	Course Name:	Data Structures and Algorithms	Course Code:	CS2002
	Program:	BS Electrical Engineering + BS Robotics	Semester:	Fall 2022
	Duration:	3 Hours	Total Marks:	75
	Exam Date:	29-Dec- 2022	Weight:	50%
	Section:	BEE-3A, BSR-3A	Page(s):	10
	Exam Type:	Final		

Student Name: _____ Roll No. _____ Section: _____

Instruction/Notes:	<ul style="list-style-type: none"> The exam is closed book and notes. Please answer all questions within the spaces provided. Use only black or blue pen to solve the paper. You may ask for extra sheets but they would not be collected or marked.
--------------------	--

Question # 1

[CLO 1, Marks: 15]

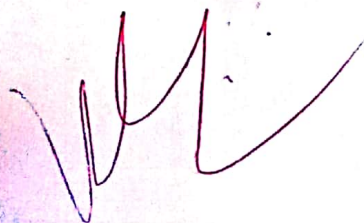
- i. Calculate the time complexity expression for the following code snippet with respect to n and express it in terms of Big-O notation. Show all working for full credit.

```

for (int i=1; i<n; i=i*2)
{
    for (j=0; j<i; ++j)
    {
        sum++;
    }
}

```

- ii. While doing worst case analysis of an algorithm, I found it takes 2^{2+n} steps to complete the job, where n is the input size. Can I say it is $O(2^n)$? Why or why not?



- iii. Calculate the appropriate time and space complexity expression of the following function with respect to N and also express both in terms of Big-O. Show all working for full credit.

```
int Enigma(int N)
{
    if(N<=1)
        return 1;
    else
    {
        Enigma(N-N/2);
        Enigma(N-N/2);
    }
}
```

Question # 2

[CLO 2, Marks: 20]

- i. An unordered singly circular linked list contains integer keys and has no sentinel nodes. The int of the node and SCLinkedList class is given below:

<pre>struct Node { int key; Node *next; }</pre>	<pre>class SCLinkedList { //points to last node of the list Node *last; public: // to be implemented bool deleteAfterKey(int m); }</pre>
---	---

You have to implement in C++ the iterative function deleteAfterKey which finds the node containing 1 equal to m (passed as parameter), and deletes this node as well as the next $m-1$ nodes from the list (we

consider the list to end at the last node). So if $m=5$, it will find the node containing 5, then delete this node as well as the 4 nodes that come after it. If there are less than $m-1$ nodes after the node with $key=m$, all these would be deleted.

For example if the list is $2 \rightarrow 9 \rightarrow 4 \rightarrow 3 \rightarrow 1 \rightarrow 2 \rightarrow 5$ then `deleteAfterKey(3)` would result in $2 \rightarrow 9 \rightarrow 4 \rightarrow 5$

And if the list is: $2 \rightarrow 9 \rightarrow 4 \rightarrow 6 \rightarrow 1 \rightarrow 2 \rightarrow 5$ then `deleteAfterKey(6)` will result in $2 \rightarrow 9 \rightarrow 4$

If there is no node in the list with $key=m$, the function simply returns false. If you use any helper function then implement it as well. Make sure your code runs in linear time and creates no memory leaks. It should also handle all special cases.

ii. Convert the following postfix expression to prefix form: $X\ Y\ Z\ /\ +\ E\ J\ \% -\ K\ F\ * +$

iii. Write a function called `copyStack` to copy the elements of an integer type stack, `s1`, into another stack, `s2`, where both `s1` and `s2` are reference parameters, and `s2` is initially empty. `copyStack` is a global function which is only allowed to use the standard stack ADT functions: `push`, `pop`, `top` (or `peek`), and `empty`. Furthermore, `copyStack` *cannot* create an extra array or any other data structure, but it can use one or two extra variables. When the function ends, `s2` contains exactly the same elements as `s1`, and `s1` is exactly as it was initially. Following picture shows an example, before and after `copyStack`:

4	
2	
13	
11	
5	

s1

s2

Before copyStack

4	4
2	2
13	13
11	11
5	5

s1

s2

After copyStack

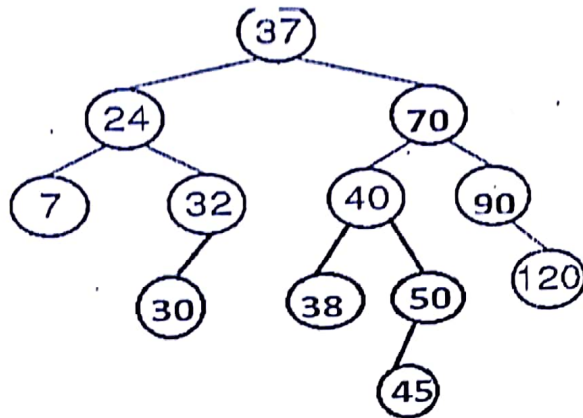
Question # 3

[CLO 4, Marks: 20]

- i. Add the numbers 120, 55, 22, 13, 1, 5, 25, 15 to a Hash table with 7 buckets using the simple modulo function as the hash function. Show the table if **Chaining** is chosen for the hash table for overflow handling.

- ii. Illustrate the execution of the first pass of the basic Quicksort algorithm on the array $A = 43, 13, 89, 34, 21, 44, 99, 56, 9$

- iii. Delete 7 from the AVL Tree shown below. Please mention imbalance type and rotation needed alongwith all steps for full credit.



iv. Write an algorithm (in terms of pseudocode) **SumOfLargestKvalues** that takes:

- An array containing n distinct natural numbers
- Size of array (n)
- A number $k \leq n$

and calculates the sum of the k largest numbers in the array.

For example, if the array is $\{3, 7, 5, 12, 6\}$ and $k = 3$, then the function should return 25 ($12+7+6$) i.e. the sum of 3 largest values of array.

You may freely use standard data structures and algorithms from the course in your solution, without explaining how they are implemented. **Your algorithm should not take more than $O(n \log n)$ time.**

- i. WhatsApp is a mobile app that allow its users to send and receive text, audio and video messages. The WhatsApp team wish to find whether a messages reach back to its sender or not? For example, a person P sends a message to his friends and they forward the message to their friends and so on. Given the users of WhatsApp and the information of all the messages sent by its users to other users, we need to determine that a message is sent back to its user or not for a particular user.
- a) What data structure(s) and algorithm(s) is most suitable to represent this data and why?
- b) How would you solve this problem with your suggested data structure? Briefly explain your idea in the form of pseudocode.

ii. You are required to develop a cab hailing app. Once a client requests a cab via your app, it should select the cabs which are free in the vicinity of the client (the one who is closest to the client should be allocated the request first, if the cab driver declines the request then it should be forwarded to the second nearest and so on) till a cab driver accepts the ride request. Once the cab has reached the client, your app should calculate the shortest distance to the destination and guide the driver accordingly.

a) Which data structures and algorithms would you like to use in your app so that it functions efficiently?

b) Give an algorithm (in terms of pseudo-code) that should run on your data structures specified above and selects cabs located in the vicinity of the client (the driver present nearest should be selected first, then the 2nd nearest and so on).