Data Science & Artificial Intelligence

Data Structure through Python

Super 1500+

Lecture No.- 04

By- Satya sir

Recap of Previous Lecture

- Hash Tables

- Queues

Topics to be Covered

- H/w Questions Solving

- Linked Lists

SUPER 1500+ - DSP- CLASS – 3 - Homework Question - 1

#Q. Consider a hash table of size 10 Hashing is done using the hash function as :
Hf(Key) = key mod 8

Find the number of collisions 6 when the below keys are inserted in hash table in given order if linear probing is used.

25, 39, 46, 55, 89, 23, 68

$h(25) = 25 \% 8 = 1$

$h(39) = 7$

$h(46) = 6$

$h(55) = 7$ Collission 1

$h(55,1) = (55+1) \% 8 = 56\% \; 0$

$h(89) = 89 \% 8 = 1$ Collission 2

$h(89,1) = 90 \% 8 = 2$

$h(23) = 7$ Collision 3

$h(23,1) = 0$ Collision 4

$h(23,2) = 1$ Collision 5

$h(23,3) = 2$ Collision 6

$h(23,4) = 3$

0	55
1	25
2	89
3	23
4	68
5	
6	46
7	39
8	
9	

$h(68) = 4$

SUPER 1500+ - DSP- CLASS – 3 - Homework Question - 2

#Q. Consider a hash table of size seven, with starting index zero, and a hash function (3x + 4)mod7. Assuming the hash table is initially empty, which of the following is the contents of the table when the sequence 1, 3, 8, 10 is inserted into the table using closed hashing? Note that '_' denotes an empty location in the table.

(A) 8, _, _, _, _, _, 10

[✓ (B) 1, 8, 10, _, _, _, 3]

(C) 1, _, _, _, _, _, 3

(D) 1, 10, 8, _, _, _, 3

0	1	2	3	4	5	6
1	8	10	_	_	_	3

$h(1) = [3(1)+4] \% 7 = 7 \% 7 = 0$

$h(3) = [3(3)+4] \% 7 = 13 \% 7 = 6$

$h(8) = [3(8)+4] \% 7 = 28 \% 7 = 0 \rightarrow 1$

$h(10) = [3(10)+4] \% 7 = 34 \% 7 = 6 \rightarrow 0 \rightarrow 1 \rightarrow 2$

SUPER 1500+ - DSP- CLASS – 3 - Homework Question - 3

#Q. Consider a hash table of size 11 that uses open addressing with linear probing. Let h(k) = k mod 11 be the hash function used. A sequence of records with keys

43, 36, 92, 87, 11, 4, 71, 13, 14

is inserted into an initially empty hash table, the bins of which are indexed from zero to ten. What is the index of the bin into which the last record is inserted?

(A) 2
(B) 4
(C) 6
[✓ (D) 7]

0	87
1	11
2	13
3	36
4	92
5	4
6	71
7	14
8	
9	
10	43

SUPER 1500+ - DSP- CLASS – 3 - Homework Question - 4

#Q. Consider the circular queue given below which has FRONT =1 and REAR=5

f=1	f=2	f=3	4	r= 5	r= 6
~~A~~	~~B~~	~~C~~	~~D~~	~~E~~	F
G	H	I			
f=1	~~f~~	r=3			

Now perform the following sequence of operations on the queue

i. Enqueue F
ii. Dequeue 2 letters
iii. Enqueue G
iv. Enqueue H
v. Dequeue 4 letters
vi. Enqueue I

f=1

r=3

Sum=1+3=4

What would be the sum of the positions of the rear and front references?

#Q. What does the following function do for a given Linked List with first node as head?

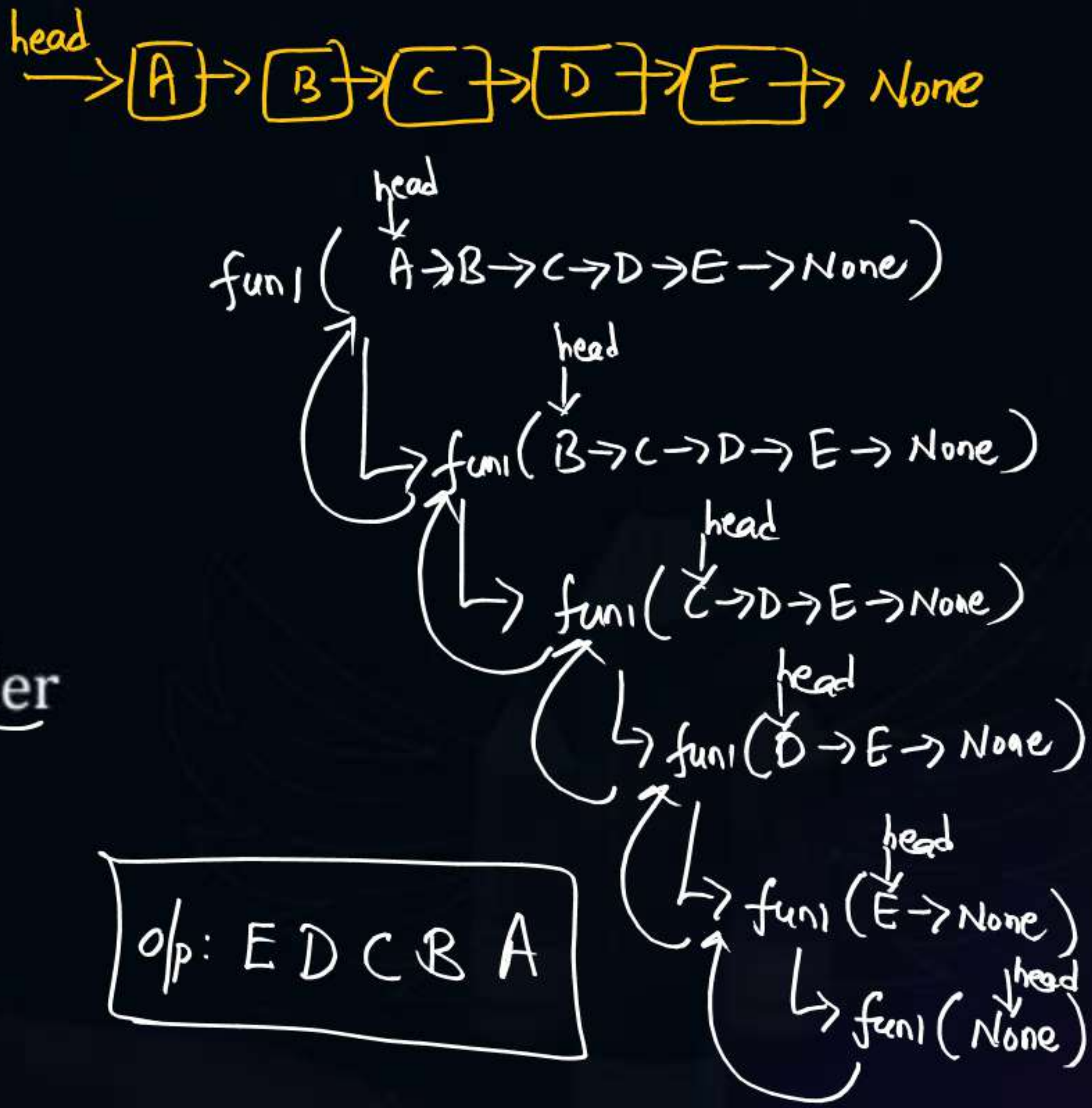
```
def fun1(head):
    if head is None:
        return
    fun1(head.next)
    print(head.data, end=' ‘)
```

a) Prints all nodes of linked lists

[✓ b) Prints all nodes of linked list in reverse order]

c) Prints alternate nodes of Linked List

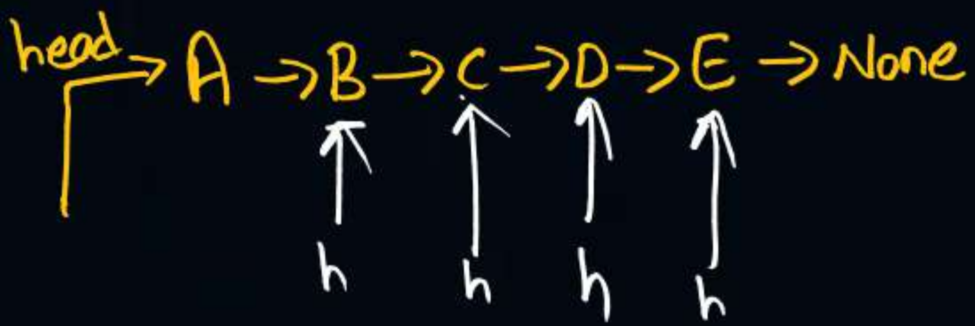
d) Prints alternate nodes in reverse order



o/p: E D C B A

#Q. What does the following function do for a given Linked List with first node as head?

```
def f(head):
    if head.next is None:
        return
    print(head..next.data, end='  ‘)
    f(head.next)
```



o/p: B, C, D, E,

a) Prints all nodes of linked lists

[✓ b) Prints all nodes of linked list except first node]

c) Prints alternate nodes of Linked List

d) Prints all nodes of linked list except last node

#Q. Which of the following operations is performed more efficiently by doubly linked list than by singly linked list?

→ SLL ⇒ O(n)

→ DLL ⇒ O(1)

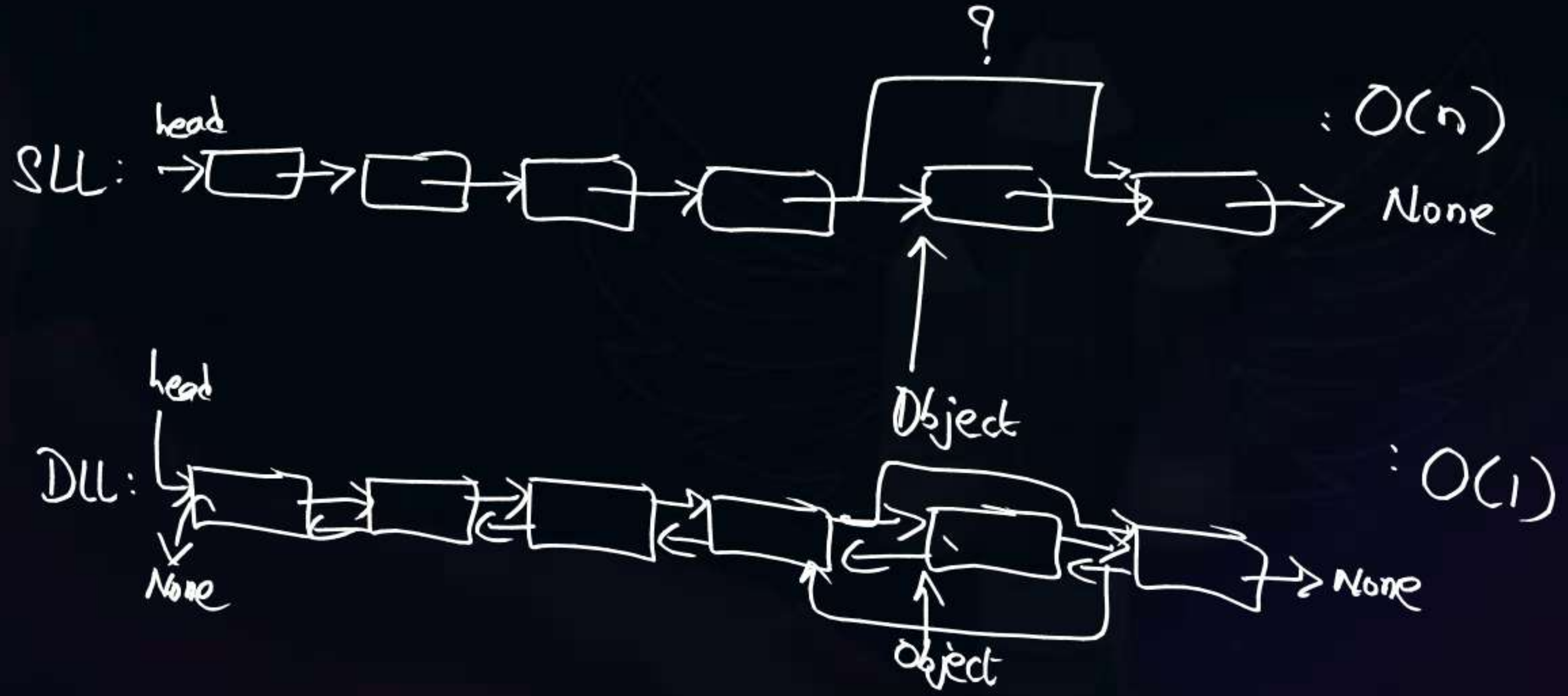
a) Deleting a node whose location in given

b) Searching of an unsorted list for a given item

c) Inverting a node after the node with given location

d) Traversing a list to process each node

$\underline{\underline{O(n)}}$



#Q. The following function takes a single-linked list (lst) of integers as a parameter and rearranges the elements of the list. The function is called with the list (lst) containing the integers 1, 2, 3, 4, 5, 6, 7 in the given order. What will be the contents of the list after the function completes execution?

```
class Node:
    def __init__(self, value=0, next=None):
        self.value = value
        self.next = next

def rearrange(lst):
    if not lst or not lst.next:
        return
    p = lst
    q = lst.next
    while q:
        p.value, q.value = q.value, p.value
        p = q.next ✓
        q = p.next if p else None
```

not lst → Empty list

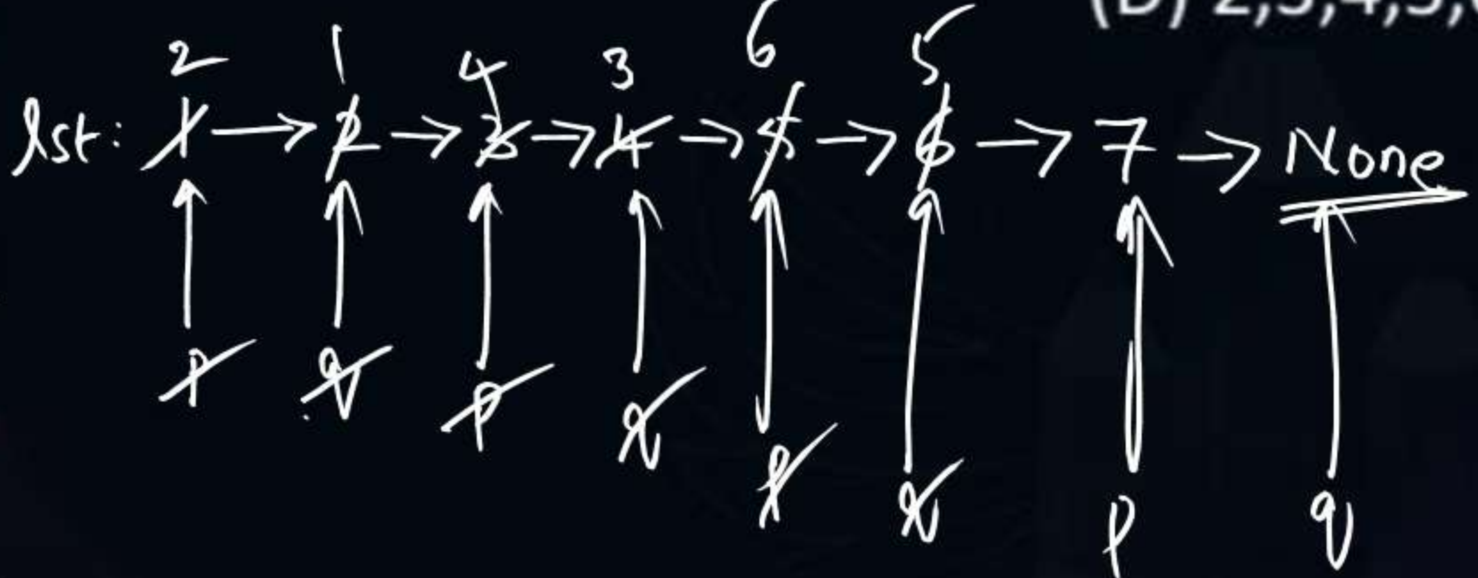
not lst.next → only 1 Element

(A) 1,2,3,4,5,6,7

✓ (B) 2,1,4,3,6,5,7

(C) 1,3,2,5,4,7,6

(D) 2,3,4,5,6,7,1



(Reference|object)

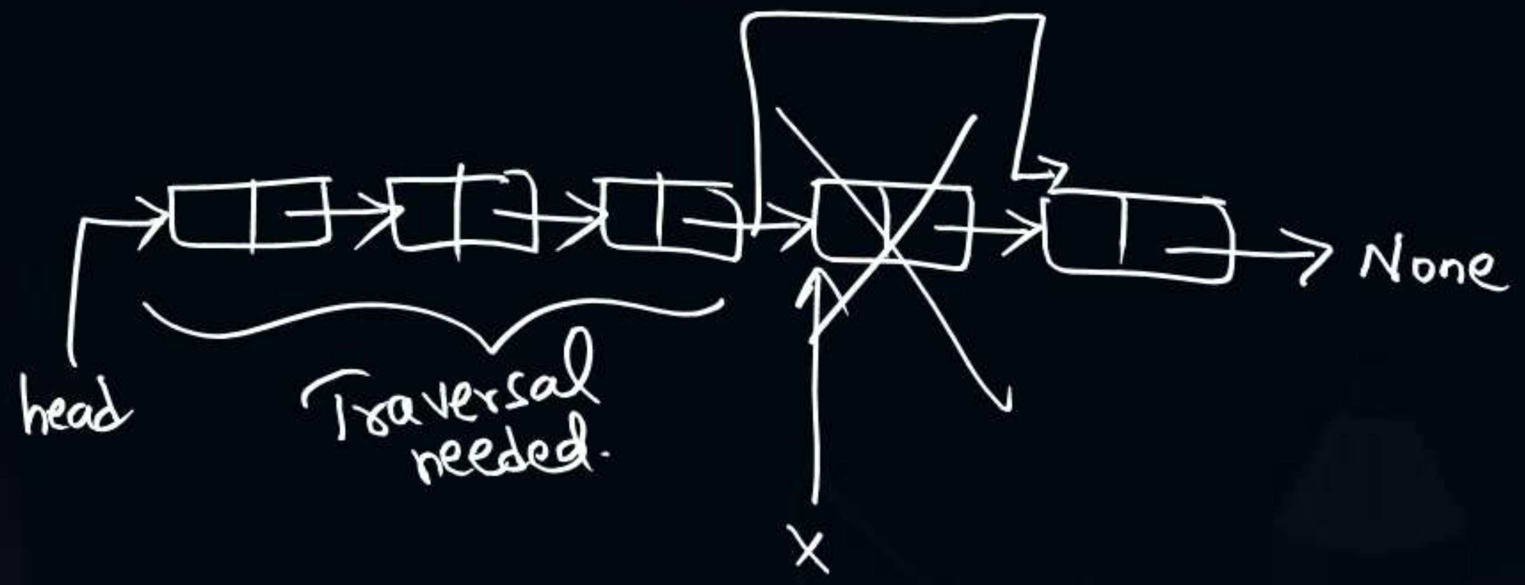
#Q. Let P be a singly linked list. Let Q be the pointer to an intermediate node x in the list. What is the worst-case time complexity of the best known algorithm to delete the node x from the list?

[✓ (A) O(n)]

(B) O(log2 n)

(C) O(logn)

(D) O(1)

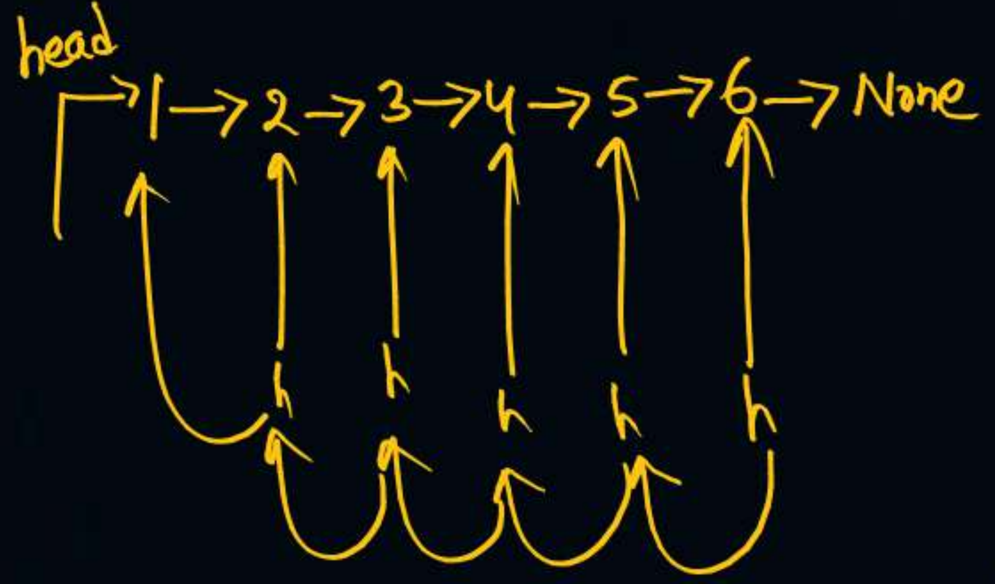


Intermediate Node ⇒ Neither first, Nor last Node.

#Q. What does the following function print for a given Linked List with input 1,2,3,4,5,6?

```
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None

def fun(head):
    if head.next is None:
        return
    fun(head.next)
    print(head.data, end=' ‘)
```



O/P: 5 4 3 2 1

- a) ~~1,2,3,4,5~~ (5,4,3,2,1) ✓
- b) 1,2,3,4,6
- c) 1,2,3,4
- d) 1,2,3,4,5,6

SUPER 1500+ - DSP- CLASS – 4 - Homework Question - 1

#Q. What does the following function print for a given Linked List with input 1,2,3,4,5,6?

```
void fun1(struct node* head)
{
  if(head->next == NULL)
  return;
  printf("%d ", head->data);
  fun1(head->next);
  printf("%d ", head->data);


}
```

a) 1,2,3,4,5,6,6,5,4,3,2,1

b) 1,2,3,4,4,3,2,1

c) 1,2,3,4,5,5,4,3,2,1

d) 1,2,3,4,5,1,2,3,4,5

SUPER 1500+ - DSP- CLASS – 4 - Homework Question - 2

#Q. Consider an implementation of unsorted singly linked list. Suppose it has its representation with a head pointer only. Given the representation, which of the following operation can be implemented in O(1) time?

A) Insertion at the front of the linked list
B) Insertion at the end of the linked list
C) Deletion of the front node of the linked list
D) Deletion of the last node of the linked list

MSQ

SUPER 1500+ - DSP- CLASS – 4 - Homework Question - 3

#Q. Which of the following points is/are not true about Linked List data structure when it is compared with array?

a) Arrays have better cache locality that can make them better in terms of performance

b) It is easy to insert and delete elements in Linked List

c) Random access is not allowed in a typical implementation of Linked Lists

d) Access of elements in linked list takes less time than compared to arrays

2 mins Summary

- Linked Lists

NEXT CLASS TOPIC: ~~LINKED LISTS~~ TREES

THANK - YOU