BRAC UNIVERSITY
Department of Computer Science and Engineering

Examination: Midterm
Duration: 70 Minutes

Semester :Summer 2022
Full Marks: 30

CSE 220: Data Structures

Answer the following questions.
Figures in the right margin indicate marks.

Name: KASHFIA SAFIR SHABAB ID: 20241037 Section: 14

1. CO1, CO5 Implement the following **addArrays** function which takes as input two linear arrays of the same length/capacity and **constructs** a new linear array of the same length/capacity as output. The first element of this new array should be the sum of the first element of the first array and the last element of the second array; the second element should be the sum of the second element of the first array and the second last element of the second array and so on. If there is any "None" value in any of the arrays, you can assume that it is a zero. 10

You are not allowed to use any built-in python functions except len() and range(). Using negative indices in python (such as ar[-1]) is not allowed.

Python:

```
def addArrays(arr1, arr2):
        # Write your code here
```

Java:

```
public int[] addArrays(int[] arr1, int[] arr2) {
        // Write your code here
}
```

Sample Input	Sample Output	Explanation
ar1 = [1, 4, 5, 8, None, None, None] ar2 = [7, 4, 8, 9, 5, None, None] ar3 = addArrays(ar1, ar2) print(ar3)	[1, 4, 10, 17, 8, 4, 7]	1+None = 1 4+None = 4 5+5 = 10 8+9 = 17 And so on.

2. CO3 Complete the following **hasCycle** function that takes the head of a singly linear linked list as an input and as output to **determine** if a particular node can be reached again and again. We denote it as a cyclic linked list. If the linked list has a cycle, return True, otherwise, return False. Importantly, no nodes should be constructed in the implementation of the method. **Hint: You can take the help of multiple data structures to solve this problem and assume the list will have a maximum of 1000 nodes.** 10

Python:

```
def hasCycle(head):
        # Write your code here
```

Java:

```
public boolean hasCycle(Node head) {
        // Write your code here
}
```

Sample Input	Sample Output	Explanation
head || v 15 --> 41 --> 33 --> -43 -- ^ | |_____|	True	There is a cycle in the linked list where the node at index 1 can be reached again and again.
head || v 15 --> 41 --> 33 --> -43 --> None	False	There is no cycle in the linked list where a particular node can be reached again and again.

3. CO1, CO5

a. **Convert** the following infix notation to postfix using stack following the given precedence of the operators. You must show the workings. You do not need to write code. 5

p - q * [o - r % s == t / (u || v <= w) != y + (z && a)]

Precedence (decreases down the order)

1. * , /
2. %
3. + , -
4. ==, !=, <=, >=
5. &&
6. ||

b. **Evaluate** the following postfix expression using stack. You must show the workings. You do not need to write code. 5

5 6 7 1 9 * 3 / 7 + 5 - 2 * 5 % - 5 + 8 - * +