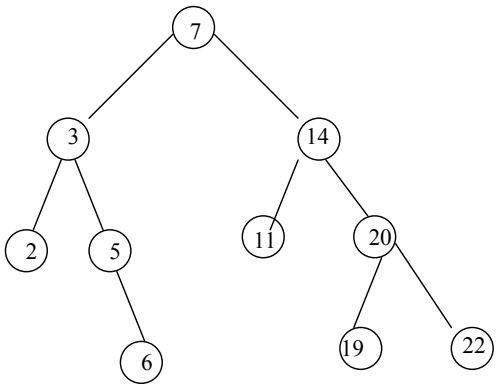
(b) How many comparisons are done when inserting a node with value 4?
Solution: 3 comparisons. 7->3->5

(c) Draw a tree after inserting a node with 18.



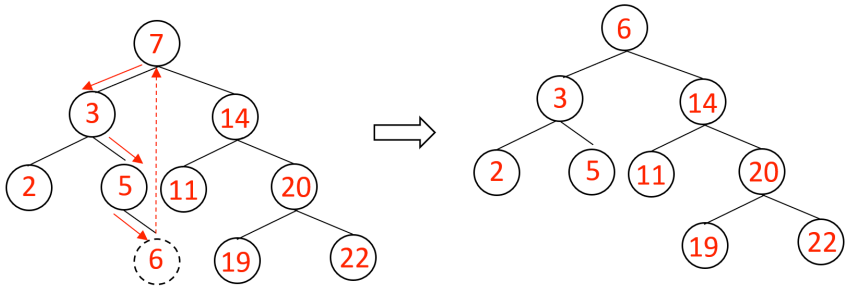
[Question 19] **(5 marks)**

Delete 7 from the following binary search tree. Draw all intermediate trees. Please use the node that PRECEDES the node in in-order traversal if needed.



Solution:

In-order traversal: 7,3,2,5,6,14,11,20,19,22



[Question 20] **(2 marks)**

Which algorithm can we use to create a sorted sequence from a binary search tree? Choose one.

(a) Algorithm mySorting(T,v)
```
    visit(v)
    if v is internal:
        mySorting (T,T.LeftChild(v))
        mySorting (T,T.RightChild(v))
```

(b) Algorithm mySorting(T,v)
```
    if v is internal:
        mySorting (T,T.LeftChild(v))
        mySorting (T,T.RightChild(v))
    visit(v)
```

(c) Algorithm mySorting(T,v)
```
    if v is internal:
        mySorting (T,T.LeftChild(v))
    visit(v)
    if v is internal:
        mySorting (T,T.RightChild(v))
```

(d) Algorithm mySorting(T,v)
```
  For (I = 0; I < n)
    removeMin(v);
```

(e) Algorithm mySorting(T,v)
```
  For (I = 0; I < n)
    removeMax(v);
```

Solution: C. in-order traversal of binary search tree outputs a sorted sequence.