 Data Structures and Algorithm | Jan 2021​

# Assignment 4 | 21st January 2021

**For any doubts regarding the assignment, ask questions in the** [**Dat**](https://community.letsupgrade.in/group/dsaes0121b2)​ [**a**](https://community.letsupgrade.in/group/dsaes0121b2)

[**Structures and Algorithms**](https://community.letsupgrade.in/group/dsaes0121b2) **Group**​ ​ **in the Community.**​

**Submit Assignments by** ​ **26**​ **th January 2021 11:59 PM**

## Assignment Submit Form: [https://forms.gle/bJBQwoRVk4P8SR4T](https://forms.gle/bJBQwoRVk4P8SR4T8)​ [8](https://forms.gle/bJBQwoRVk4P8SR4T8)

**Submit assignments in Appropriate Dropdowns.**

**Question 1**

Implement deletion operation from the end of the linked list and Insertion operation from the beginning of the linked list

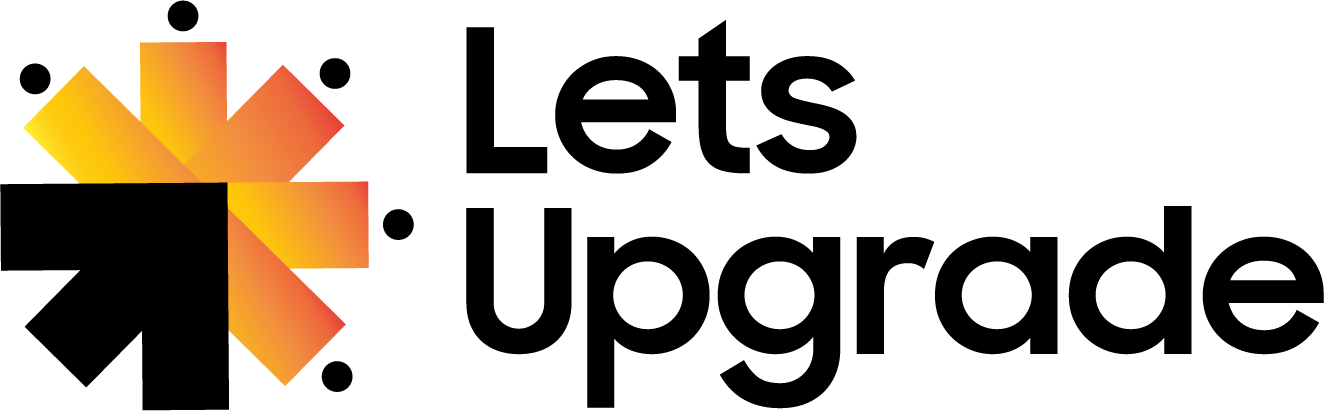
**Question 2**

Implement binary search using python language.

(Write a function which returns the index of x in given array arr if present, else returns -1)

**Question 3**

Write a Python program to find the middle of a linked list.

 Data Structures and Algorithm | Jan 2021​

## FAQs

**Q. When do I submit the Assignments and how?**

1. The assignments for the week should be submitted by 26th January 2021 i.e.Tuesday 11:59 PM IST.
2. You need to submit the answers in Document Format

**Q. Where do I get class links for the next session?**

1. All sessions will be Live on our Youtube Channel. Subscribe to LetsUpgrade​​[YouTube Channel](https://www.youtube.com/channel/UCWUDiLzQZr4VDHNyMsVYn-g)​[.](https://www.youtube.com/channel/UCWUDiLzQZr4VDHNyMsVYn-g)

You'll also get an email with the link to the live session.

1. It will be also updated in the Community Group in the pinned post.

**Q. I have some doubt, who do I ask?**

A. Post your Queries on the community, someone will help you out.

**Q. How can we know if my assignment is verified or not? And is it successfully submitted or not?**

A. You will receive a mail for your successful submission.

ANSWER:

QUESTION 3:

**class Node:**

**def \_\_init\_\_(self, data):**

**self.data = data**

**self.next = None**

**class LinkedList:**

**def \_\_init\_\_(self):**

**self.head = None**

**self.last\_node = None**

**def append(self, data):**

**if self.last\_node is None:**

**self.head = Node(data)**

**self.last\_node = self.head**

**else:**

**self.last\_node.next = Node(data)**

**self.last\_node = self.last\_node.next**

**def print\_middle(llist):**

**current = llist.head**

**length = 0**

**while current:**

**current = current.next**

**length = length + 1**

**current = llist.head**

**for i in range((length - 1)//2):**

**current = current.next**

**if current:**

**if length % 2 == 0:**

**print('The two middle elements are {} and {}.'**

**.format(current.data, current.next.data))**

**else:**

**print('The middle element is {}.'.format(current.data))**

**else:**

**print('The list is empty.')**

**a\_llist = LinkedList()**

**data\_list = input('Please enter the elements in the linked list: ').split()**

**for data in data\_list:**

**a\_llist.append(int(data))**

**print\_middle(a\_llist)**

**QUESTION 2:**

def binary\_search(arr, low, high, x):

# Check base case

if high >= low:

mid = (high + low) // 2

# If element is present at the middle itself

if arr[mid] == x:

return mid

# If element is smaller than mid, then it can only

# be present in left subarray

elif arr[mid] > x:

return binary\_search(arr, low, mid - 1, x)

# Else the element can only be present in right subarray

else:

return binary\_search(arr, mid + 1, high, x)

else:

# Element is not present in the array

return -1

# Test array

arr = [ 2, 3, 4, 10, 40 ]

x = 10

# Function call

result = binary\_search(arr, 0, len(arr)-1, x)

if result != -1:

print("Element is present at index", str(result))

else:

print("Element is not present in array")

OUTPUT:

Element is present at index 3