T.C. ISTANBUL AREL UNIVERSITY						
FACULTY OF ENGINEERING						
DEPARTMENT OF COMPUTER ENGINEERING						
Academic	**Exam**			**Course**		
Year	**Type**	**Date**	**Time**	**Code**	**Name**	**Instructor**
Fall 2024	Final	10.01.2025	10:00	CENL201	Data Structures	Tuğberk Kocatekin

INSTRUCTIONS

1. Cell phone use is prohibited.
2. Leaving the classroom within the first 15 minutes of the exam is prohibited.
3. Leaving the classroom during the last 5 minutes of the exam is prohibited.
4. You can enter the exam no later than 15 minutes after it starts.
5. There must be at least two students in the classroom until the end of the exam.
6. Students whose names are not on the exam list cannot take the exam.
7. In certain questions, even if your answer is partially correct, I may choose to give 0 points due to the nature of the question.

Question:	1	2	3	4	5	6	7	8	9	Total
Points:	8	16	10	5	20	10	21	5	5	100
Score:										

answer sheet

Name & Surname: _____ ID: _____

Signature: _____

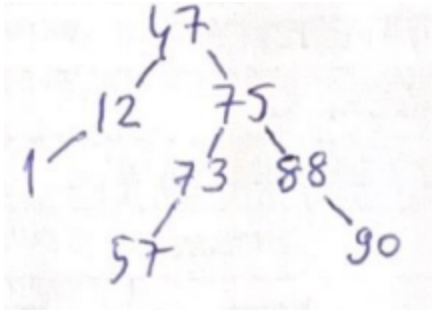
I will probably not be on school premises next week. We have the make-up exam on Friday.
You are responsible from every topic we've covered. Make sure to study Greedy Algorithms.
See you on Friday.

Questions

1. **(Binary Search Trees)**

   (a) (5 points) Draw a Binary Search tree of the given numbers: `{47, 12, 75, 88, 90, 73, 57, 1}`

Remember, we start by adding the first number as root. Later, for each number we start comparing it with the root. So, 47 is root. Next, we have 12. 12 < 45 -> we add it to left. We have 75 next. We compare it with 47 again, it is larger -> Goes to right. This is not a heap, a BST. We don't have to have a complete binary tree.

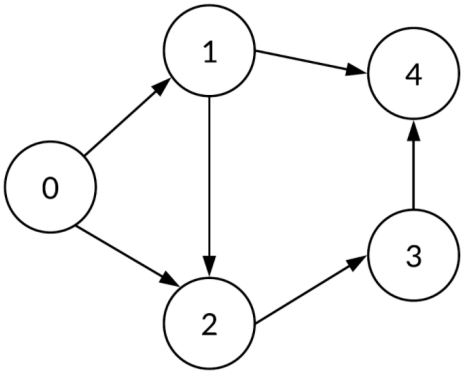


   (b) (3 points) Apply **inorder traversal** to this tree and write down the order.

In BST, inorder is sorted: 1, 12, 47, 57, 73, 75, 88, 90

2. **(Graph)**

   Answer the questions below based on the given graph.



Adjacency Matrix

	0	1	2	3	4
0		1	1		
1			1		1
2				1	
3					1
4					

   (a) (5 points) Fill in the adjacency matrix above.

   (b) (5 points) Create the adjacency list of the given graph.

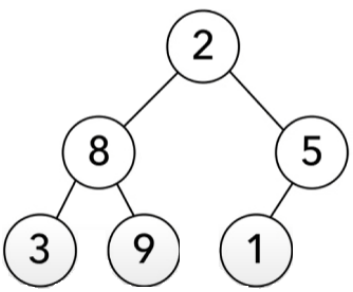
0 -> 1,2
1 -> 2,4
2 -> 3
4 ->

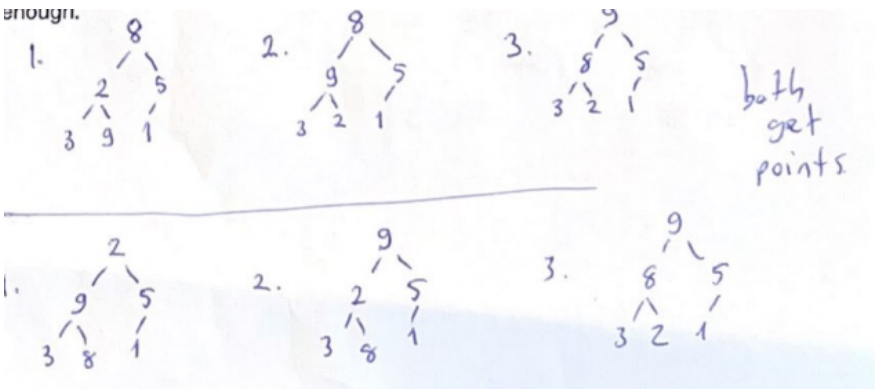
   (c) (3 points) Is the graph weighted or unweighted?

   (d) (3 points) Is the graph directed or undirected?

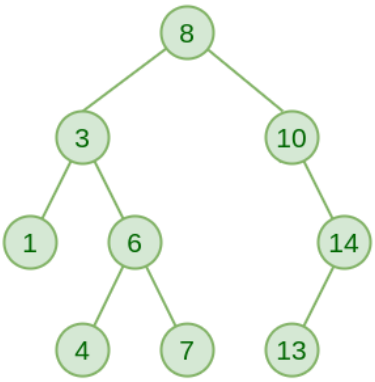
3. (10 points) **(Heap)**

   In *heap sort*, we turn a given tree into a **max-heap**. Later, we delete items one by one. This gives us **in-place** sorting. At the end, we end up with a sorted list without using any additional lists or arrays. Below, there is a tree. **Turn this tree into a max-heap. Please show your steps to get points.** Don't need construct the heap sort. Giving the max-heap is enough.

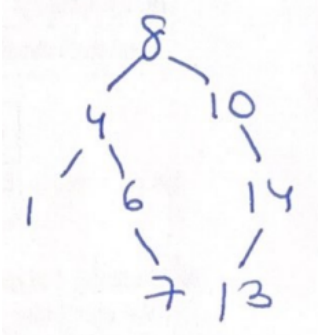




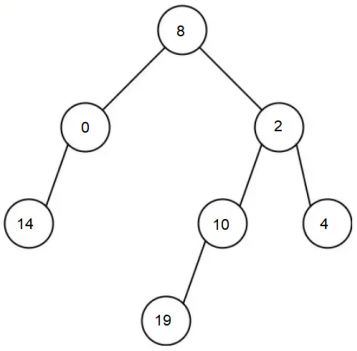
4. (5 points) **(Trees)** The tree below is a binary search tree. Delete the node with value **3**. Draw the remaining tree next to it, show your steps.



Inorder successor of 3 is 4. So, when we delete 3 we need to replace it with 4
Many of you put 6.
Remember, if you are selecting the right subtree, you choose the one on the most left (3)
You can also do it the other way (selecting the left subtree) but only one person did it and I gave pts to that too.



5. **(Tree Traversal)** Answer the questions based on the tree below.



(a) (5 points) What kind of a tree is this? *(Only one right answer)*

A. Perfect Binary Tree
B. Complete Binary Tree
C. Full Binary Tree
D. Binary Search Tree
E. None of the above

(b) (3 points) Which node(s) is (are) the root(s) of this tree?

In a tree there is only one root. 8

(c) (3 points) Which node(s) is (are) the leaves of this tree?

14, 19 and 4

(d) (3 points) Apply **postorder traversal** *(left, right,root)* to this tree and write down the order.

14 - 0 - 19 - 10 - 4 - 2 - 8

(e) (3 points) Apply **preorder traversal** *(root, left, right)* to this tree and write down the order.

8 - 0 - 14 - 2 - 10 - 19 - 4

(f) (3 points) Apply **level-order traversal** (breadth-first) to this tree and write down the order.

Start from the top, right every level from left to right: 8 0 2 14 10 4 19

I also gave points to those who left it empty or wrote -1.

6. (**Sorting**) Answer the questions below. Do not write code, do sorting step by step. Write down each step.

(a) (5 points) Apply bubble sort to this array of numbers: `[1,20,16,10]`

(b) (5 points) Apply selection sort to this array of numbers: `[9,0,16,4]`

bubble
1. 1, 20, 16, 10
1, 16, 20, 10
1, 16, 10, 20
1, 10, 16, 20

s.sort
1. 0, 9, 16, 4
0, 4, 16, 9
0, 4, 9, 16.

7. **(Hash tables)**

(a) (5 points) What is a collision in a hash table implementation? *(There is only one right answer)*

A. Two key-value pairs that have equal keys but different values.

B. Two key-value pairs that have different keys and hash to different indices.

C. Two key-value pairs that have different keys but hash to the same index.

D. Two key-value pairs that have equal keys but hash to different indices.

(b) (8 points) For given (key, value) pairs, fill in the hash table below with the following conditions:

- We use **closed addressing**. *(hint: linked list)*
- Hash function is $H(k) = k \ mod \ 8$
- Key value pairs are:

(8, E) - (48, L) - (90, N) - (18, H) - (91, D) - (74, U) - (44, A) - (61, G)

Many of you confused b) and c)
I deducted some points but not whole.

Apply the hash function to everyone and add it to the slot. If there is collision, put an arrow and add to the same slot.
The hint was useful.

0	1	2	3	4	5	6	7
E -> L		N -> H -> U	D	A	G		

(c) (8 points) For given (key, value) pairs, fill in the hash table below with the following conditions:

- We use **linear probing**.
- Hash function is $H(k) = k \ mod \ 8$
- Key value pairs are:

(8, E) - (48, L) - (90, N) - (18, H) - (91, D) - (74, U) - (44, A) - (61, G)

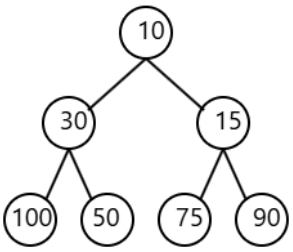
tThe same, get the hash and write to slot.
If slot is full (collision), you select the next empty slot.

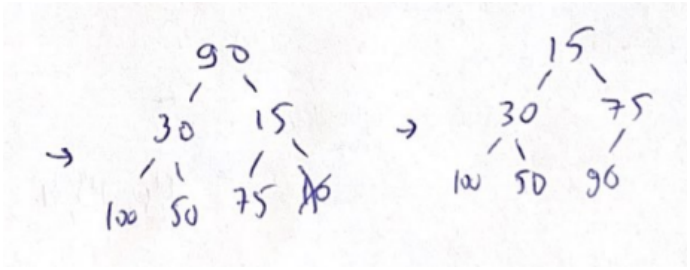
If you wrote keys, letters -> still got point.

8 -> 0
48 -> 0 -> 1
90 -> 2
18 -> 2 -> 3
91 -> 3 -> 4
74 -> 2 -> 3 -> 4 -> 5
44 -> 4 -> 5 -> 6
61 -> 5 -> 6 -> 7

0	1	2	3	4	5	6	7
E	L	N	H	D	U	A	G

8. (5 points) In heaps, when we apply an operation on to it, we need to make sure that it satisfies two conditions: a **complete binary tree**, and a **min(max) heap**. In the min-heap below, delete the root. Show the resulting min heap.





9. (5 points) (**Max Heap**) Insert the following numbers into a max heap. Draw a tree for each heap insertion.
Numbers: `8, 48, 90, 18, 91, 74, 44, 61`

Many of you confused this question, you were lucky it was 5 points.
You need to draw the tree. Previously I gave you are tree and asked it to turn into a heap.
Now, you are going to insert each number and create a heap