BRAC UNIVERSITY

Department of Computer Science and Engineering

Examination: Midterm
Duration: 80 Minutes
No. of Questions: 3

CSE 220: Data Structures

Semester: Spring 2023
Full Marks: 30
No. of Pages: 2

Name: (Please write in CAPITAL LETTERS)	ID:	Section:

A

✔ Use the back part of the answer script for rough work. No washroom breaks.
✔ At the end of the exam, put the question paper inside the answer script and return both.

Question 1: CO1, CO5 [2 + 8 Points]

I. Can we change the size of an array after initialization? If yes **explain** how.

II. **Complete** the function **multiple_repeat** that takes an array as a parameter and counts the repetition of each element. If an element has appeared in the array more than once, then its 'repetition' is its number of occurrences. The function returns True if there are at least two elements with the same number of 'repetition'. Otherwise, return False. **You cannot use any built-in function except len() and range().**

Python Notation	**Java Notation**

def multiple_repeat (arr):
    # To Do
	public boolean multiple_repeat (int[] arr){
	    // To Do
	}

Sample Input array	**Sample Output**	**Explanation**
[4,5,6,6,4,3,6,4]	True	Two numbers repeat in this array: 4 and 6. 4 has a repetition of 3, 6 has a repetition of 3. Since two numbers have the same repetition output is True.
[3,4,6,3,4,7,4,6,8,6,6]	False	Three numbers repeat in this array: 3, 4 and 6. 3 has a repetition of 2, 4 has a repetition of 3, 6 has a repetition of 4. Since no two numbers have the same repetition output is False.

Question 2: CO3 [2 + 8 Points]

I. **Write** down two advantages of Linked List over arrays.

II. **Complete** the function **remove_from_last** which will take a head of Singly Linear Linked List and a number N. The function should remove Nth node from the last and then return the head of the linked list. If N is not valid then return the head of the unchanged linked list. **You need to modify the given linked list.**

Sample Input	**Resulting List**	**Explanation**
10 → 20 → 5 → 15 → 25 N = 2	10 → 20 → 5 → 25	The 2nd node from last is 15 and thus 15 is removed
10 → 20 → 5 → 15 → 25 N = 8	10 → 20 → 5 → 15 → 25	There are total 5 nodes. So no such 8th node from end exists and thus the linked list is unchanged

Python Notation	**Java Notation**

def remove_from_last(head, n):
    # To Do
	public Node remove_from_last(Node head, int n) {
	    // To Do
	}

Question 3: CO1, CO5 [4 + 6 Points]

I. **Evaluate** the postfix expression using stack. You must show the workings. You do not need to write code. Values and operators are separated by space.

2 7 * 5 13 + 6 / 2 8 + - -

II. Faisal is working in a diamond mine, trying to extract the highest number of diamonds "<>". A "<" followed by ">" forms one new diamond. He must exclude all the sand particles found (denoted by ".") in this process to extract new diamonds. For instance, if he has an input ".<...<<..>>....>....>>>." three diamonds are formed. The first is taken from <..> resulting ".<...<>....>....>>>." The second diamond is then removed, leaving ".<.......>....>>>." The third diamond is then removed, leaving at the end ".....>>>." without the possibility of extracting new diamonds. Hence, 3 diamonds have been extracted.

You need to solve the above problem using Stack. Consider that a **MidStack** class has been created containing the push(element), pop(), and peek() functions. **No need to implement MidStack class.**

Complete the function **count_diamond** which will take an object of MidStack and a string and then return the number of diamond can be extracted using the mentioned process.

Note: The MidStack class implements a singly linked list-based Stack hence overflow is not possible. The pop() and peek() functions return None in case of the underflow. The node class has a elem and a next variable.

Python Notation	Java Notation

def count_diamond(stack, st):
# To Do 	
	public int count_diamond(MidStack stack, String st) {
	// To Do
	}

Sample Input String	Sample Output
<..><.<..>>	3
<<<..<......<<<<....>	1

Sample Driver Code. (You do not need to write this on your script).

stack = MidStack() # and empty stack

st = "<..><.<..>>" # input string will always contains "<", ">" and "."

print(count_diamond(stack, st)) # This should print 3