Question 1

Suppose you have the following runtime data for an algorithm. What complexity class do they indicate?

Input size	Seconds
1000	5
2000	19
4000	78
8000	313
16000	1251

Answer :- O(N^2)

Explanation :- Every time the input value doubles, runtime increases 4 times. Which means,

2 ➔ 2^2

N ➔ N^2

Question 2

Consider the following code fragment. Based on its structure, what is its complexity class?

```
int m = 0;
for ( int i = 0; i < n ; i ++)
  for ( int j = 0; j < n ; j ++)
    for ( int k = 0; k < i ;    k ++)
      m += i ;
```

Answer :- O(N^3)

Explanation:- First and second for loops directly depend on “N” to determine the number of loops. The third for loop depends on the value “i” to determine the number loops. “i” depends on “N”. Which means the third for loop indirectly depends on “N” to determine the number of loops. This the runtime of all three for loops increases as the value of “N” Increases. Which means all three for loops has a time complexity of O(N) individually. Because of this the total time complexity is O(N^3).

Question 3

Suppose we use Binary Search to find the value 11 on the following array. Which array elements get checked, in which order, and why?

index	0	1	2	3	4	5	6
value	2	3	5	7	11	13	17

Answer :- 7, 13, 11

Explanation :- We first check with the middle index of the array, which is value “7” in index 3. Since 11 is larger than “7” we consider the left sub array next. In the left subarray the middle value is “13” located in index 5. Since 11 is smaller than 13 we consider the right sub array of index 5. In this subarray the only value is “11”. So we found 11 :).

Question 4

Suppose we run Selection Sort to sort the following array in increasing order. What does the array look like after the first 3 iterations, and why?

index	0	1	2	3	4	5	6
value	13	3	17	5	2	7	11

Answer:- 7 3 2 5 11 13 17

Explanation:- First we consider the 6th index(11) and swap it with the largest value in the unsorted part of the array. Since the largest value in the unsorted part is 17, we swap 11 and 17.

13 3 11 5 2 7 17

Then we consider the 5th index(7) and do the same thing. Since the largest value in the unsorted part is 13, We swap 13 and 7.

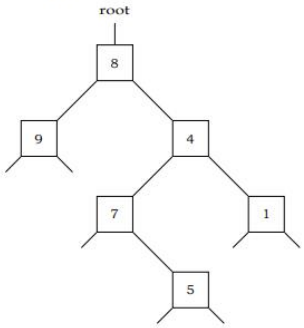
7 3 11 5 2 13 17

Then we do the same thing with the 4th index(2). And swap 2 and 11.

7 3 2 5 11 13 17

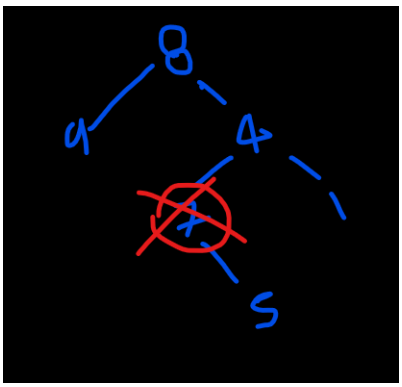
Question 5

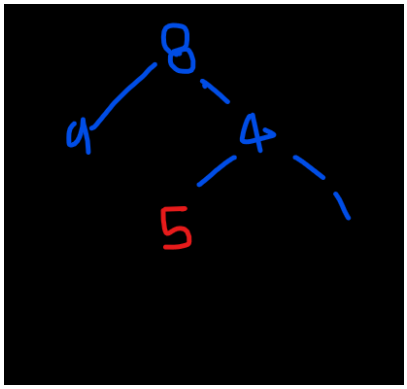
Suppose we remove the value 7 from the following binary search tree. What does the resulting tree look like?



To enter your answer, write the contents of the tree one level at a time, using "-" to indicate missing nodes.
For example, the above tree would be written

```
8
9 4
- - 7 1
- - - - - 5 - -
```





Answer :-

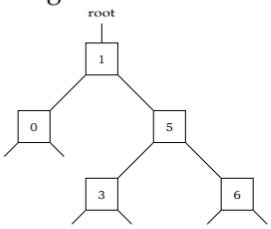
8

9 4

- - 5 1

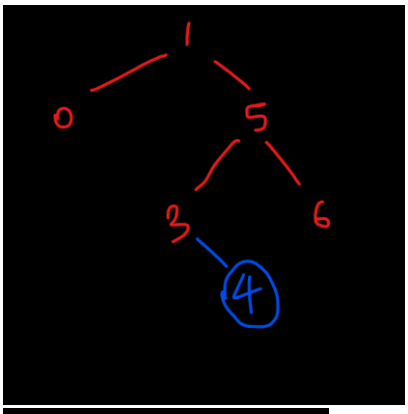
Question 6

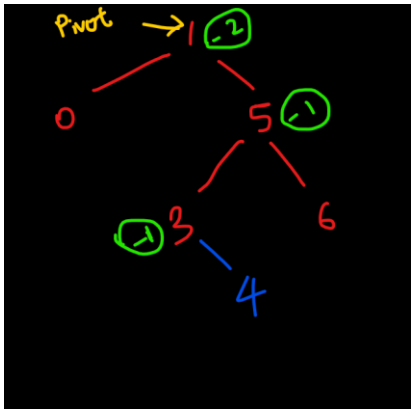
Suppose we insert the value 4 in the following AVL tree, and re-balance it. What does the resulting tree look like?

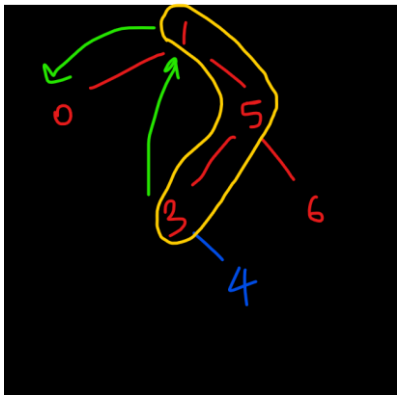


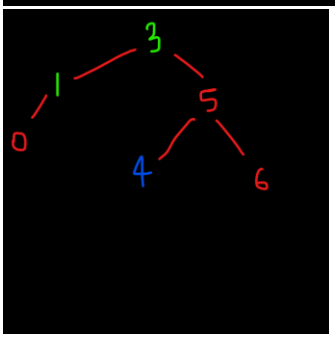
To enter your answer, write the contents of the tree one level at a time, using "–" to indicate missing nodes.
For example, the above tree would be written

1
0 5
– – 3 6









Answer:-

3

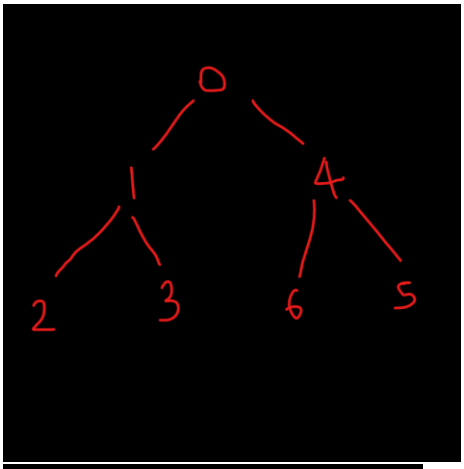
1 5

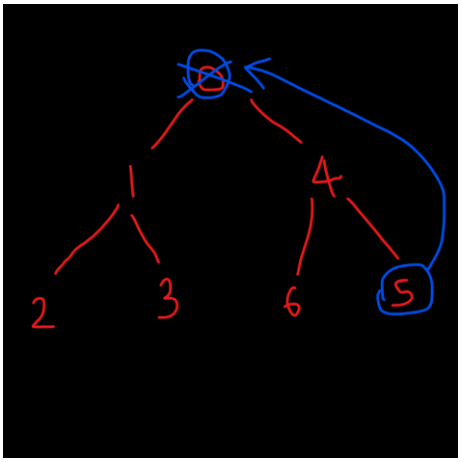
0 - 4 6

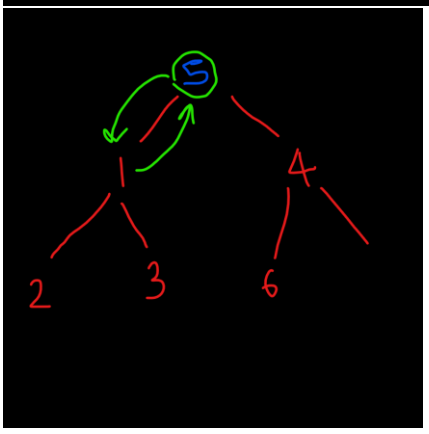
Question 7

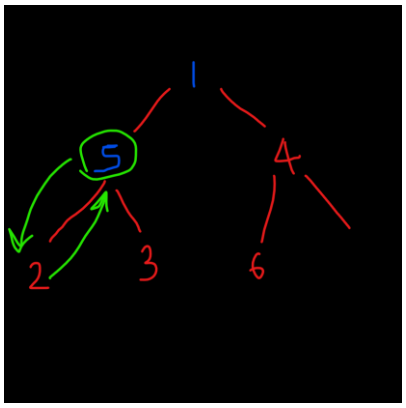
Consider the following array representation of a Min-Heap. How does this change after extracting the minimum element?

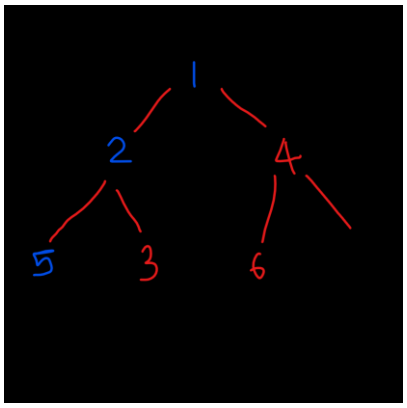
index	0	1	2	3	4	5	6
value	0	1	4	2	3	6	5











Answer:-

Index 0 1 2 3 4 5

Value 1 2 4 5 3 6

Question 8

Consider the undirected graph given by

V = {a ,b ,c, d, e, f}

E = { {a,b}, {a,c}, {a,e}, {a,f}, {b,c}, {b,d}, {b,f}, {c,d}, {d,e}, {d,f}, {e,f} }

What is an Euler walk? Does this graph have one? If yes, give an example. If not, why not?

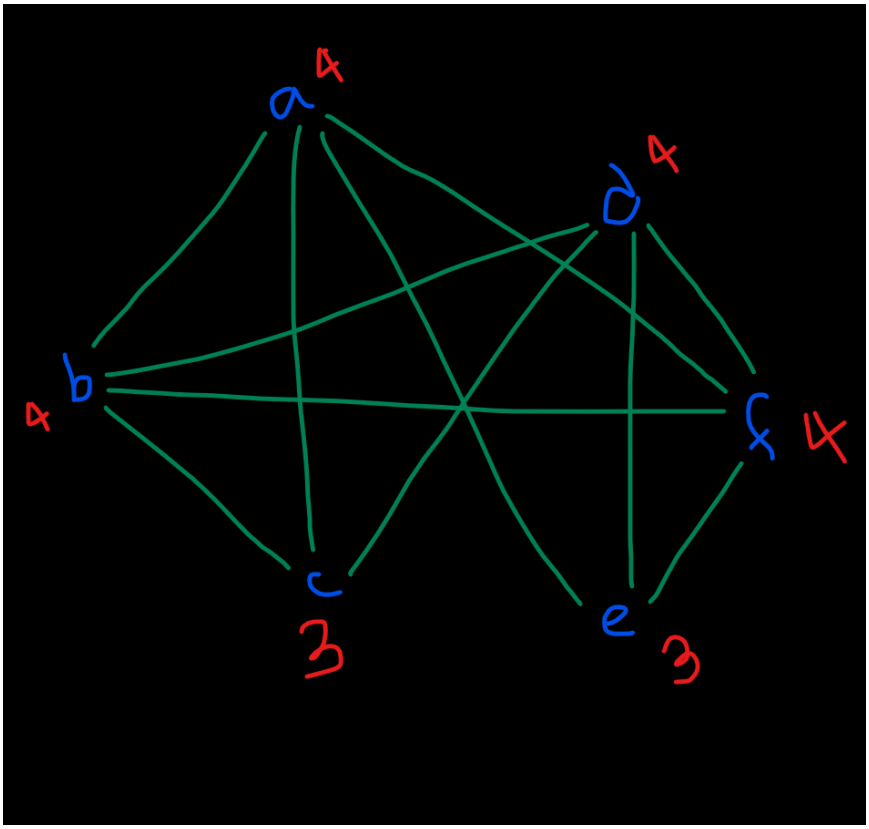
Answer: - Euler walk is a walk that uses every edge exactly once.

For an Euler walk(Euler path) to exist in an undirected graph,

i) Every node in the graph should have even degrees.

OR

ii) Exactly to nodes should have odd degrees (rest will have even).

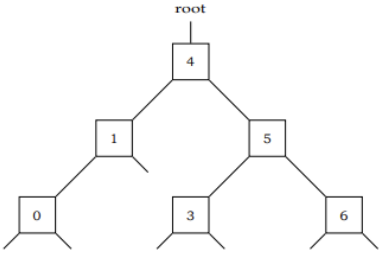


The given graph satisfies the 2nd condition. Exactly 2 nodes have odd degrees(c, e). Because of that, this graph has a Euler walk.

I am too lazy to find the path ;)).

Question 9

Suppose we use a pre-order traversal to output the following tree. What does this traversal do? What is the resulting output?



Answer :-

Pre order traversal first outputs the Root. Then the left then applies pre order traversal to the left child recursively. And then applies to the right child recursively.

Pre order traversal ➔Root, Left child, Right child.

Output is 4,1,0,5,3,6

Question 10

For this question, consider the Booean satisfiability problem:

Input: A Boolean formula using n variables Output: true, if there is some combination of variable values that makes the formula true; otherwise false.

What would a brute force algorithm for this problem do? What is its complexity in terms of n?

Not sure about this answer.

Answer :- As a brute force algorithm, we can save all the TRUE combinations in one array and all the FALSE combinations in another array. These will be arrays of arrays. Because they contain an array of inputs in each index.

With two non-connected for loops, we can check in which array the combination is in. As the number of variables increases, the number of combinations increases. Because of this the number of elements in our TRUE and FALSE arrays will also increase. Because of this the time complexity of each array will be O(N). Since they aren't nested, the total time complexity will also be O(N).