3013 - Algorithms
Spring 2022
Exam 2
March 31 2022

Name: _____

READ THESE INSTRUCTIONS

- Use pencil only
- Initial top right corner of all pages (except the first one).
- Do NOT write on the exam.
- Do not remove the staple from your exam.
- Do not crumple or fold your exam.
- **Answering Questions:**
  - Handwriting that is illegible (messy, small, not straight) will lose points.
  - Indentation matters. Keep code aligned correctly for any answers that require code.
  - Answer all questions on the answer sheet(s) provided, not on the exam.
  - Make sure you write your answers on the answer sheets IN ORDER.
  - Do not cram answers together. Leave space.
- Help me ... help you!

Failure to comply will result in loss of letter grade

Grade Table (don't write on it)

Question	Points	Score
1	21	
2	20	
3	15	
4	19	
5	15	
6	10	
7	15	
8	10	
9	10	
10	10	
11	24	
Total:	169	

1. Remember to answer all questions on your answer sheet, not directly on the test (including a-g below).

   (a) (3 points) Linked lists are always allocated in the:

   (a) **heap**

   (b) (3 points) Linked lists use what type of memory allocation?

   (b) **Dynamic**

   (c) (3 points) Arrays provide us with what type of access?

   (c) **direct or random**

   (d) (3 points) Accessing an item in a linked list costs?

   (d) **O(n)**

   (e) (3 points) Insertion and deletion operations are faster in?

   (e) **linked lists**

   (f) (3 points) You receive values in some random order, but you want to store them in sequential order. Which is better, array or linked list?

   (f) **linked lists**

   (g) (3 points) You have a set of ordered values. Now you want to search for a value in that set. Which is better, array or linked list?

   (g) **array**

2. (20 points) List the complexities from fastest to slowest.

Complexity Choices

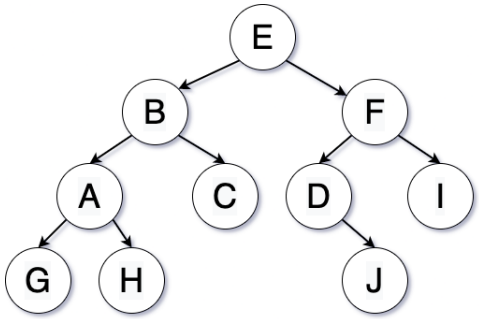
$O(n!)$ $O(2^n)$ $O(1)$ $O(n\ lg\ n)$ $O(n^2)$ $O(n^n)$ $O(n)$ $O(log\ n)$ $O(1000)$

Solution: The spreadsheet below shows the growth of each choice based on the N value in column 1. The columns go from least cost on the left to greatest cost on the right. And even though it doesn't look correct, the O(1000) is a constant, therefore it falls right after O(1). Again, this is why I dislike asymptotic notation for certain aspects of algorithm analysis.

N	O(1)	O(lg N)	O(N)	O(N lg N)	O(N^2)	O(2^N)	O(N!)	O(N^N)
1	1	0	1	0	1	2	1	1
2	1	1	2	2	4	4	2	4
3	1	2	3	5	9	8	6	27
4	1	2	4	8	16	16	24	256
5	1	2	5	12	25	32	120	3125
6	1	3	6	16	36	64	720	46656
7	1	3	7	20	49	128	5040	823543
8	1	3	8	24	64	256	40320	16777216
9	1	3	9	29	81	512	362880	387420489
10	1	3	10	33	100	1024	3628800	10000000000
11	1	3	11	38	121	2048	39916800	285311670611
12	1	4	12	43	144	4096	479001600	891610044825
13	1	4	13	48	169	8192	6227020800	302875106592

O(1000)

3. (15 points) Give a Pre-order / Post-order / In-order traversal of the following tree:

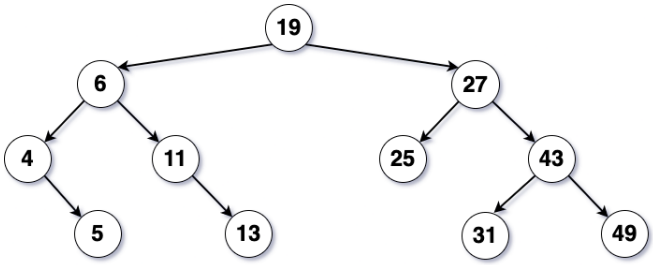


PreOrder	E	B	A	G	H	C	F	D	J	I
InOrder	G	A	H	B	C	E	D	J	F	I
PostOrder	G	H	A	C	B	J	D	I	F	E

4. Given the following list of numbers: **19, 6, 11, 4, 13, 5, 27, 43, 49, 31, 25**

   (a) (10 points) Process them from left to right and draw your resulting Binary Search Tree.

Solution:



   (b) (3 points) Is this tree complete?

Solution: No

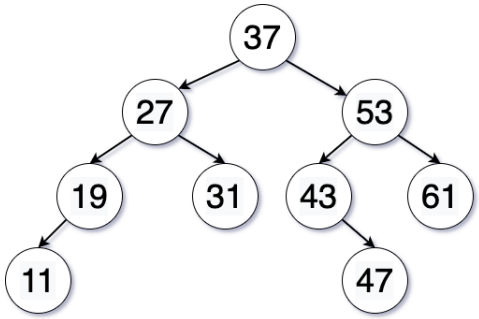
   (c) (3 points) Is this tree full?

Solution: No

   (d) (3 points) Is this tree balanced?

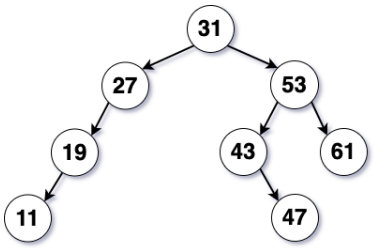
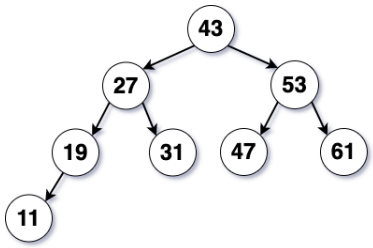
Solution: Yes

Use the following tree for the following questions dealing with node deletion and AVL trees. For each question, start with the original tree.

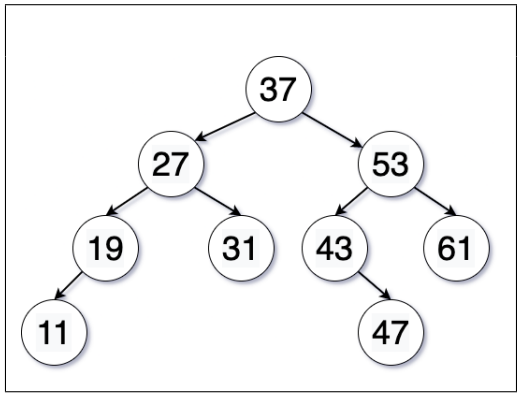


Assuming the tree is a plain Binary Search Tree:

5. (15 points) Delete **37** from the binary search tree above and draw the TWO possible resulting trees.

Predecessor (31)	Successor (43)
	

6. (10 points) Delete **43** from the binary search tree above and draw the resulting tree.



7. (15 points) Write a pseudo code implementation for deleting a node from a binary search tree.

if node is null:
Just return and stop the recursion (looking for value to delete)

if current node is less than the value we're looking for:
recursively call delete with right child

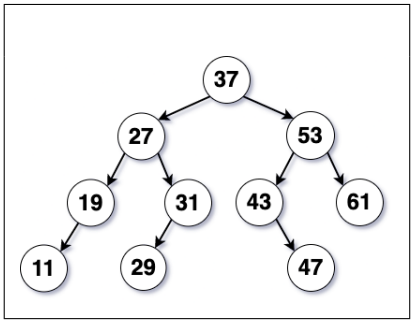
if current node is great than the value:
recursively call delete with left child

otherwise, we've found the node we want to delete:
- if the node has no children, delete node were done
- if the node has one child, replace deleted node with existing child
- node has 2 children:
    - replace deleted node with inorder successor or inorder predecessor

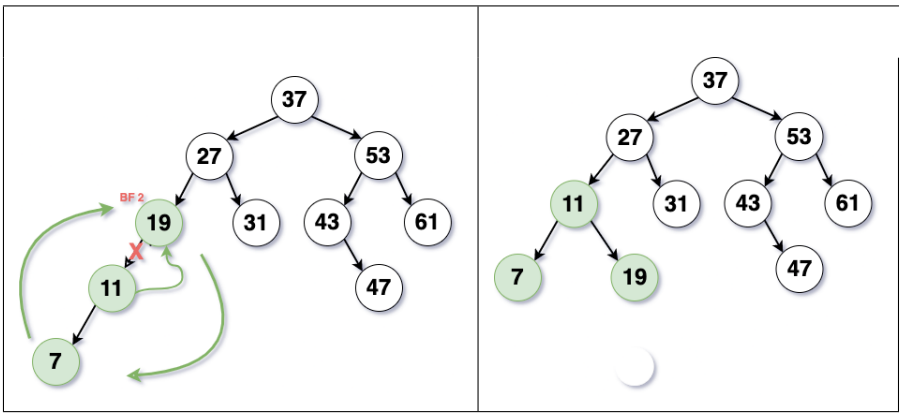
---

Now assume the tree is an AVL tree. Remember for each question use original tree.

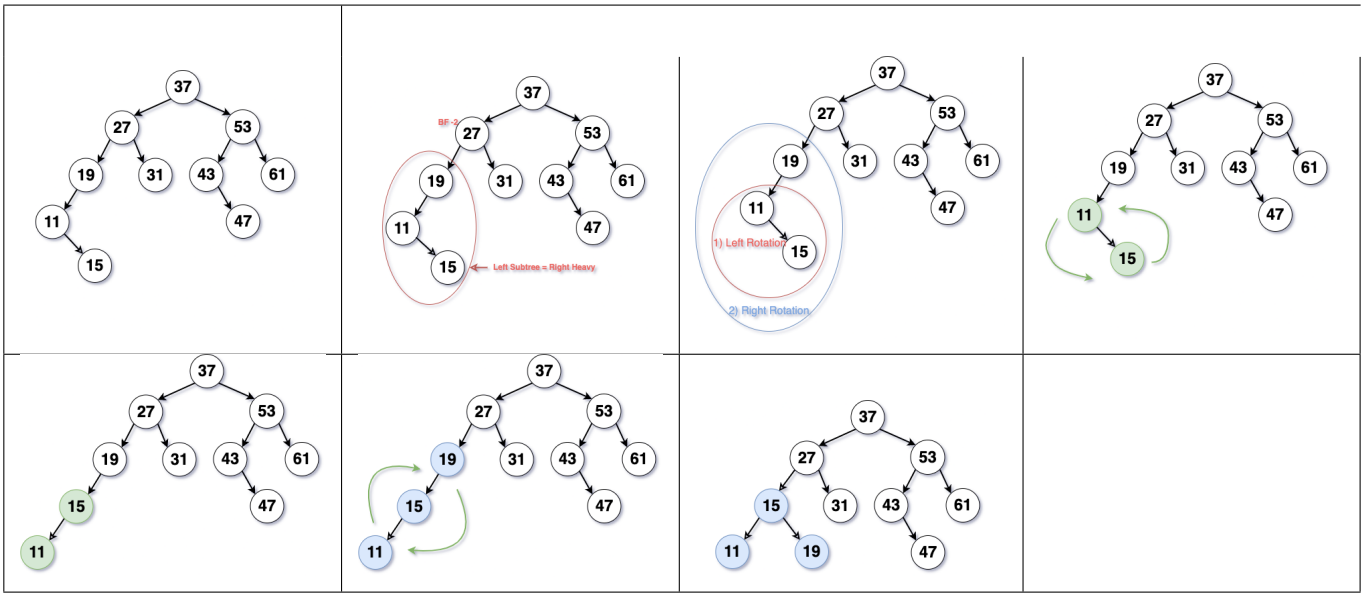
8. (10 points) Insert the value **29**.



9. (10 points) Insert the value **7**.



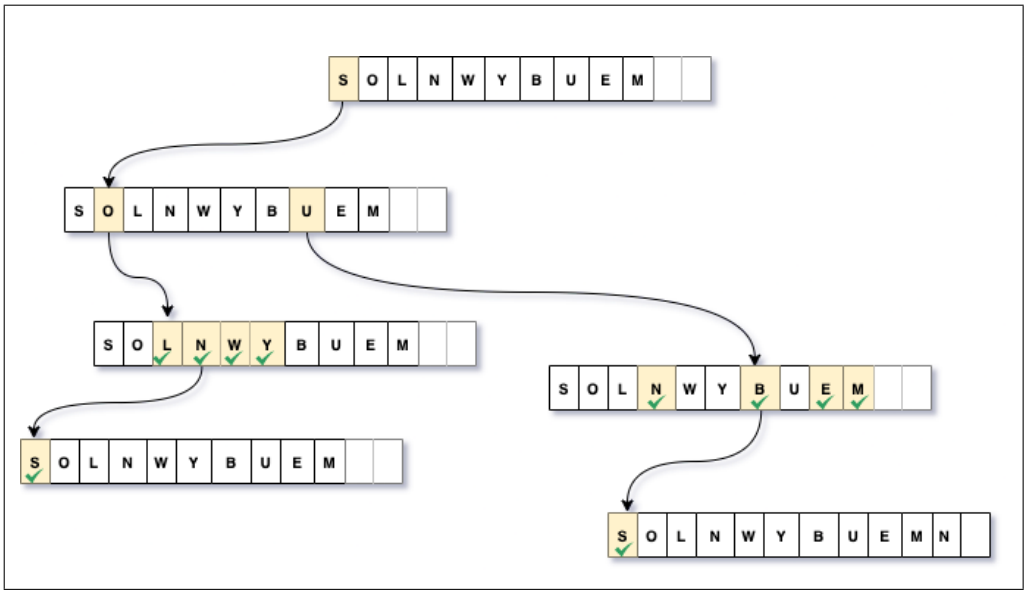
10. (10 points) Insert the value **15**.



---

11. (a) (15 points) Add the following words to a Trie data structure.

sol, sons, sow, soy, subs, sue, sum, sun



(b) (3 points) What is the complexity of creating a Trie?

Solution: O(N)

(c) (3 points) What is the complexity of searching for a specific word in a Trie?

Solution: O(N)

(d) (3 points) What is the complexity of searching for all words with a given prefix in a Trie?

Solution: O(N)