United International **University**
Department of **C**omputer **S**cience and **E**ngineering
CSE 2216, Data Structure & Algorithm Lab

1. For this problem you can use the singly linked list code implemented in lab class. Take an integer N (0<N<1000) as input from user. Generate N random values (range: 1-100) and store them sequentially in the list. Now implement the following function:

    - node* customOrder(node *head)

    This function will reorder the values within the list in such a way that numbers appeared most will place first. Ascending or descending order is not required for the numbers appeared the same time.

Sample Input	Sample Output
N = 5 List: 14 9 11 3 14	14 14 9 11 3
N =7 List: 14 9 11 3 14 3 20	14 14 3 3 9 11 20

2. Write a program which will take an integer N (0<N<1000) from the user. Generate N random numbers (range: 1-100) and store them into an array. Now write a function which will print the value that occurred most of the time. If multiple values occurred maximum time, then print all the values with number of occurrence.

Sample Input	Sample Output
N = 5 Random values: 12 8 25 9 25	25 occurred 2 times
N = 7 Random values: 25 8 25 9 25 9 9	9, 25 occurred 3 times

3. For this problem you must use the singly linked list code implemented in lab class. Take an integer N (0<N<1000) as input from user. Generate N random values (range: 1-100) and store them sequentially. Now implement the following function:

    - node* seperateOddEven(node *head, int direction)

This function will separate the odd and even values stored in the linked list. If direction=1 then odd numbers will place first and even numbers after that. If direction=2 then odd numbers will come after even numbers.

Sample Input	Sample Output
N = 5, direction=2 List: 12 8 25 9 14	14 12 8 25 9
N = 7, direction=1 Random values: 25 8 25 9 15 9 90	25 25 9 15 9 90 8

Order of numbers is not important. Your only focus will be separate the even values from odd values.

4. For this problem you can use the singly linked list code implemented in lab class. Take an integer N (0<N<1000) as input from user. Generate N random values (range: 1-100) and store them sequentially in the list. Now implement the following function:

    - node* customSort(node *head)

    This function will sort the values within the list in such a way that even numbers appeared first in ascending order and then the odd numbers in descending order.

Sample Input	Sample Output
N = 5 List: 14 9 11 3 14	14 14 11 9 3
N =7 List: 14 9 11 3 14 3 20	14 14 20 11 9 3 3

5. Write a program which will take an integer N (0<N<1000) from the user and print the cubic root of N. You cannot use any built in or library function to calculate the cubic root. Consider 0.005 as threshold value.

Sample Input	Sample Output
N = 15	2.466
N =27	3.000

6. For this problem you can use the singly linked list code implemented in lab class. Take an integer N (0<N<1000) as input from user. Generate N random values (range: 1-100) and store/delete them sequentially by implementing the following function:

    - node* notMoreThanTwo(node *head, int value)

This function will add the number in head if this is first appearance, insert in tail if it is a duplicate value, and delete the first value if it is in list twice already. For each insertion print **%d inserted** and for deletion print **%d deleted,** where %d is the randomly generated value.