✓ Question 1

10 / 10

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

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

Your Answer
The algorithm is quadratic. The input size doubles between inputs and the runtimes grow by a factor of approximately 4. Since 4 = 2^2, the complexity class is O(n^2).

✓ Question 2

10 / 10

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;
```

Your Answer
The algorithm is cubic. The variable i runs up to n, variable j runs up to n and variable k runs up to i which in turn runs up to n. Since all of the nested has an upper bound which grows as n, the overall complexity is O(n^3).

✓ Question 3

10 / 10

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

Your Answer

The elements that get checked are a[3], a[5], a[4].
a[3] is the middle element and is smaller than 7, so we restrict the range a[4] to a[6].
a[5] is the middle of the remaining range, and is greater than 11, after restricting again, a[4] is the middle of the remaining range, and it matches the target value.

✓ Question 4

10 / 10

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

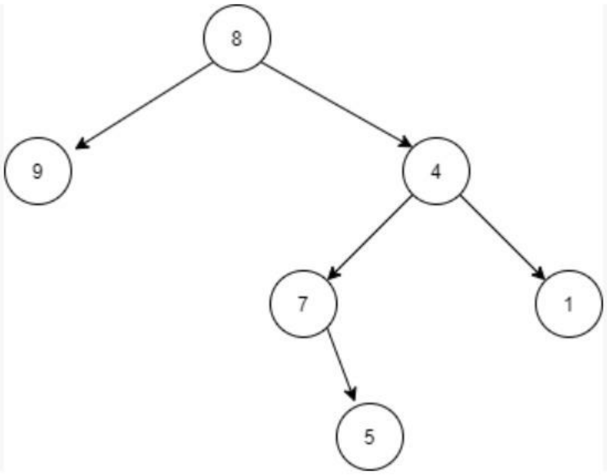
Your Answer

In each iteration, we go through the unsorted section, finding the maximum value in the unsorted section and move it to the end of the unsorted section which now becomes the sorted section.
In the first iteration, the result is (13, 3, 11, 5, 2, 7, 17)
In the second iteration, the result in (7, 3, 11, 5, 2, 13, 17)
In the third iteration, the result is (7, 3, 2, 5, 11, 13, 17)

Question 5

10 / 10

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 - -

Your Answer

8

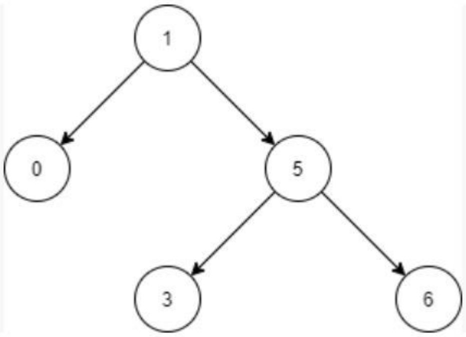
9 4

- - 1 5

Question 6

10 / 10

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

Your Answer

0

1 5

0 - 4 6

✓ Question 7

10 / 10

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

Your Answer

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

✓ Question 8

10 / 10

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}}

Write down the adjacency list representation of this graph.

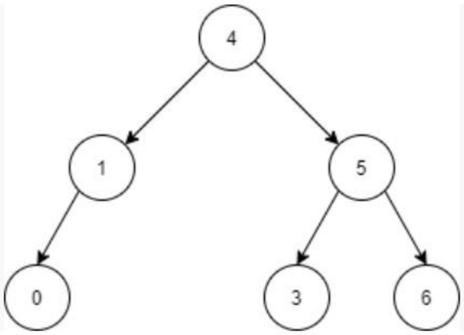
Your Answer

a : b , c , e , f
b : f, a ,c ,d
c : d, a, b
d : e, f, c, b
e : f, d, a
f : a, e, d, b

Question 9

10 / 10

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



Your Answer
A pre-order traversal outputs the graph recursively by first printing the node, then traversing the left subtree and printing it, then traversing the right subtree and printing it. The resulting output is
4 1 0 5 3 6

Question 10

2 / 10

For this question, consider the Boolean 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 the complexity of this algorithm in terms of n?

Your Answer
A brute force algorithm first checks for possible combinations in the array of variable values, by checking more than one variable. It checks each possible value by comparing it to all values in the array. Since we will have to iterate through the whole array twice we will also check the n values twice as the brute force algorithm will compare values one by one to check if it true or false. This will make the complexity of the algorithm of O(N^2).