Name ..... *Roll#* .....

Q1	Find Time Complexity.	15
1	**void recursiveFunction(int n) {** **if (n <= 1)** **return;** **for (int i = 0; i < n; i++) {** **cout << "Processing: " << i << endl;** **}** **recursiveFunction(n / 2);** **recursiveFunction(n / 2);** **}**	

2

```
for (int i = 1; i <= n; i++) {
    for (int j = 1; j < i; j *= 3) {
        cout << "i: " << i << ", j: " << j << endl;
    }
}
```

$n/2$. — n

— $\log_3 n$.

$n-1$

$n \log n$.

int i = 1	1	$(1 + n + n-1)(1 + (n-1) + ($
$i \le n$	n	Second for loop.
i++	$n-1$	3/
int j = 1	1	
$j < i$	$n-1$	
$j *= 3$	$\log_3 n$	$O(n \log n)$

first for loop

Second for loop.

INFORMATION

Name **.......................................** ***Roll#*** **...................................**

3

```
int recursiveFunction(int n) {
   if (n <= 1)
      return 1;
   return 2 * recursiveFunction(n / 2) + n;
}
```

3

4

```
for (int i = 1; i <= n; i++) {
    for (int j = 1; j < i; j *= 2) {
        for (int k = 1; k <= j * j; k++) {
            // Some constant time operation
        }
    }
}
```

3

Name .... Roll# ....

5	(code below)	3

```
for (int i = 1; i < n; i++) {          —— n
    for (int j = 1; j < i * i; j++) {      —— n
        for (int k = 0; k < j; k++) {      —— n.
            // Constant time operation
        }
    }
}
```

n × n × n.

= n³.

	1st loop		2nd loop	
	j=1	1	j=1	1
	i<n	n	j<i	(n−1)2.
	j++	n−1	j++	(n−1)²−1.
		2n		n².
		n		

n²)

1.5

$O((n)(n^2)(n^2))$

$O(n^3)$. $= O(n^4)$.

Q2 **Sorting Floating Point Numbers using Bucket Sort** — 15 (14)

You are given an array of floating-point numbers with up to two decimal places. These numbers need to be sorted in ascending order. However, you are required to use Bucket Sort, which typically operates on integer values.

To avoid losing precision, follow these steps:

1. Multiply each floating-point number by 100 to convert it into an integer.
2. Ensure the array contains only unique numbers (i.e., no repeated values).
3. Sort the array using the Bucket Sort algorithm on these integer values.
4. After sorting, divide the entire array by 100 to convert the numbers back to their original floating-point form.

Name ..... *Roll#* .....

Q3	**Priority-Based Task Management System with Completed Tasks** You need to build a task management system where users can add, complete, and remove tasks based on priority. Each task has a title, description, and priority level (1 for High, 2 for Medium, 3 for Low). Tasks should be managed as follows: **Features:** • **Add Task:** Insert new tasks in the active list after the last task of the same or higher priority. • **Complete Task:** Mark a task as complete, remove it from the active list, and add it to a separate "completed tasks" list in the order completed. • **Remove Task:** Remove a task by title, regardless of its priority. **Questions:**	30
Part a)	**Q1:** Which data structure (linked list, stack, array, etc.) would you choose to implement this system? Explain your choice, focusing on task insertion, removal, and completed task management.	15
Part b)	**Q2:** Implement a method to insert tasks into the active task list based on priority.	5
	Q3: Implement a method to mark a task as complete, removing it from the active list and adding it to the completed tasks list.	5
	Q4: Implement a method to remove a task from the active list by its title.	5

10.5