Question - 1

Arrays

SCORE: **5 points**

Select the option(s) which best describes array data structure :

○ is of variable size and is accessed sequentially from the head

● is of fixed size and can be accessed randomly (by index)

○ is grow-able and is accessed by key

○ is of fixed size and can be accessed only by iterating from start index

Question - 2

Stack

SCORE: **5 points**

Which Data-Structure is more apt for implementing Stack?

○ Array

● Linked List

○ Vector

○ Hash Map

Question - 3

Time Complexity

SCORE: **5 points**

What is the time complexity to count the number of elements in the linked list?

○ O(1)

● O(N)

○ O(logN)

○ None of the above

Question - 4

BigO

SCORE: **5 points**

Help

The Big O notation is mostly concerned with:

○ describing the complexity of for the average case

○ describing the lower bound on complexity

● describing the upper bound on complexity

○ describing both the lower bound and upper bound on complexity

Question - 5

SCORE: **50 points**

Delete Nodes Greater Than X

Easy | Linked Lists | Data Structures | Algorithms | Problem Solving | Core Skills

Given a singly linked list, remove nodes greater than X.
Example:

```
List = 100 → 105 → 50
X = 100
List becomes 100 → 50
```

Return a reference to the root node of the list after removing 105.

Function Description
Complete the function *removeNodes* in the editor below. The function must return a reference to the root node of the final list.

removeNodes has the following parameter(s):
listHead: a reference to the root node of the singly-linked list
x: integer, the maximum value to be included in the returned singly-linked list

Constraints

- $1 \le n, x \le 10^5$
- $1 \le SinglyLinkedListNode\ values \le 10^5$

▼ Input Format for Custom Testing

Input from stdin will be processed as follows and passed to the function.

The first line contains an integer *n*, the number of nodes in the linked list.
The next *n* lines each contain an element to insert into the linked list.
The last line contains *x*, the maximum value allowable in the linked list.

▼ Sample Case 0

Sample Input 0

```
5
1
2
```

```
3
4
5
3
```

Sample Output 0

```
1
2
3
```

Explanation 0

n = 5, x = 3

list = 1 → 2 → 3 → 4 → 5

After removing the nodes having *value > 3, list = 1 → 2 → 3.*

▼ Sample Case 1

Sample Input 1

```
5
5
2
1
6
7
5
```

Sample Output 1

```
5
2
1
```

Explanation 1

n = 5, x = 5

list = 5 → 2 → 1 → 6 → 7.

After removing the nodes having *value > 5, list = 5 → 2 → 1.*