Department of Data Science

Name				**Id**	
Course	Data Structures and algorithms				
Date	8 January 2024				

Question I:

Write a Java method called **areElementsEven** which takes a Node parameter and check if the elements of a linked list are even.
To check if the elements of a linked list are even
For example, if the list was: List1 head→12 →4 →20 →22 →null
The method returns true

if the list was: List1 head→12 →13 →21 →20 →null
The method returns false

The header of the method is as follows:

public static boolean areElementsEven(Node head)

Assuming that the size of the first list is n, and without showing the calculations details, what do you think is the time complexity of your algorithm using the big Oh notation. Explain your answer briefly.

Question II:

Write a void method named splitStack() that receives 3 parameters: a stack , a list, and a queue. The method copies all positives values in the linkedlist to a stack , and all negatives values to a queue. The header of the method is:

public static void splitStack (Stack s, Linkedlist ll, Queue q)

For example :
If L will contain

13
-2
7
-9
6
4
-5
8

S will contain: head→**8**→ **4** →6→ **7** → **13** → **null**

Q will contain :

-2	-9	-5

Question III:

Write the **extractPartFromStack()** method which takes the original Stack, a start index, and an end index as input and returns a Stack which contain all values from the original stack between start index and end index.

For example :

If the original stack was

12
4
7
-1
6
4
-5
8

And start index = 1 and end index =4

The resulting Stack will contain

4
7
-1
6

public static Stack extractPartFromStack(Stack originalStack, int startIndex, int endIndex)