CS210 Data Structures
(211) Final Exam

Name:_____ ID_____

Check your section:

☐ Dr. Sawsan Alhalawani ☐ Dr. Najla ☐ Dr. Siwar	☐ Dr. Basit Qureshi ☐ Dr. Muhammad Akour ☐ Dr. Umar Amin ☐ Dr. Sakahr Alkhreyef

Instructions:

- This exam contains four questions with multiple parts, on 6 sheets of papers (2 sided).
- Time allowed: 180 minutes
- Closed Book, Closed Notes.
- Use of Calculators is ALLOWED. Use of other computing devices / smartphones etc is strictly prohibited.
- Answer the problems on the exam sheets only. No additional attachments would be accepted.
- When the "time is over" is called, it is the students' responsibility to submit his exam to the invigilator. Submitting completed exam 3 minutes after the "time is over" will incur a penalty of **5 points**.

Few gentle reminders:

- If you get stuck on some problem for a long time, move on to the next one.
- You should be better off by first reading all questions and answering them in the order of what you think is the easiest to the hardest problem.
- Keep the points distribution in mind when deciding how much time to spend on each problem.

Question No.	Part a	Part b	Part c	Part d	Student's Score
Question 1 (CLO 1)	/3	/2	/2	/3	/10
Question 2 (CLO 2)	/4	/3	/3		/10
Question 3 (CLO 3)	/3	/3	/3	/3	/12
Question 4 (CLO 4)	/6	/2			/8
Total					/40

Question 1. [3 + 2 + 2 + 3 = 10 points]

Part a. [3 points] Draw a directed weighted graph for the following edge-list.

Edge	Weight
0-1	2
0-2	5
1-3	2
1-7	8
2-1	1
2-4	3
3-5	7
3-6	2
4-2	3
4-5	2
5-3	2
5-7	3
6-5	3
7-8	1

Do a BFS and DFS run starting at 0, considering the lowest weighted edge for choosing the next vertex.

DFS at 0	
BFS at 0	

Part b. [2 points] Given the Alg2 method, show the recursion trace for Alg2(0, 32) call. Give the estimated run-time as a function of time T(n) and Big-Oh.

```
public static int Alg2(int m, int n)
{
   if ( n == 1 )
       return m;
   else
       return m + Alg2 (m, n/2);
}
```

Part c. [2 points] Give the best Big-O characterization for each of the following running time estimates (where n is the size of the input problem).

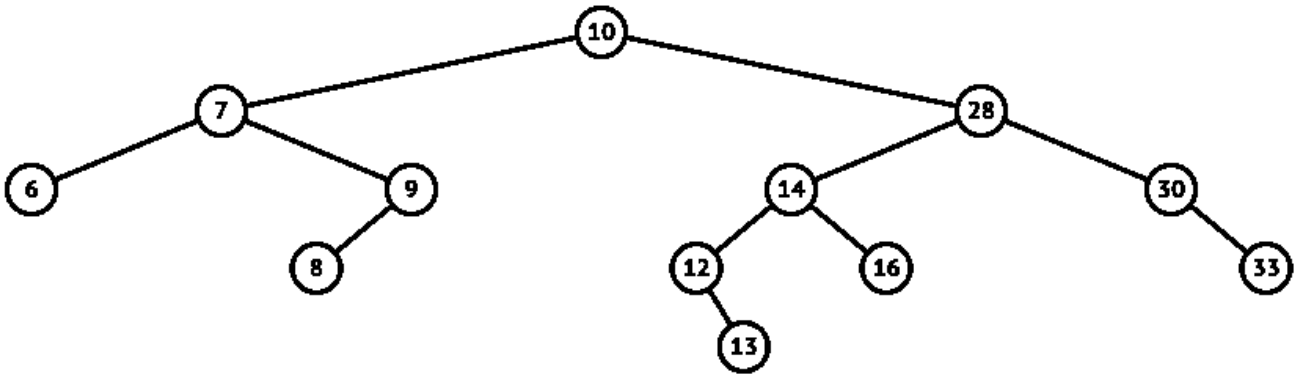
	Running time estimate	Big-O characterization
(a)	$\log n + 10000000$	
(b)	$2^{10} + 100^{50}$	
(c)	$1+2+ \ldots +(n-2)+(n-1)+n$	
(d)	$2^{\log n} + \log n$	

Part d. [3 points] The following AVL tree of integer keys is given. Execute the following insertion and deletion calls on this tree. <u>Show all rotations that apply</u>.

(i) Apply delete(28) on the given tree

(ii) Apply delete(6) on the resulting tree from (i)

(iii) Apply insert(17) on the resulting tree from (ii)



Question 2. [4 + 3 + 3 = 10 points]

Part a. [4 points] Sort the elements of the following array using quick-sort approach. Show all operations (pivot positions etc.). Assume the array is already shuffled. Always choose the first element as pivot. Show all operations.

Cat	Bit	Hat	Mat	Rat	Bat	Hot	Bot	Sit

Part b. [3 points] Circle the correct answer

Least number of comparisons required to sort {C, C, C, C, C, C, C, C}.	Selection-sort or Insertion-sort
Least number of shifts (data moves) required to sort {C, C, C, C, C, C, C, C}.	Selection-sort or Insertion-sort
Faster run-time to sort {A, B, C, D, E, F, G, H}.	Quick-sort or Insertion-sort
Faster run-time searching for value X using Binary-Search algorithm.	Singly-Linked-List or a 1D-Array
Takes less space to sort {A, B, A, B, A, B, A, B}	Merge-sort or Selection-sort
Faster searching for an element in a tree that is not available.	AVL-search or BST-search

Part c. [3 points] Heaps are an important data structure and have many applications. Assume that an array of integers is given; insert the following data in a MAX-heap (read left to right). Show all sink (heap-down) and swim (heap-up) operations for each insertion.

18	12	6	15	9	3	27	21	30

Question 3. [3 + 3 + 3 + 3= 12 points]

Part a. [3 points] A web-browser such as Google-Chrome, opens web-pages as you click on different links on a web-site. A stack is maintained in the browser to store the URL for all pages visited. The back-button, if pressed, would take the user to the most recent previously visited web-page. Write a method in java that returns a String array of URL links for all the pages visited by the user. The method receives a Stack of type String where each String is a URL link. The first element in the returned array should contain the most recently visited URL.

NOTE: You don't want to destroy the Chrome Stack!

```
public String [] URL (Stack ChromeStack){
```

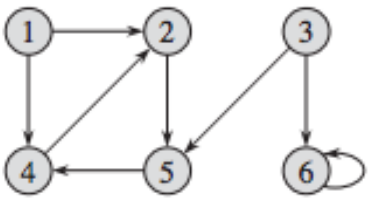
Part b. [3 points] A Binary Search Tree (BST) stores keys/values following the Symmetric-order, i.e., all values smaller than a key go left, all values larger than a key go right. Write the insert method for a BST that stores integers as keys. Assume duplicates (equal keys) are not allowed.

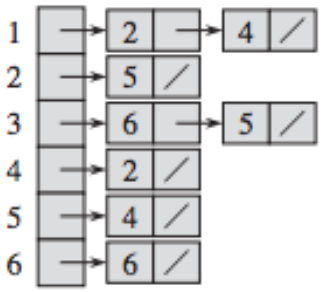
```
public void insert (Node N){
```

Part c. [3 points] In an AVL Tree, if a node to-be-deleted has two children, a candidate node needs to be found in the sub-tree. Assume an AVL Tree T that stores integers as keys, is given. Write a method that returns the candidate node in the sub-tree T.

```
public Node returnCandidate(AVL T){
```

Part d. [3 points] Given an adjacency-list A for a directed Graph G, write a method remove that takes an edge (v, w) as a parameter and removes this edge from the adjacency-list. It returns true if successful, false otherwise. The following is an adjacency-list for the given directed graph.





Hint: The adjacency-list is an array SLL of singly-linked-lists. The array SLL position [0] is left blank intentionally; the first vertex 1 is stored at SLL[1], the next is SLL[2] and so on.

```
public boolean remove (int v, int w){
```

Question 4. [6 + 2= 8 points]

Part a. [6 points] Ahmed wants to store some integers in a hash-table T1 of <u>size 13</u>. He wants to try out some hash functions to see if they would ensure uniform distribution of the data. The following functions are available f(x), g(x) and h(x).

$\boldsymbol{f(x) = x\%5,} \qquad \boldsymbol{g(x) = (x * 2)\%11,} \qquad \boldsymbol{h(x) = (x\%6) * 2.}$

Insert the following data in the hash table: 3, 15, 17, 22, 19, 16, 27, 35, 46, 8.

Assume that collisions are resolved using linear-probing method; compute the number of collisions and probes/displacement for each of the hash functions.

0		0		0	
1		1		1	
2		2		2	
3		3		3	
4		4		4	
5		5		5	
6		6		6	
7		7		7	
8		8		8	
9		9		9	
10		10		10	
11		11		11	
12		12		12	

Number of collisions with f(x) ____________________

Number of collisions with g(x) ____________________

Number of collisions with h(x) ____________________

Number of probes/displacements with f(x) ____________________

Number of probes/displacements with g(x) ____________________

Number of probes/displacements with h(x) ____________________

Part b. [2 points] For the information provided in part(a); select the best hash function. Using this hash function, insert the same data (as in part a) into the hash tables T2 of size 17.

0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	

Number of collisions in T2

_____

Number of probes/displacements in T2

_____

Did the number of collisions and displacements decrease as the size of the table increase? Explain

Note: In case you forgot your ABCs! use this to help you determine the order

a b c d e f g h i j k l m n o p q r s t u v w x y z

<Scratch sheet. DO NOT detach>