upGrad

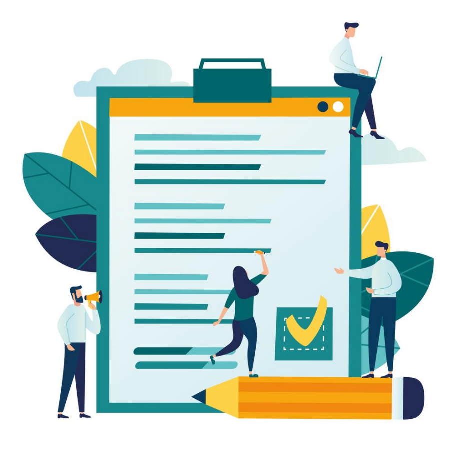
Full-Stack Software Development

Lecture On: Doubly & Circular Linked List

Instructor: Arkoprovo Dey

In Last Class, we covered….

- Introduction to Linked List
- Problems on singly linked list



Homework Discussion

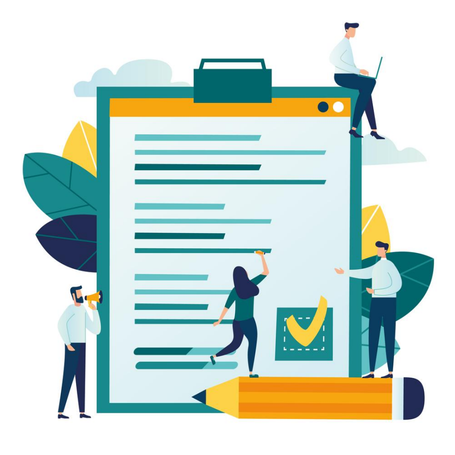
1. Remove loop in a LL.

2. Check for duplicate nodes in LL and delete if any.

Today’s Agenda

1 Circular Linked List

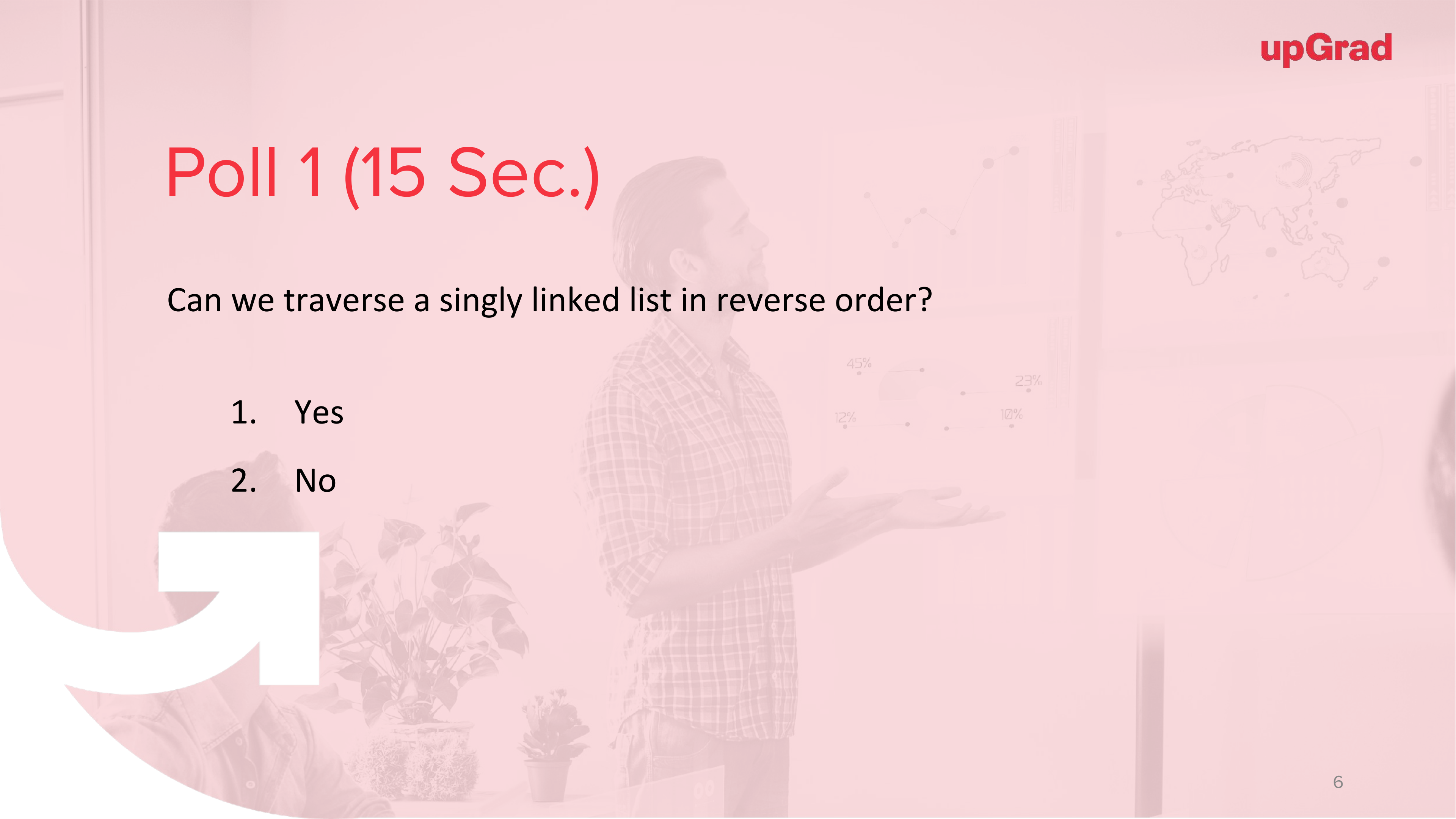
2 Doubly Linked List

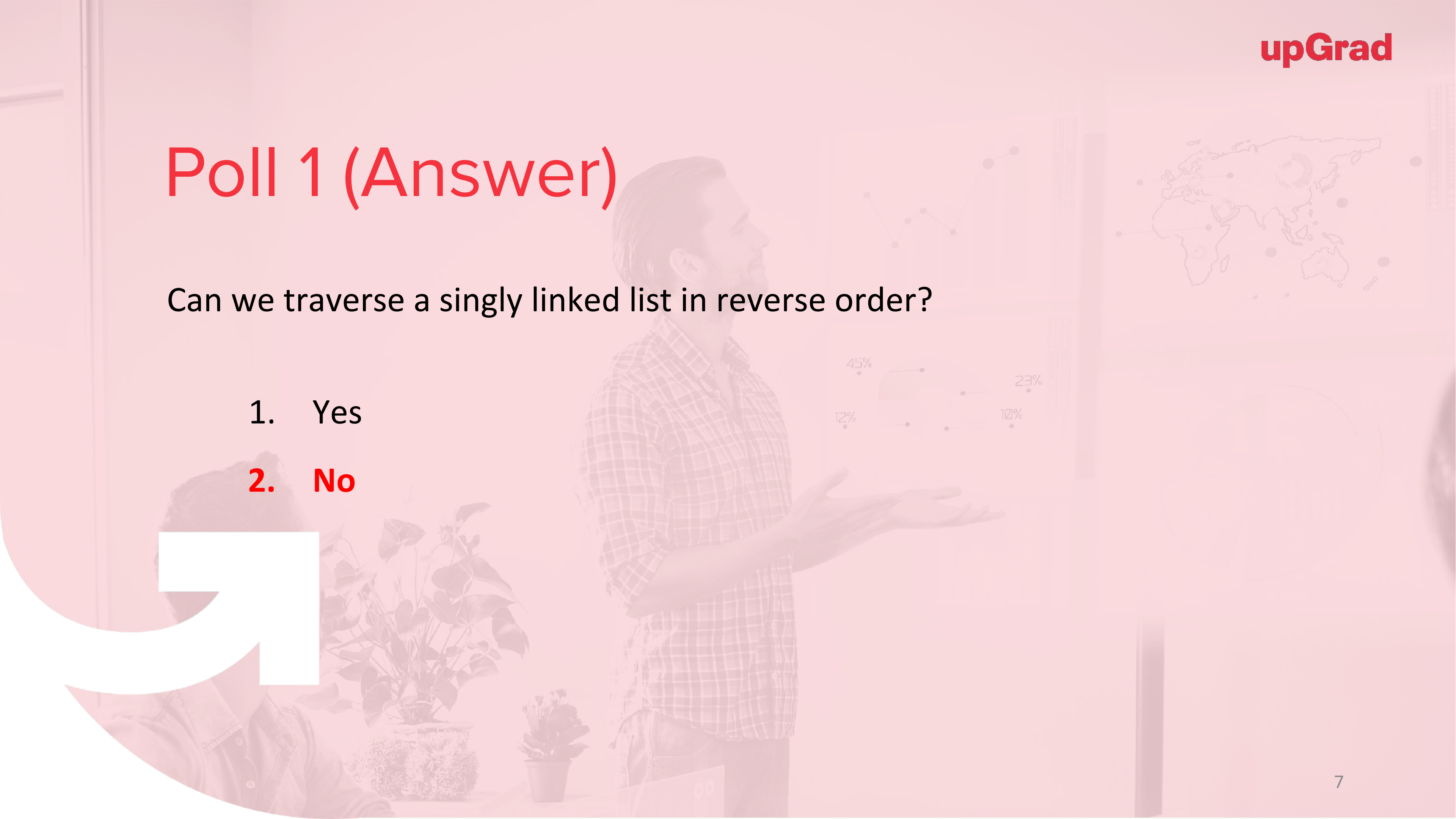


Poll 1 (15 Sec.)

Can we traverse a singly linked list in reverse order?

1. Yes
2. No





Poll 1 (Answer)

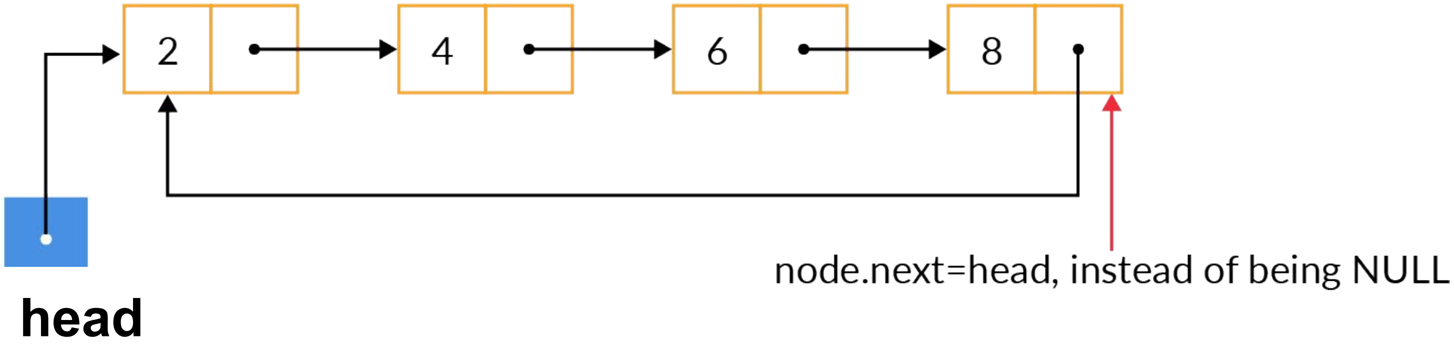
Can we traverse a singly linked list in reverse order?

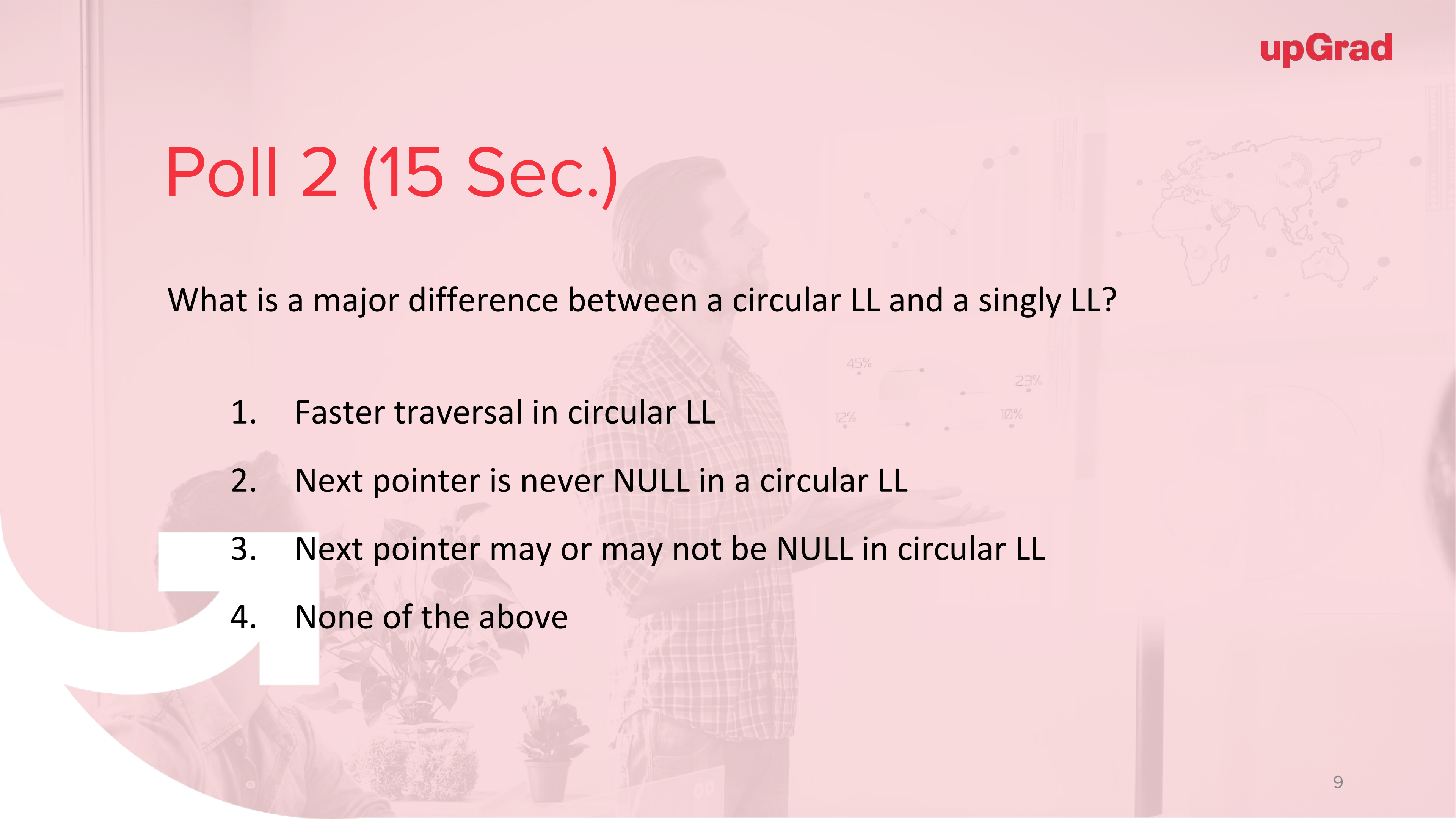
1. Yes
2. **No**

Circular Linked List

A slight variation of the traditional Linked List is something known as a Circular Linked List. In this variant, the last node in the list does not have its ‘pointer’ value as NULL.

Instead, it points to the first element of the list.



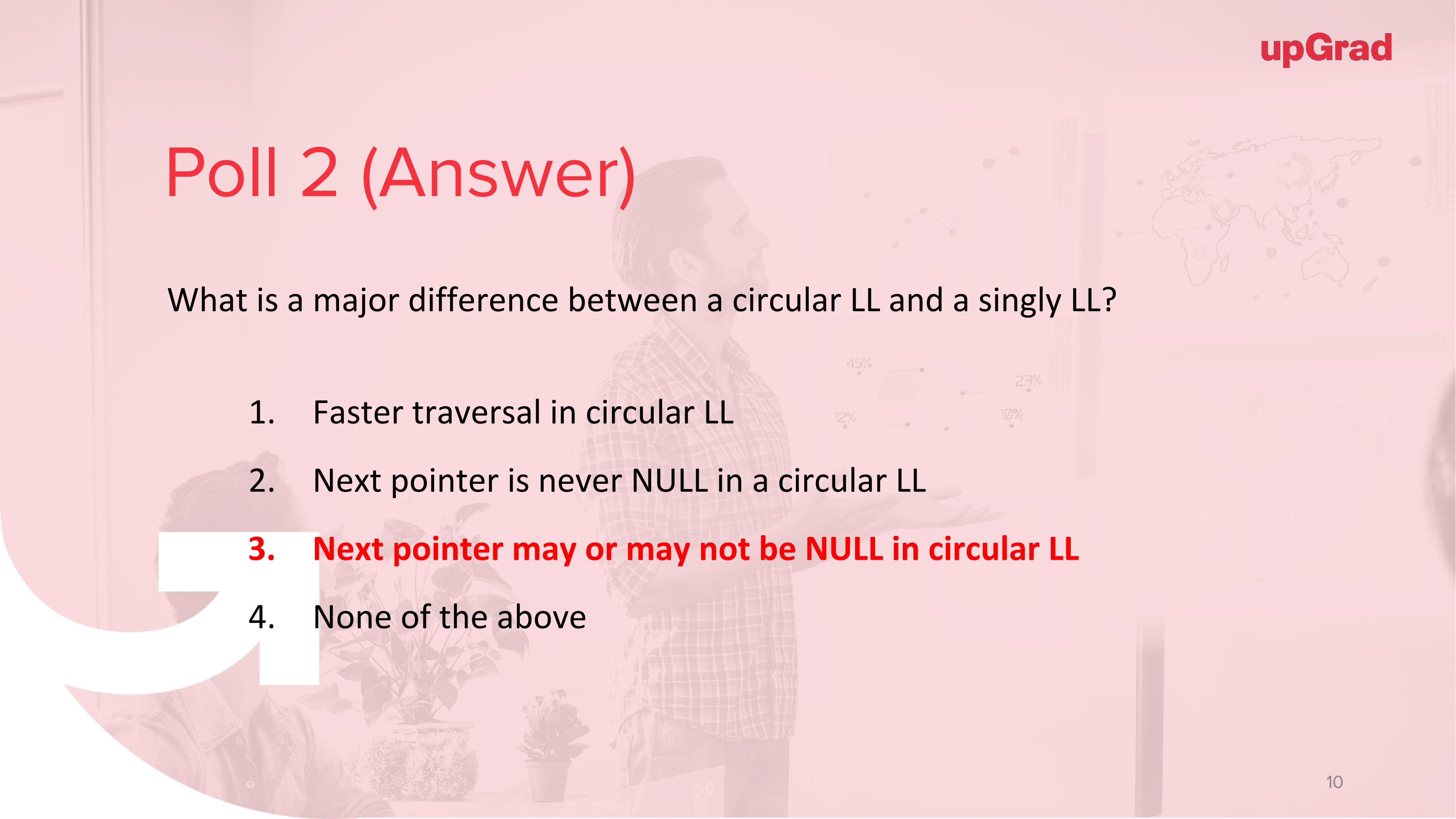


Poll 2 (15 Sec.)

What is a major difference between a circular LL and a singly LL?

1. Faster traversal in circular LL
2. Next pointer is never NULL in a circular LL
3. Next pointer may or may not be NULL in circular LL
4. None of the above

Poll 2 (Answer)



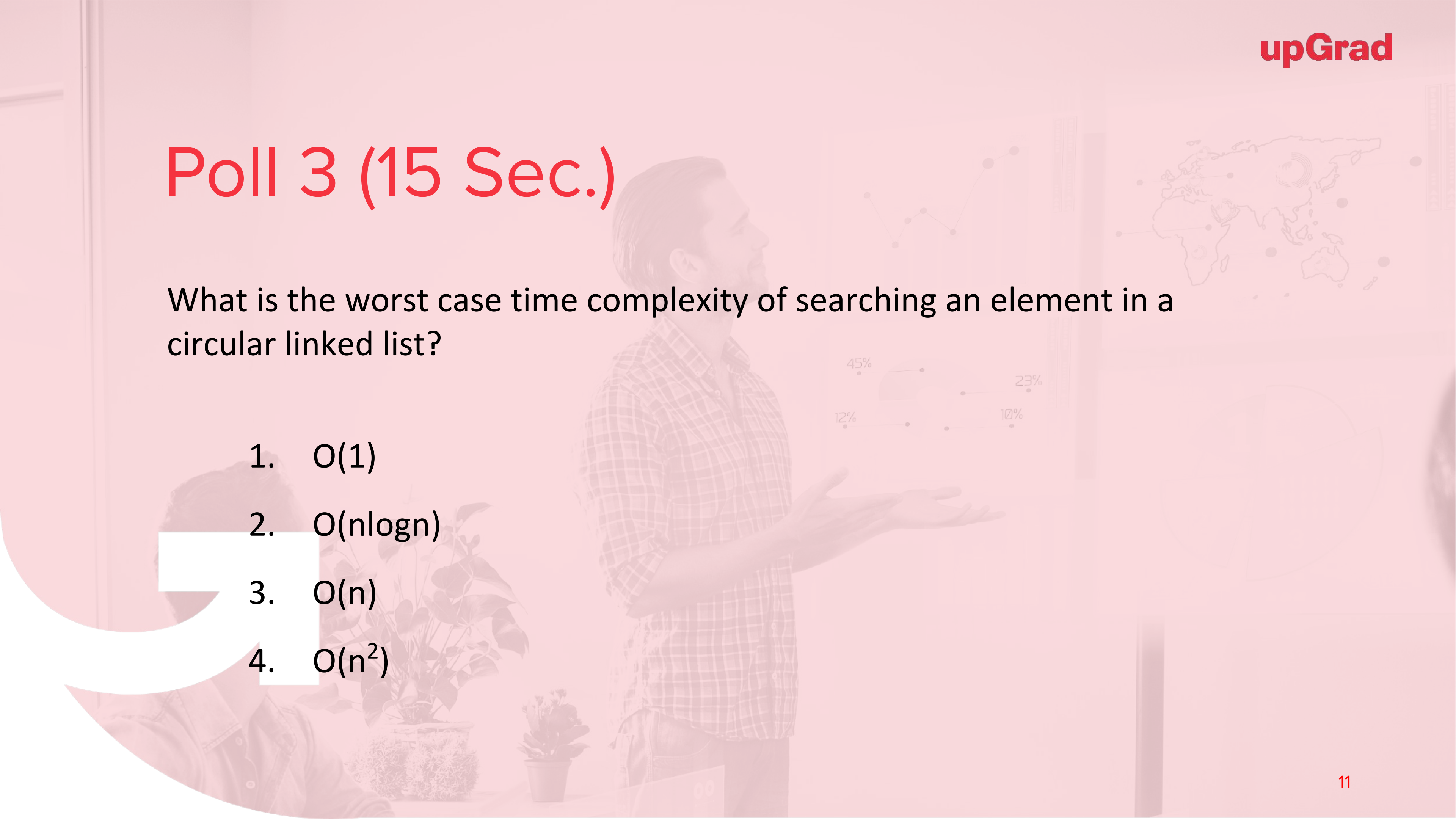
What is a major difference between a circular LL and a singly LL?

1. Faster traversal in circular LL
2. Next pointer is never NULL in a circular LL
3. **Next pointer may or may not be NULL in circular LL**
4. None of the above

Circular Linked List

Let us see how we can perform some basic operations on a circular linked list:

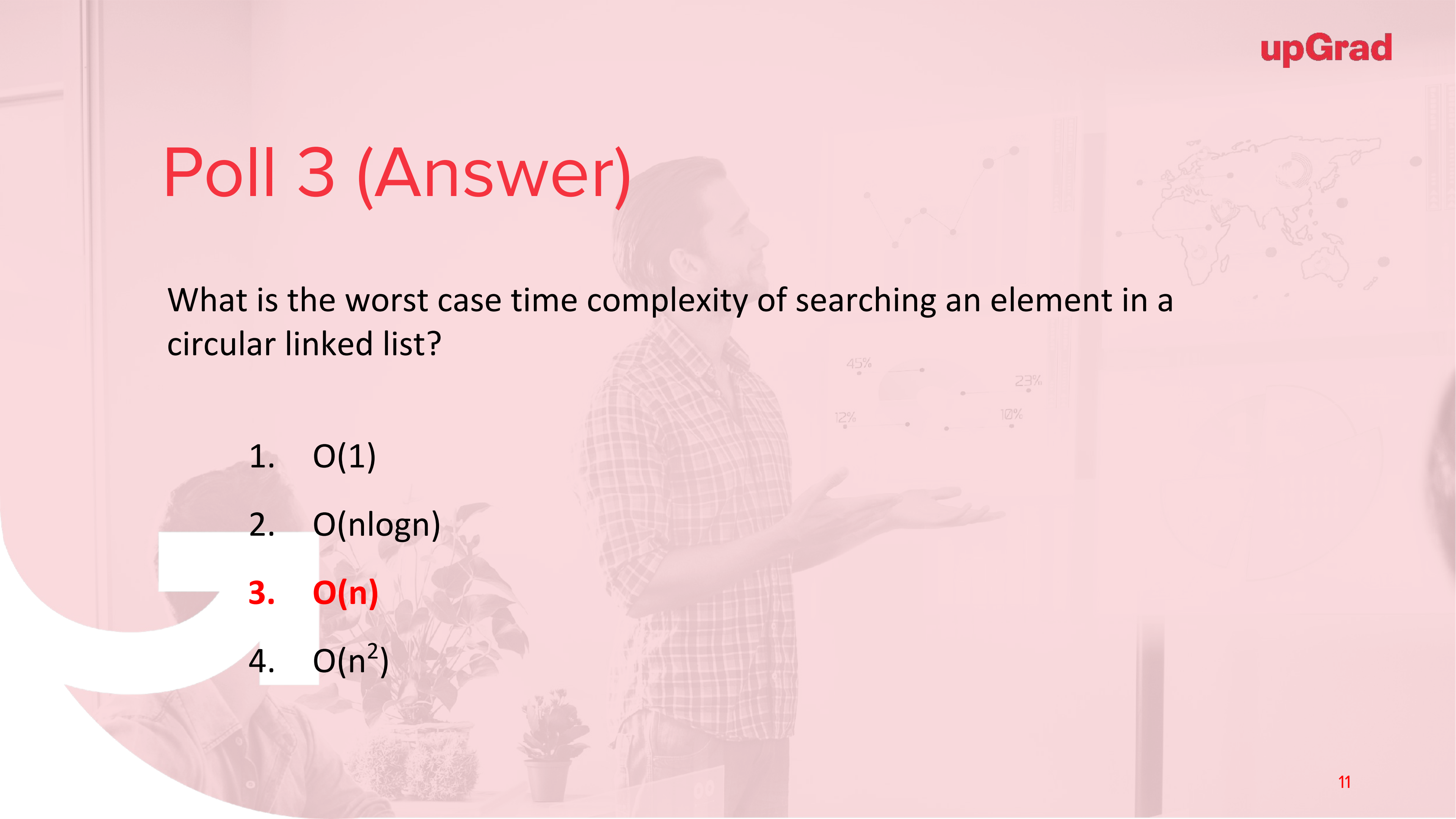
- Traverse a Circular LL
- Search for a node in Circular LL
- Insert a new node at the end of Circular LL
- Delete a node at the end in Circular LL



Poll 3 (15 Sec.)

What is the worst case time complexity of searching an element in a circular linked list?

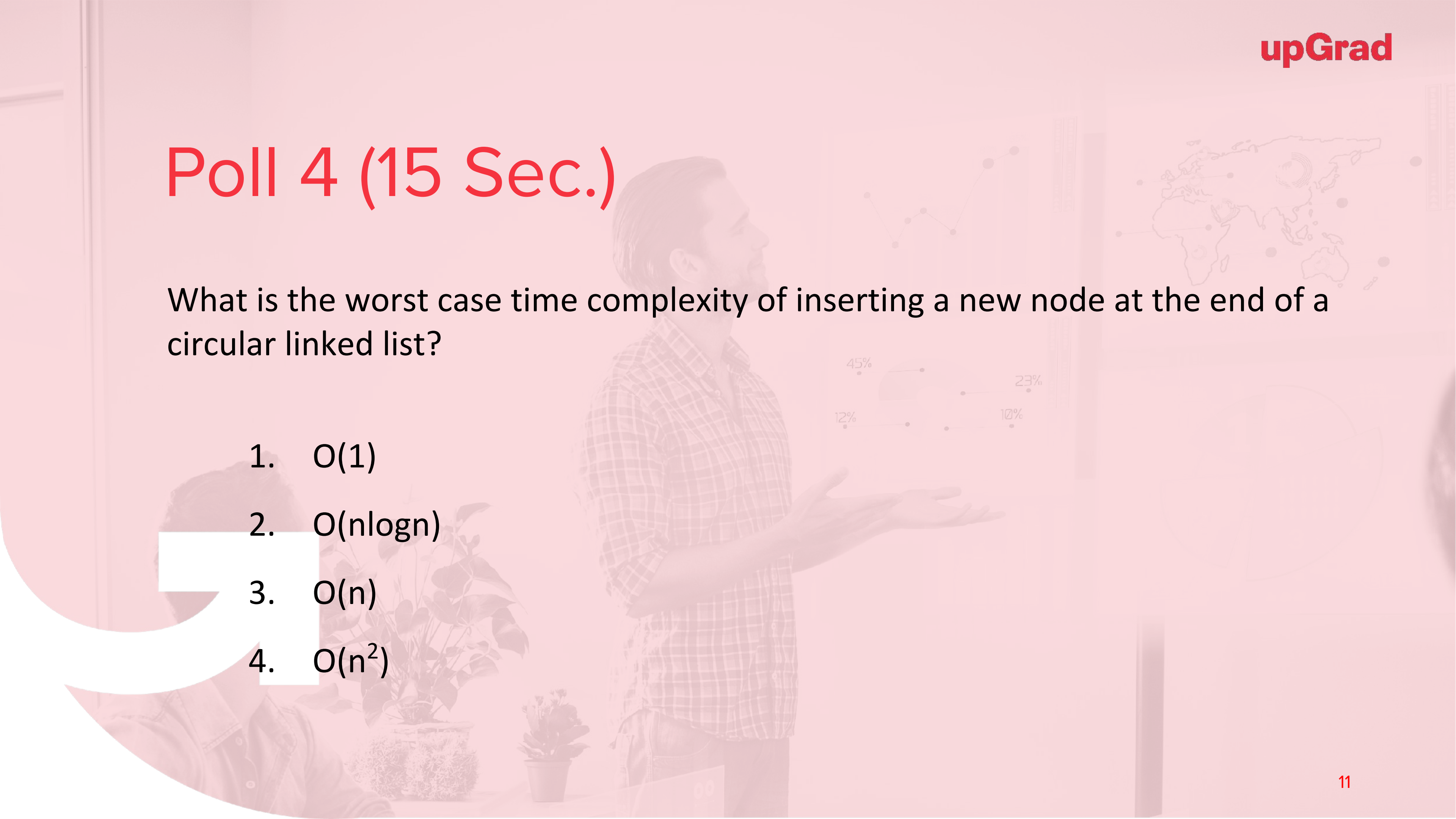
1. O(1)
2. O(nlogn)
3. O(n)
4. O(n^2)



Poll 3 (Answer)

What is the worst case time complexity of searching an element in a circular linked list?

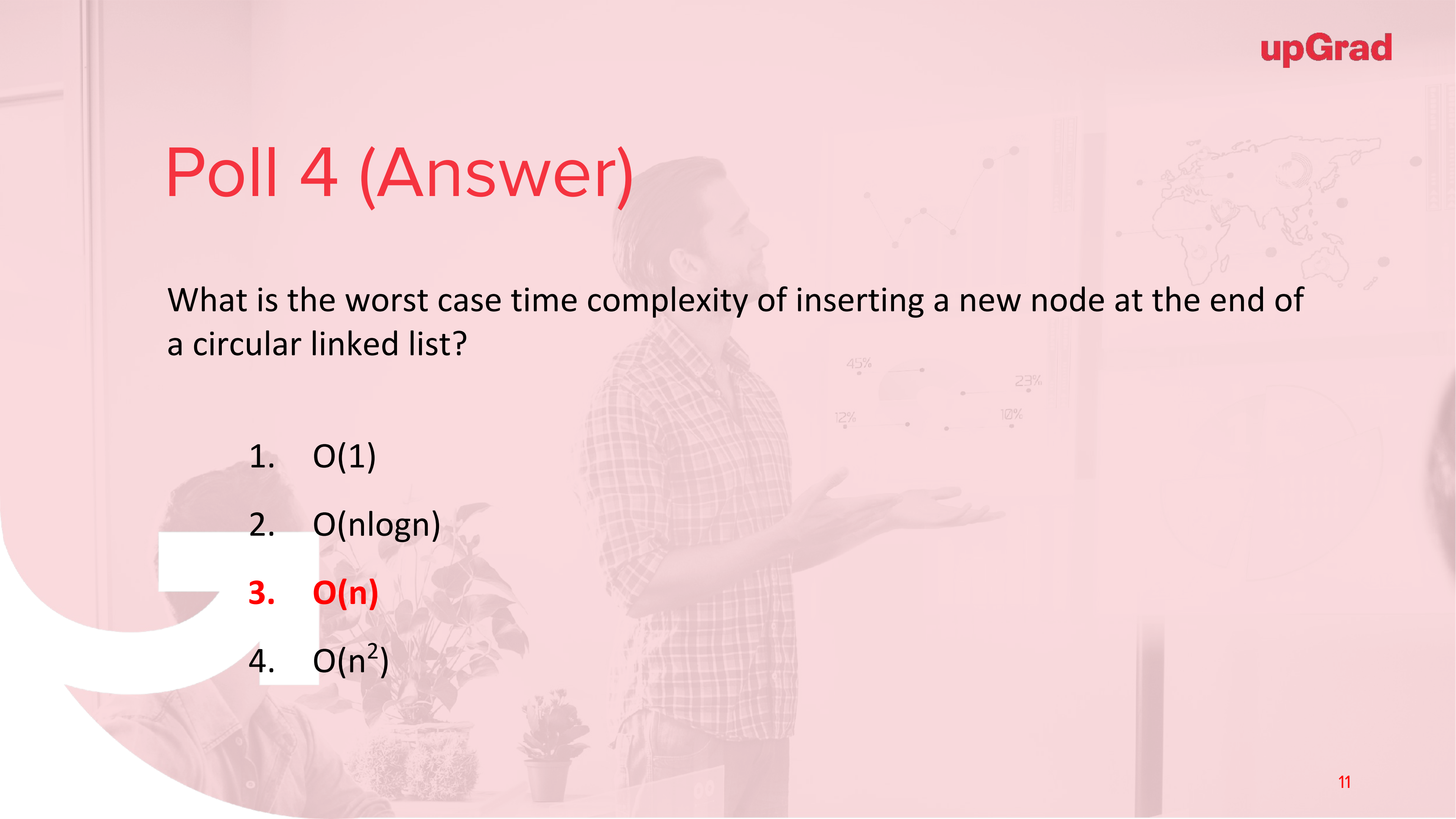
1. O(1)
2. O(nlogn)
3. **O(n)**
4. O(n^2)



Poll 4 (15 Sec.)

What is the worst case time complexity of inserting a new node at the end of a circular linked list?

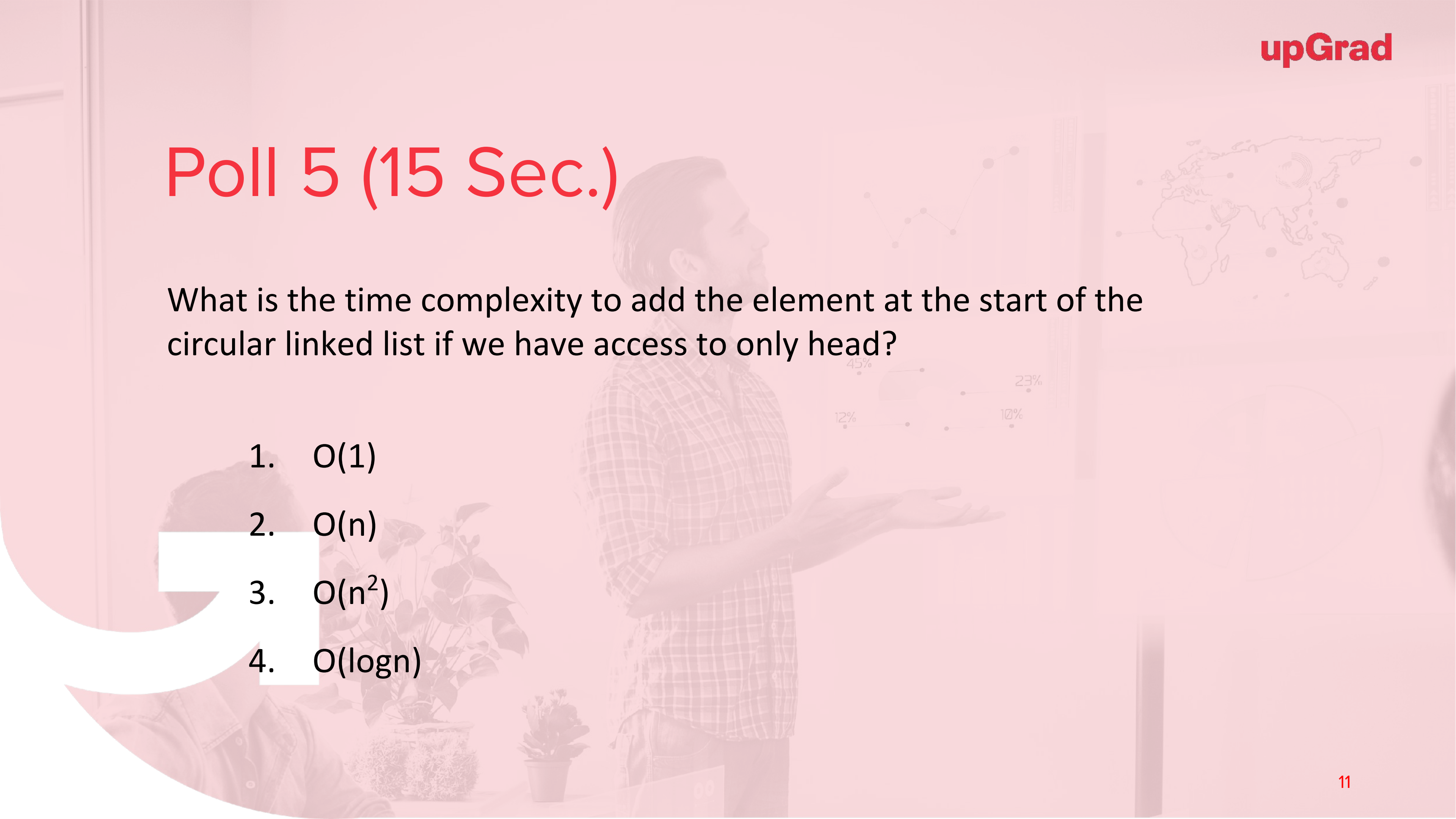
1. O(1)
2. O(nlogn)
3. O(n)
4. O(n^2)



Poll 4 (Answer)

What is the worst case time complexity of inserting a new node at the end of a circular linked list?

1. O(1)
2. O(nlogn)
3. **O(n)**
4. O(n^2)



Poll 5 (15 Sec.)

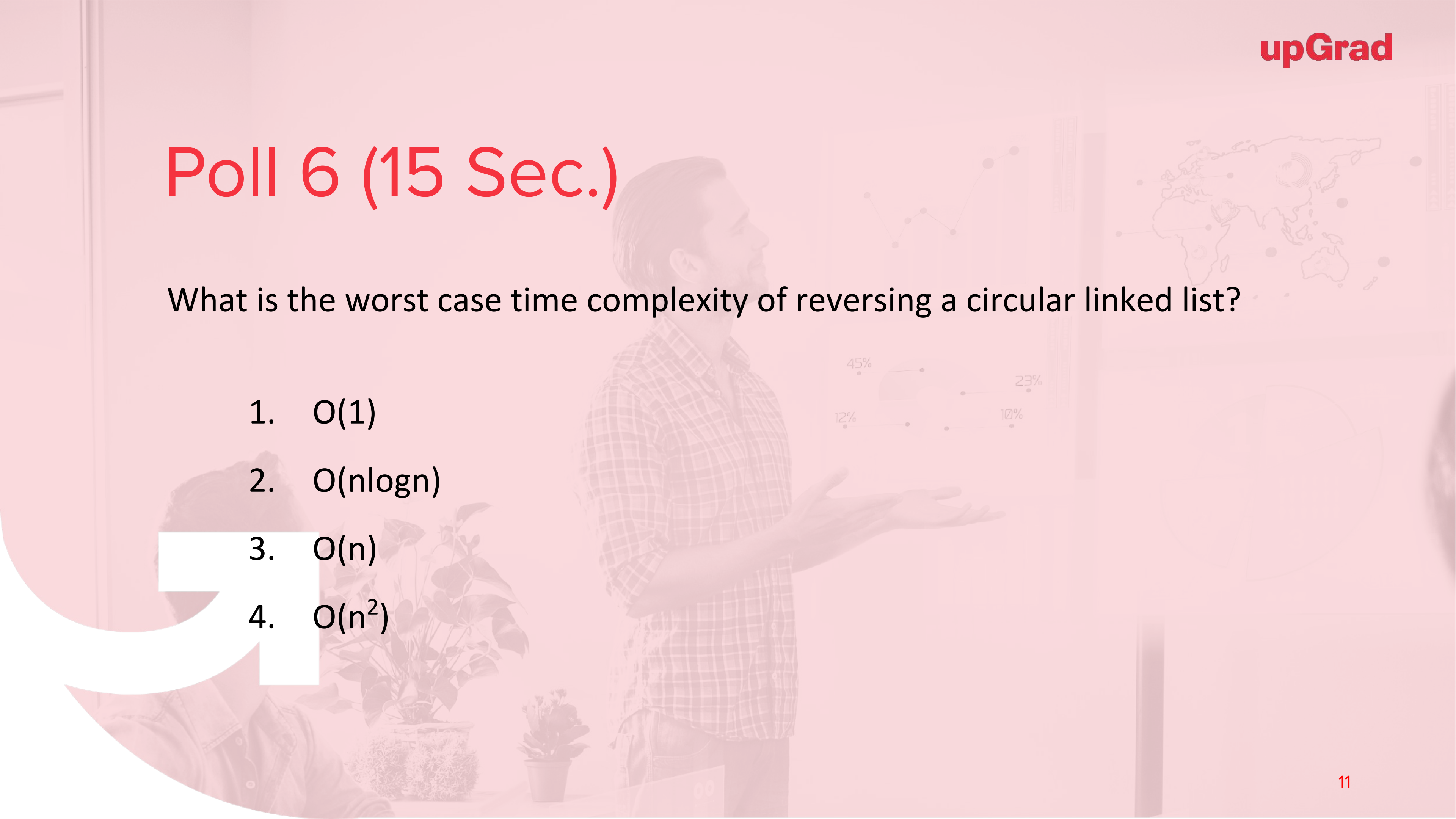
What is the time complexity to add the element at the start of the circular linked list if we have access to only head?

1. O(1)
2. O(n)
3. O(n^2)
4. O(logn)

Poll 5 (Answer)

What is the time complexity to add the element at the start of the circular linked list if we have access to only head?

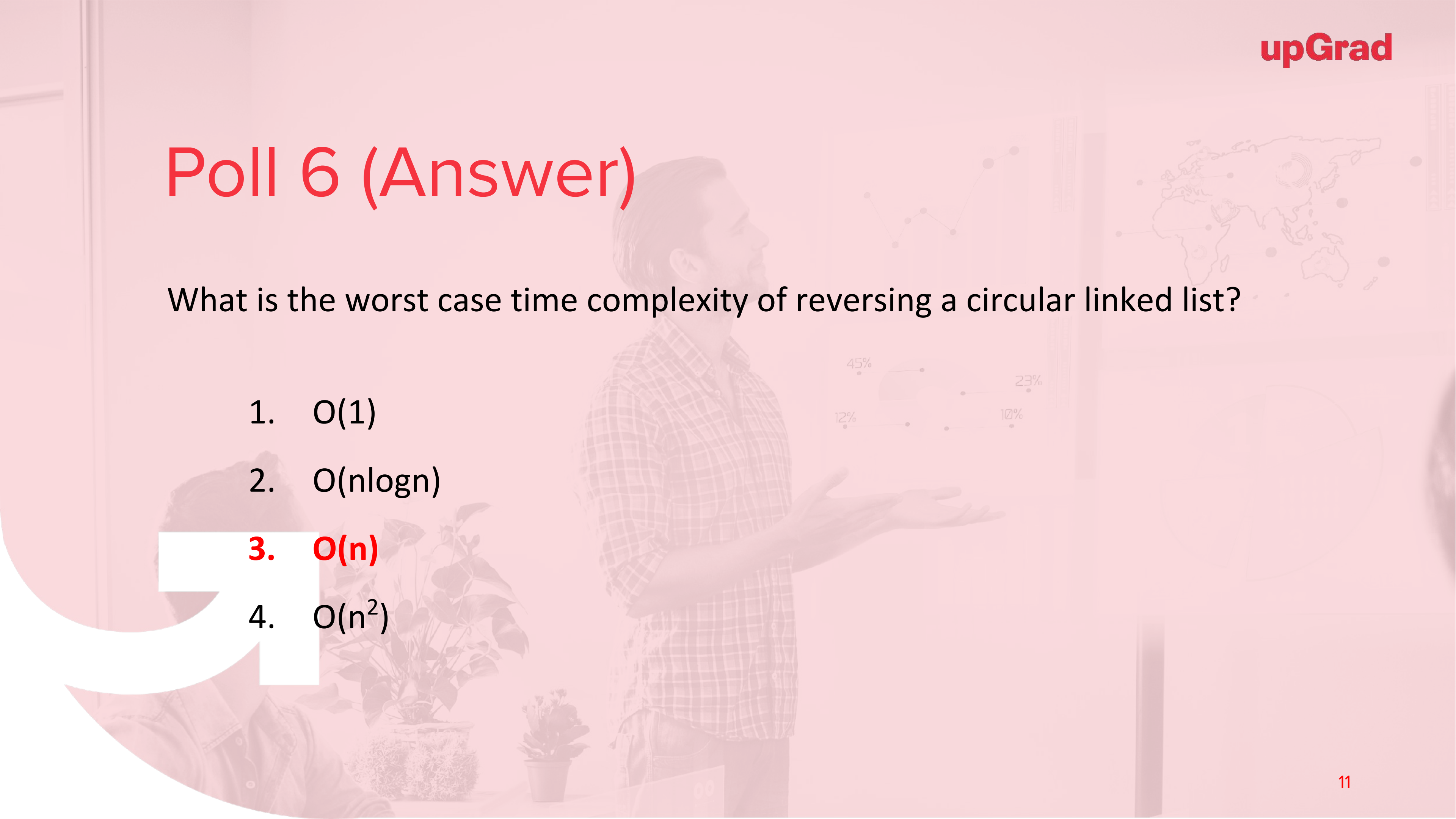
1. O(1)
2. **O(n)**
3. O(n^2)
4. O(logn)



Poll 6 (15 Sec.)

What is the worst case time complexity of reversing a circular linked list?

1. O(1)
2. O(nlogn)
3. O(n)
4. O(n^2)



Poll 6 (Answer)

What is the worst case time complexity of reversing a circular linked list?

1. O(1)
2. O(nlogn)
3. **O(n)**
4. O(n^2)

Doubly Linked List

upGrad

Doubly Linked List

Another powerful variation of the Linked List is something known as the Doubly Linked List. In this type of list, every data element is aware of the address of both its 'predecessor' and its 'successor'. This kind of construct makes the list more flexible and convenient to search and traverse through node.

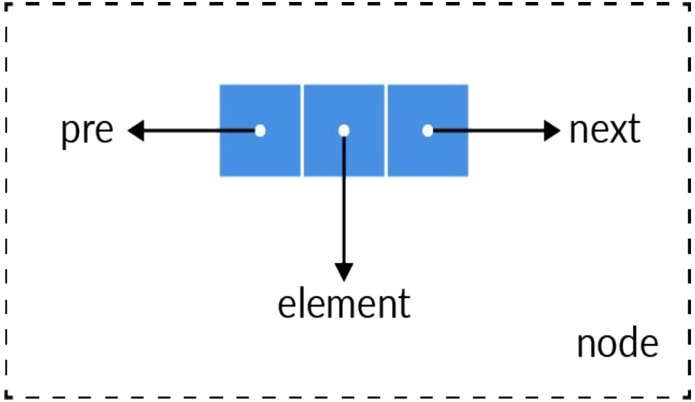
As we can understand, the nodes now need to store more information and, thus, will need more space.

In a doubly linked list, both the **'predecessor'** of the **first node** and the **'successor'** of the **last node** will contain **NULL**. We can also make a Circular Doubly Linked list to 'wrap' these free pointers around.

Doubly Linked List

For a doubly linked list, the structure of the Node will now need to change. Along with containing the data and the reference to the **next** Node, it will also have to contain a reference to the **previous** Node.

When we perform Linked List operations on a Doubly Linked List, we must carefully assign values to all the extra pointers that this variant brings in.



Doubly Linked List

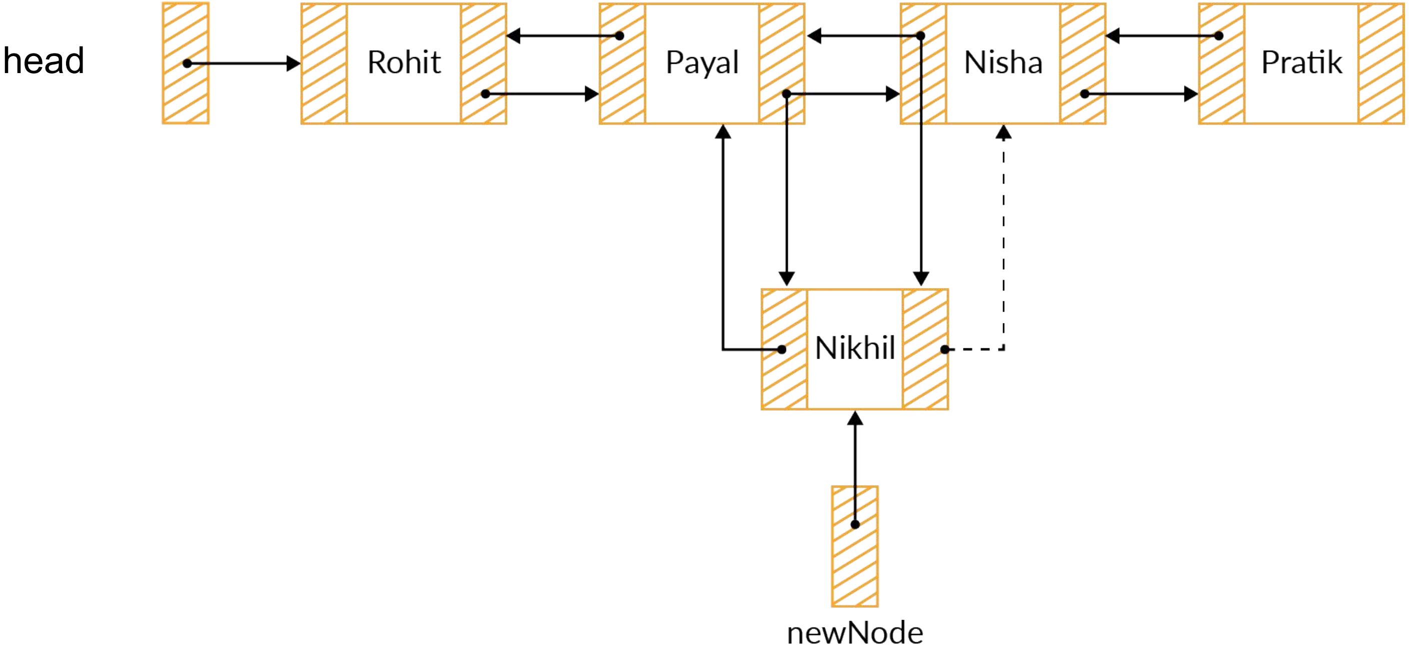
upGrad

Doubly Linked List Operations

As we have seen, we will need to perform a few extra steps to do basic operations on Doubly Linked Lists.

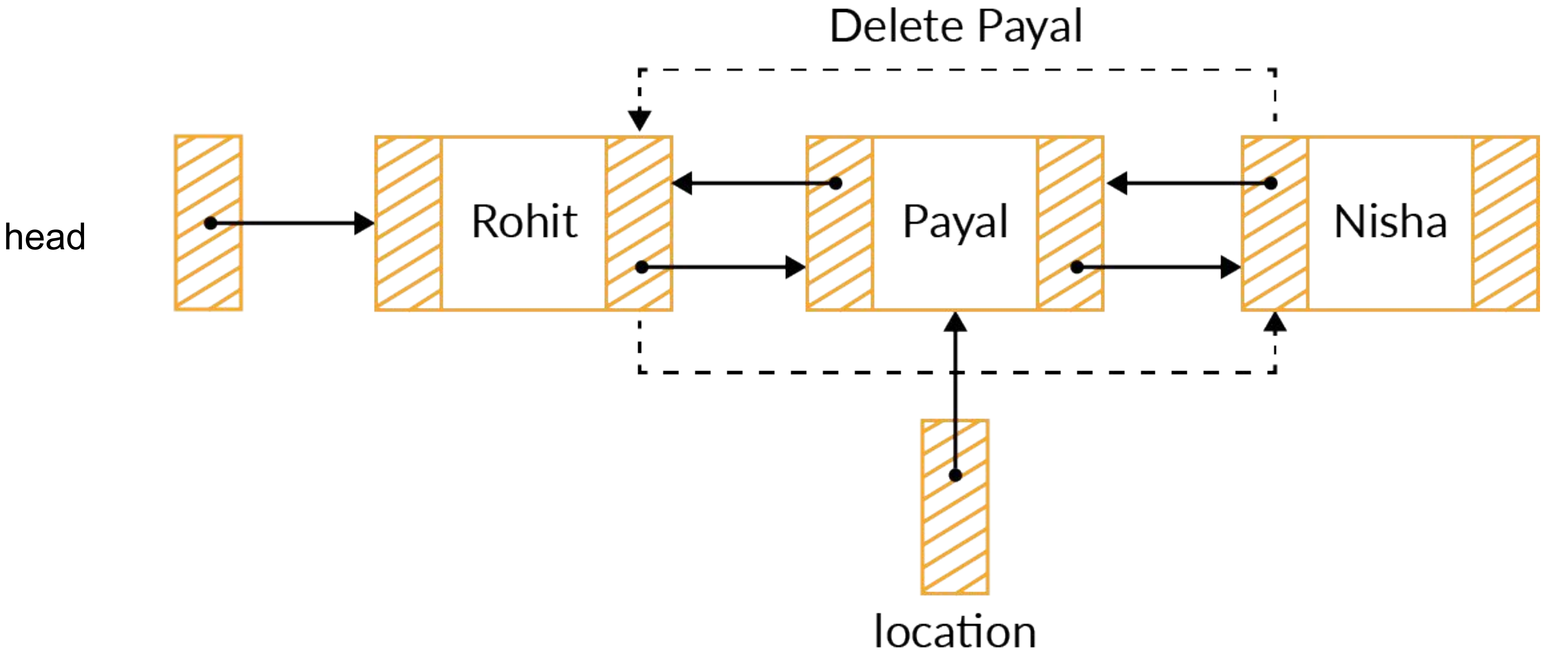
Insertion in Doubly Linked List

upGrad



Deletion in Doubly Linked List

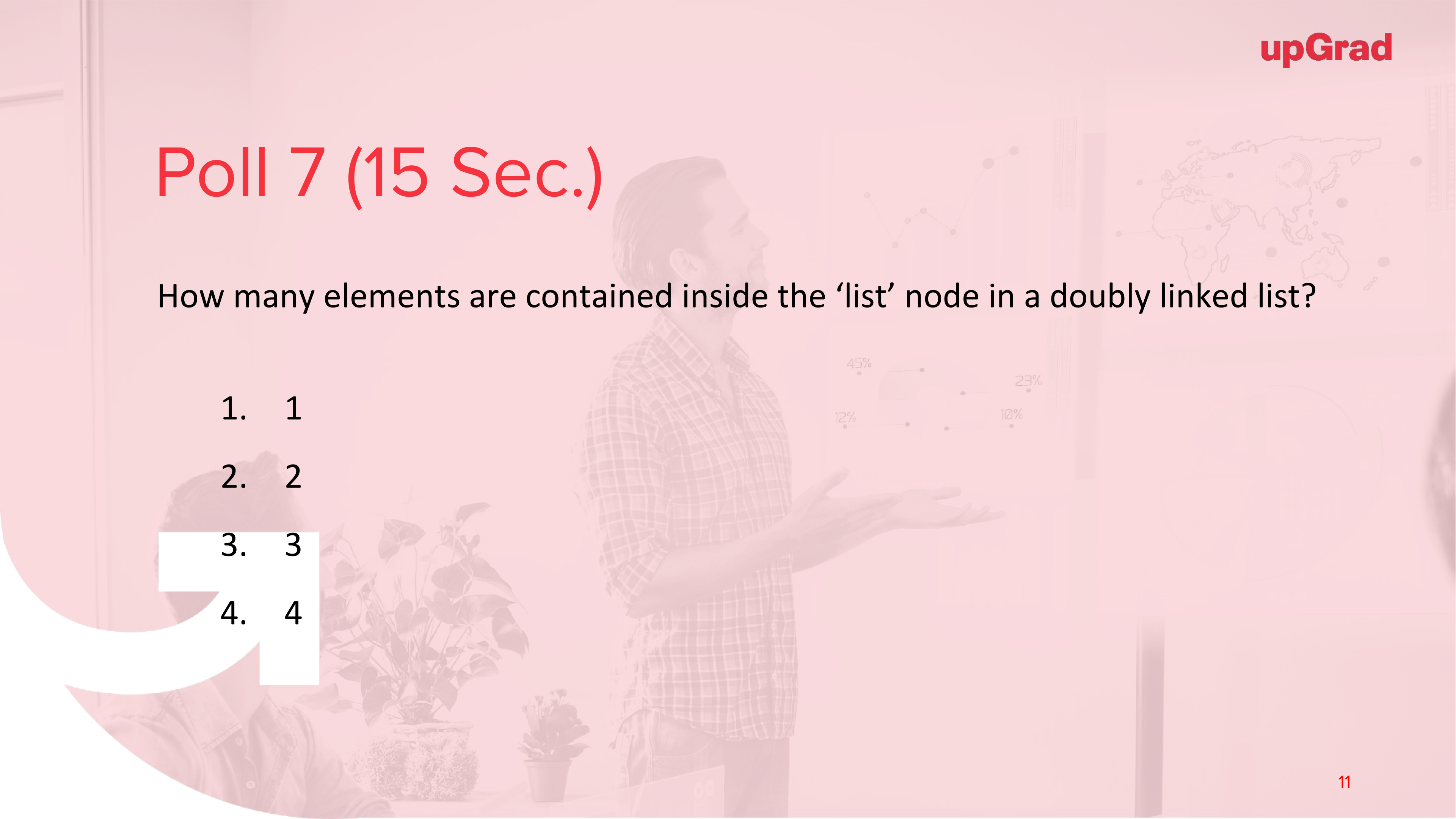
upGrad



Poll 7 (15 Sec.)

How many elements are contained inside the ‘list’ node in a doubly linked list?

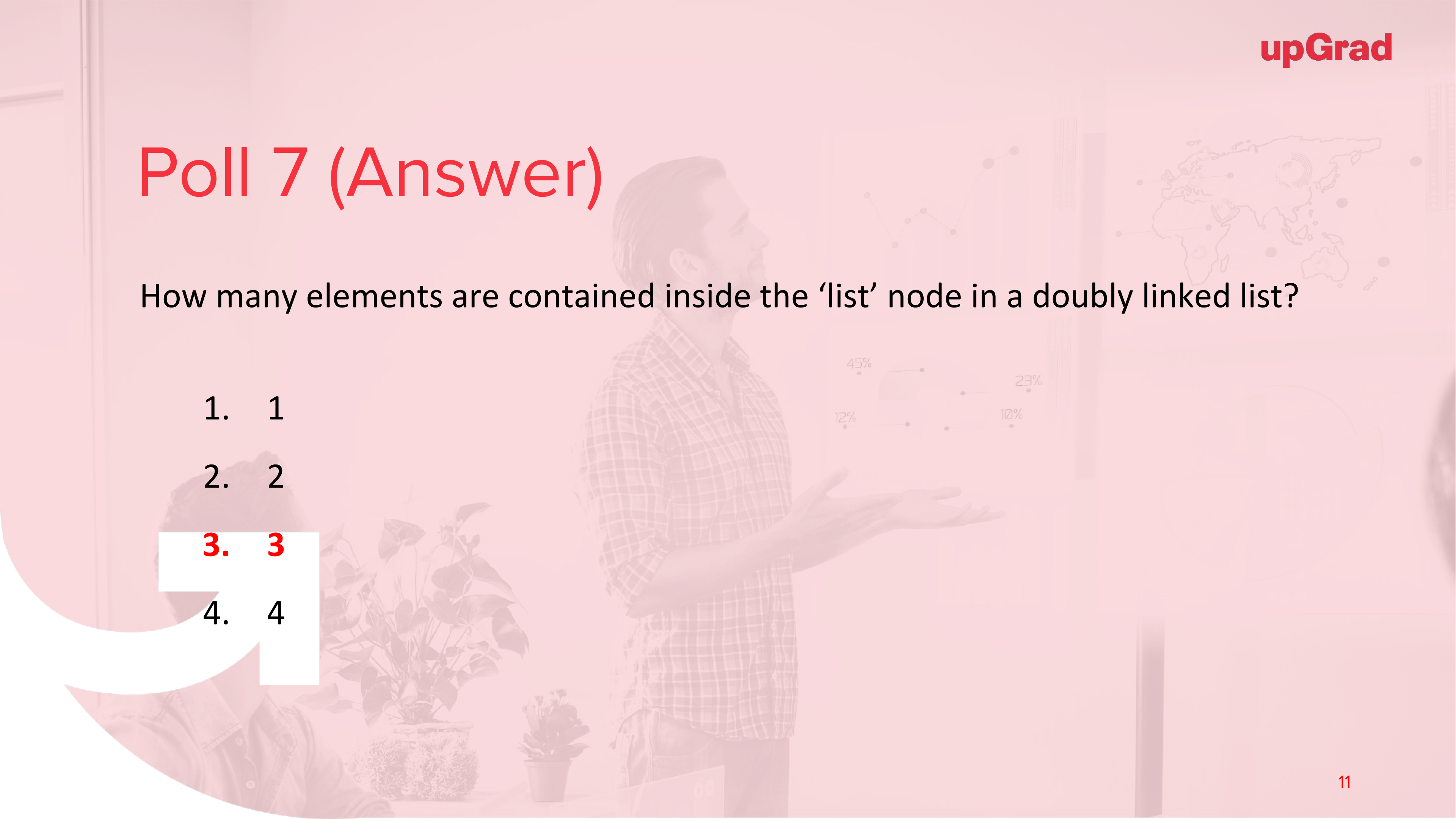
1. 1
2. 2
3. 3
4. 4

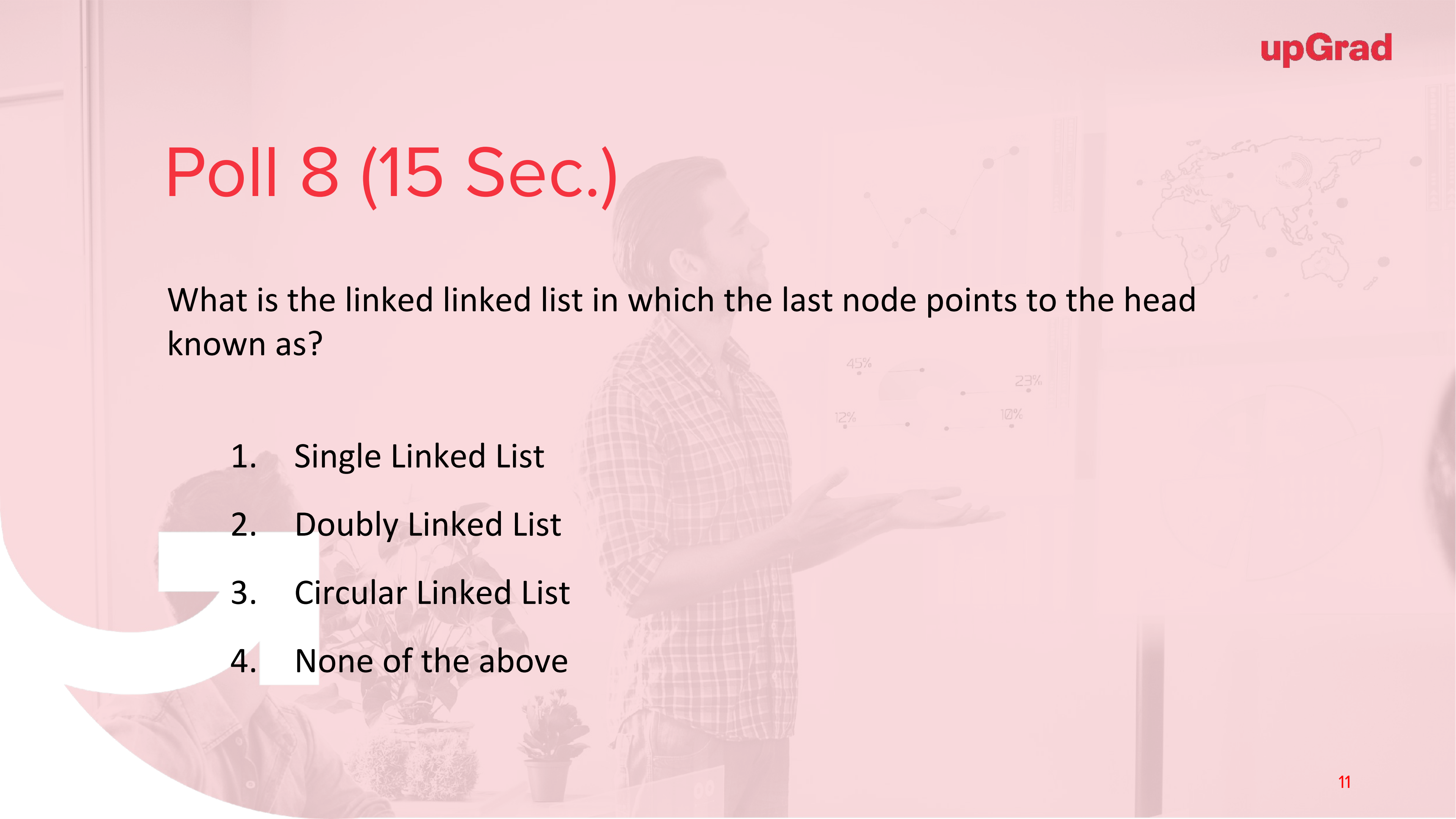


Poll 7 (Answer)

How many elements are contained inside the ‘list’ node in a doubly linked list?

1. 1
2. 2
3. **3**
4. 4

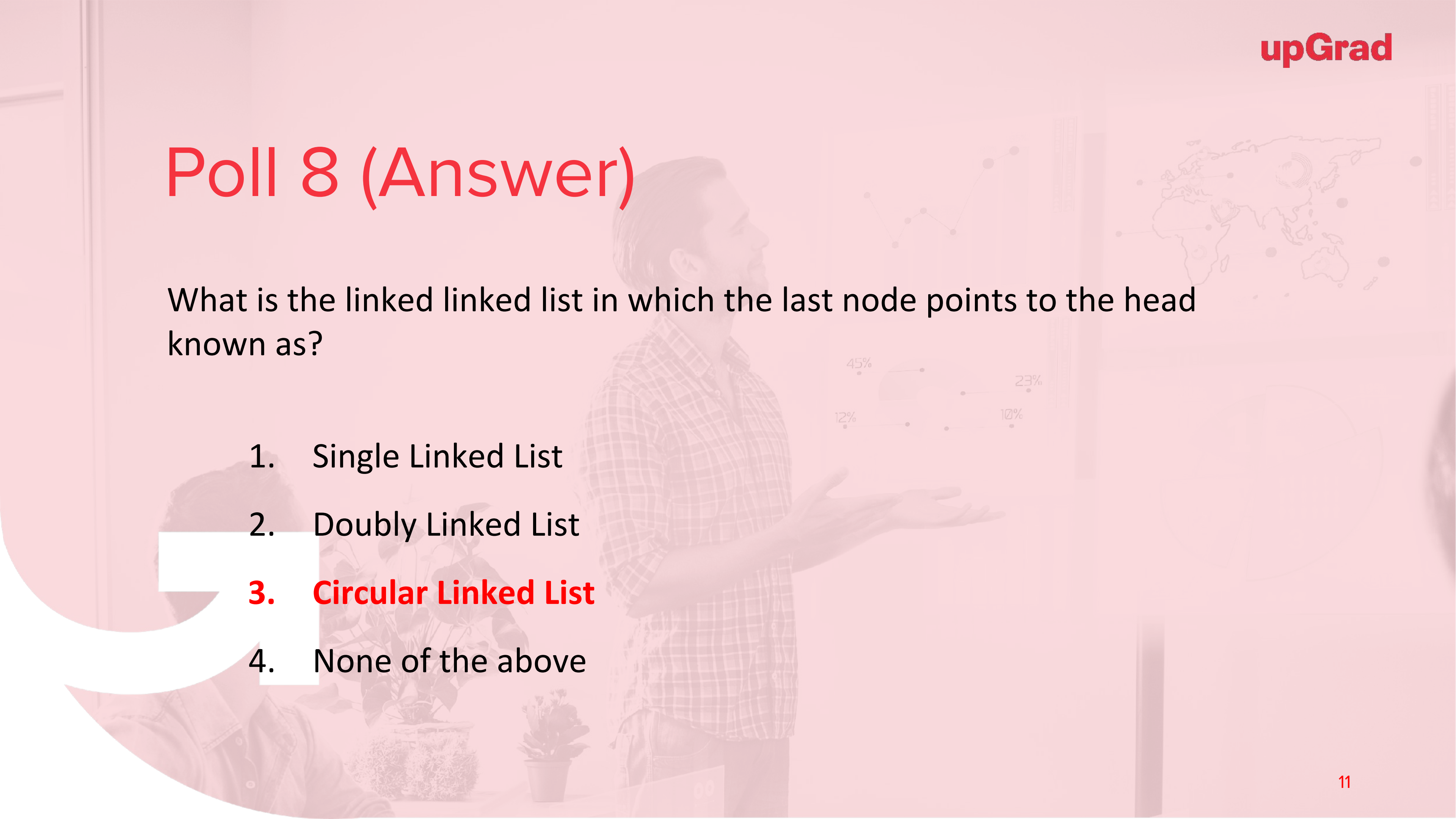




Poll 8 (15 Sec.)

What is the linked linked list in which the last node points to the head known as?

1. Single Linked List
2. Doubly Linked List
3. Circular Linked List
4. None of the above



Poll 8 (Answer)

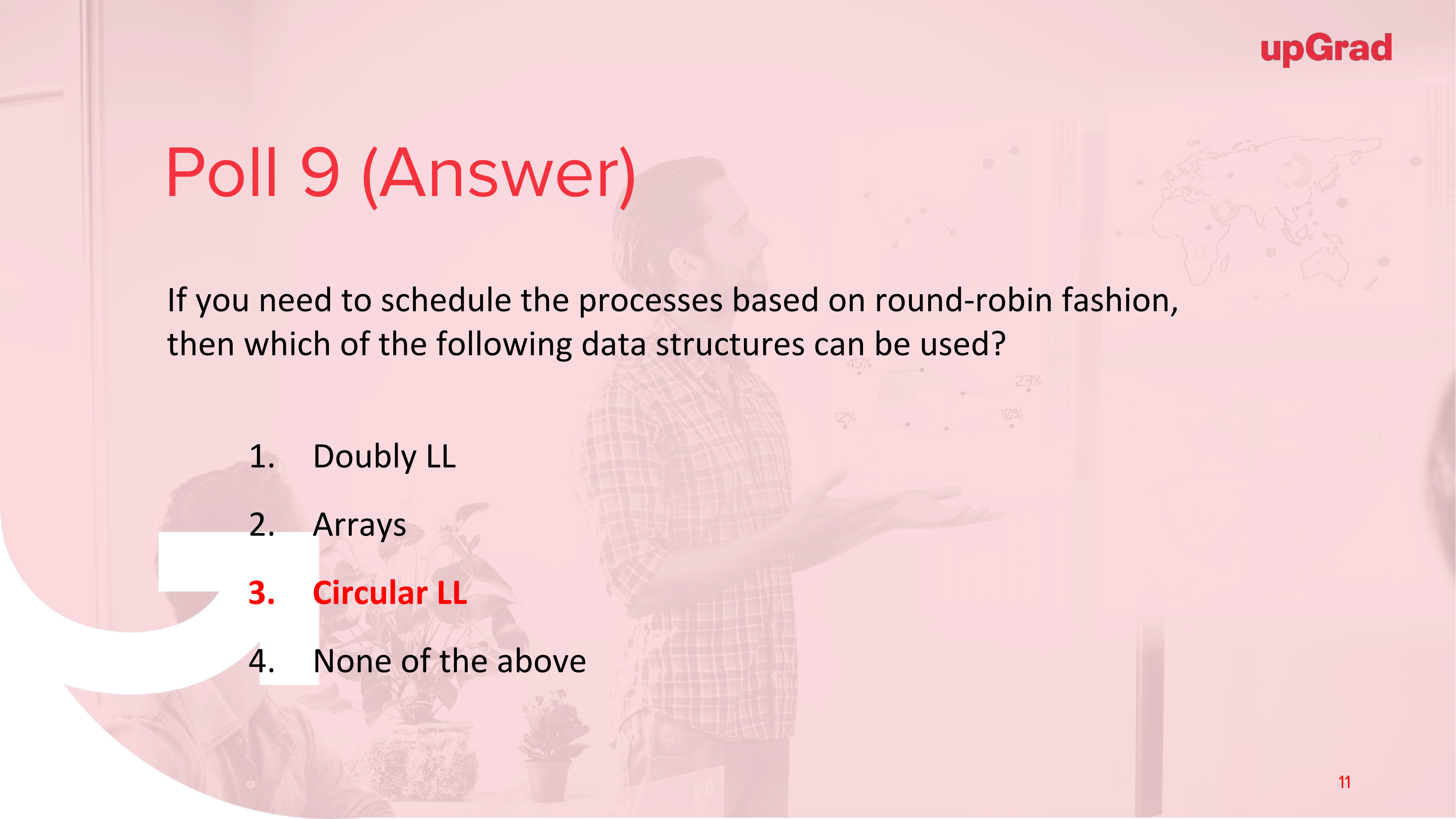
What is the linked linked list in which the last node points to the head known as?

1. Single Linked List
2. Doubly Linked List
3. **Circular Linked List**
4. None of the above

Poll 9 (15 Sec.)

If you need to schedule the processes based on round-robin fashion, then which of the following data structures can be used?

1. Doubly LL
2. Arrays
3. Circular LL
4. None of the above

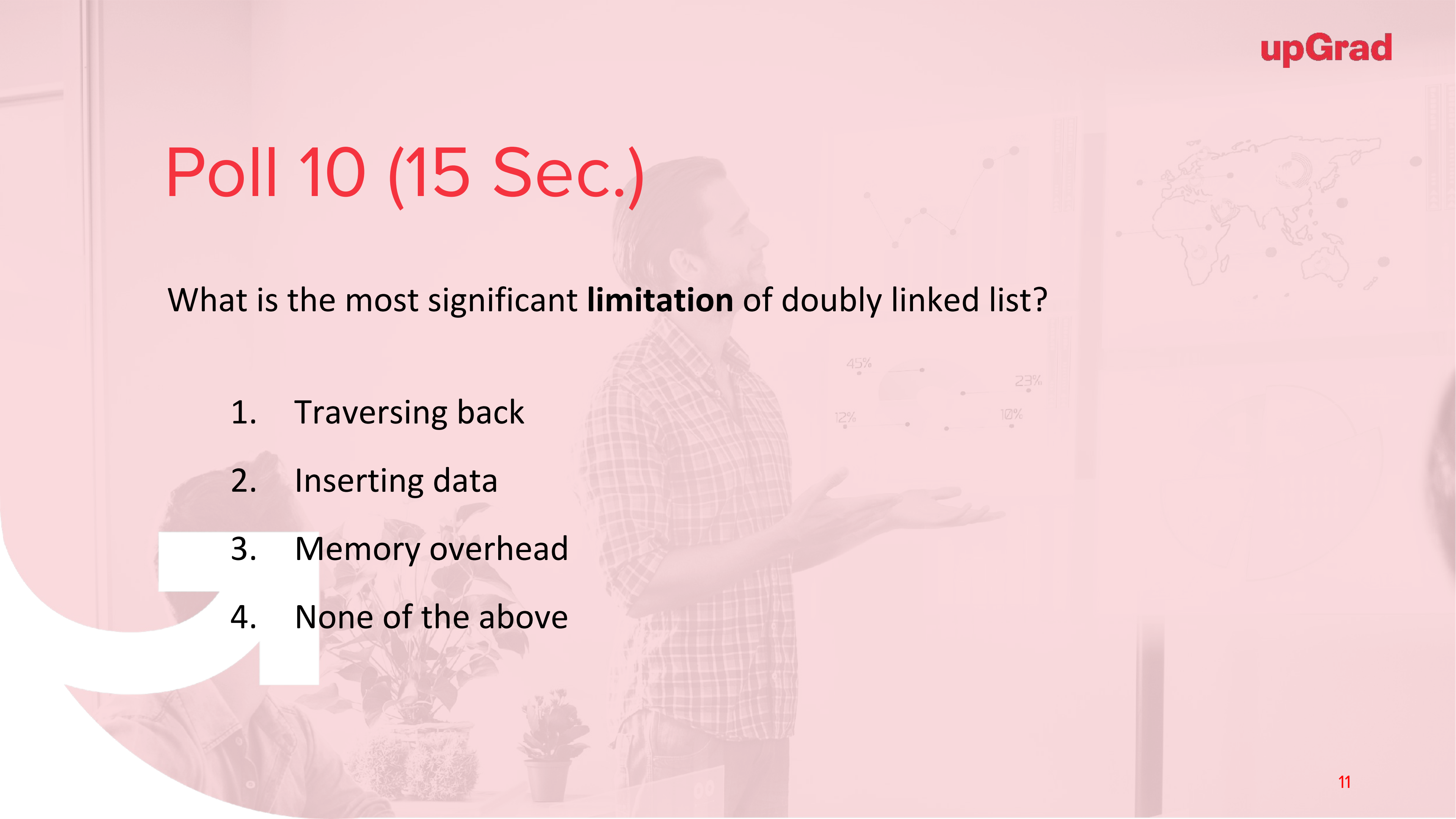


Poll 9 (Answer)

If you need to schedule the processes based on round-robin fashion, then which of the following data structures can be used?

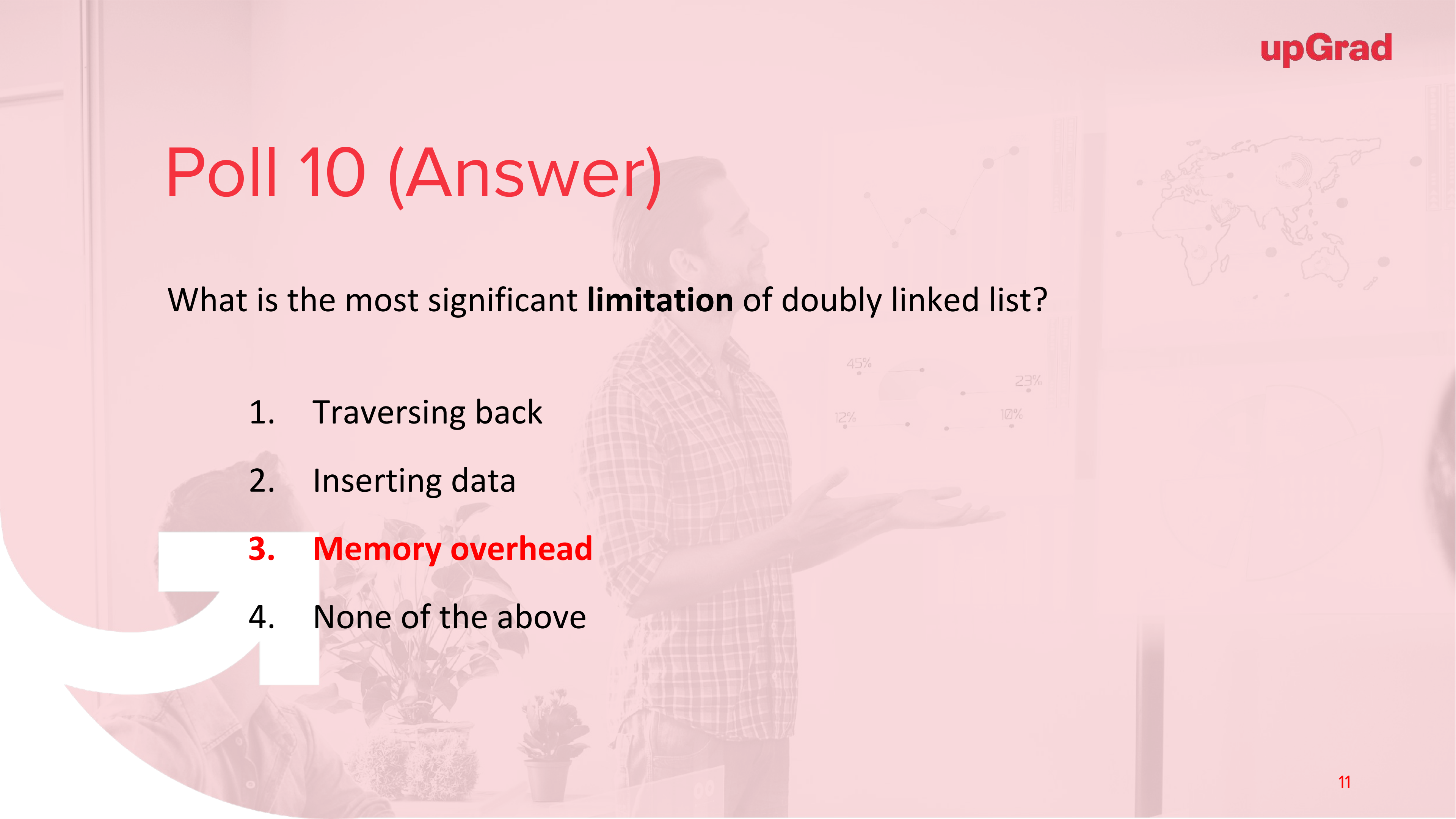
1. Doubly LL
2. Arrays
3. **Circular LL**
4. None of the above

Poll 10 (15 Sec.)



What is the most significant **limitation** of doubly linked list?

1. Traversing back
2. Inserting data
3. Memory overhead
4. None of the above



Poll 10 (Answer)

What is the most significant **limitation** of doubly linked list?

1. Traversing back
2. Inserting data
3. **Memory overhead**
4. None of the above

Hands-on Coding

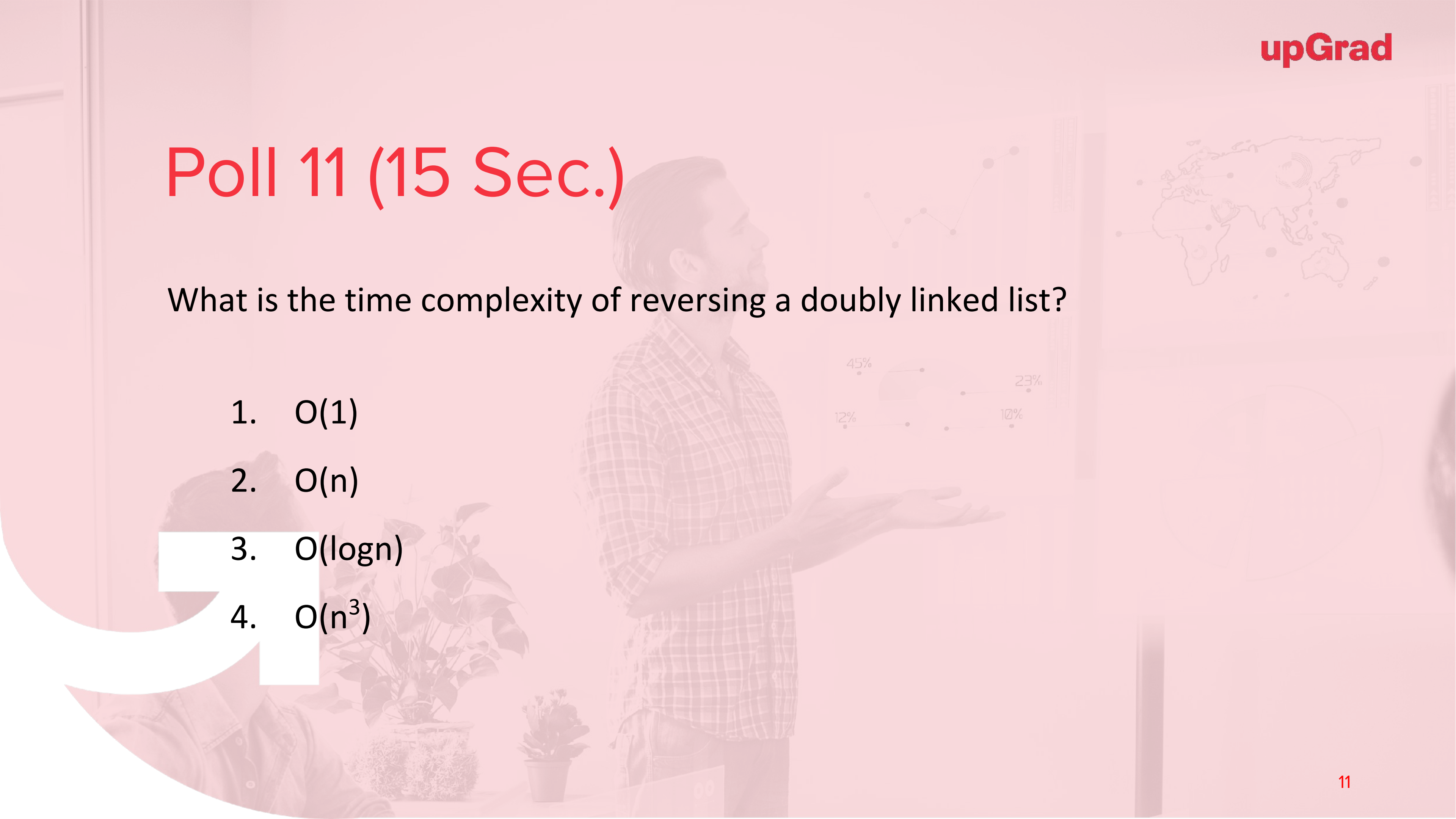
Write Java program to:

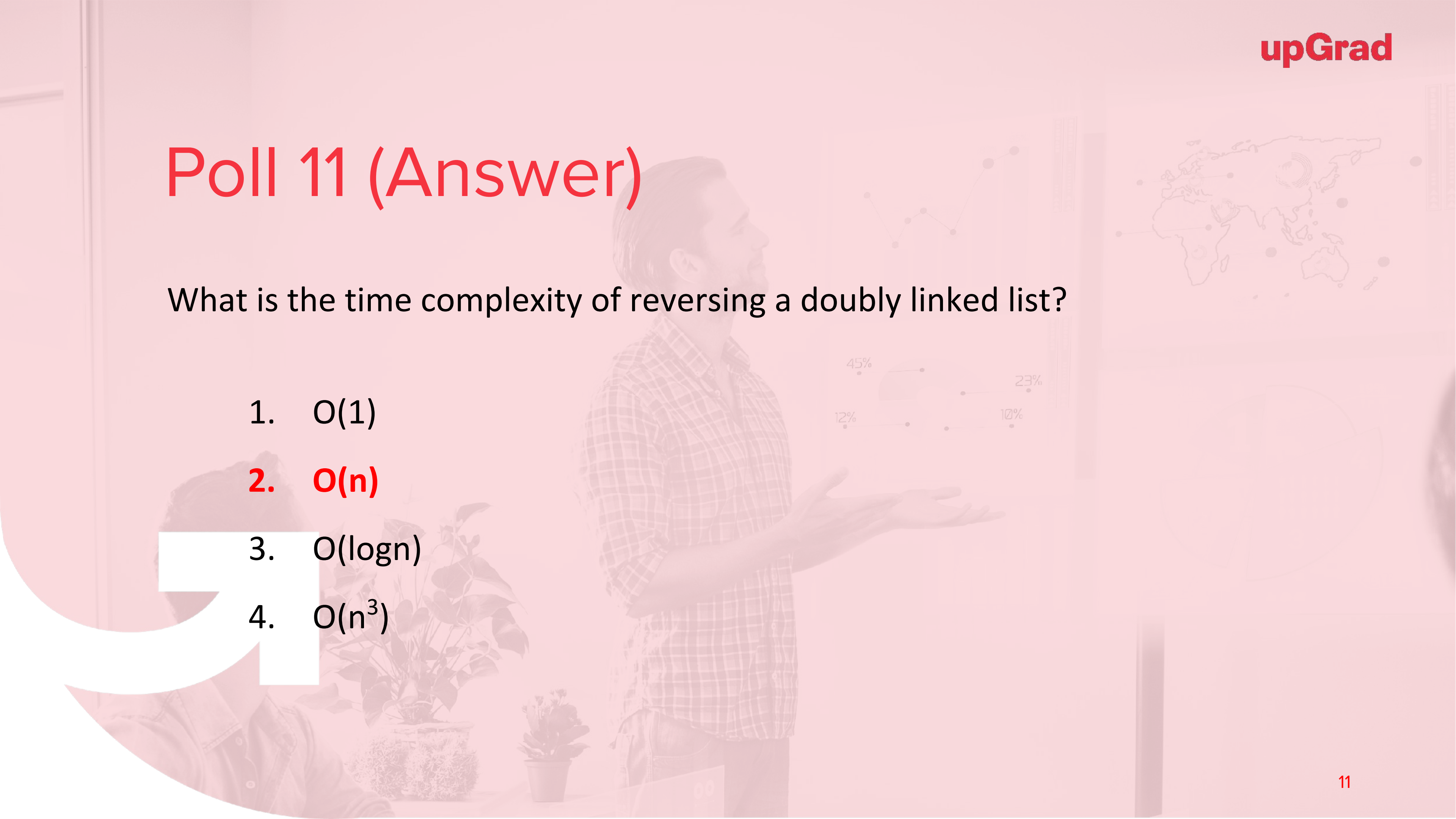
- Traverse a DLL
- Search for a node in DLL
- Insert a node in DLL
- Delete a node in DLL

Poll 11 (15 Sec.)

What is the time complexity of reversing a doubly linked list?

1. O(1)
2. O(n)
3. O(logn)
4. O(n^3)





Poll 11 (Answer)

What is the time complexity of reversing a doubly linked list?

1. O(1)
2. **O(n)**
3. O(logn)
4. O(n^3)

Homework

1. Reverse a Circular LL

Tasks to complete after the session

Homework Questions
MCQs
Coding Questions

In the next class...

- Stacks

upGrad

#RahoAmbitious

Thank You!