MBE (expert Interview)

↓

- ✓ DSA →
- ✓ SQL →
- ✓ LLD →
- ✓ HLD →
- ✓ Project →

MBE ⇒ ready

(60 minute interview)

20 mentor sessions

Interview (12)   Interactive (8)

(7) Floater (Practice)   MBE (5)

Cool down (2 weeks)

VIVA ⇒ 13th & 14th May

(30 min interview)

Arrays , Bit Manipulation

Intermediate ⇒ 5th ⇒ Linked List

8th ⇒ Trees.

9 (Backlog)

↓

Adv DSA ⇒ after 17th

Context

Reattempts

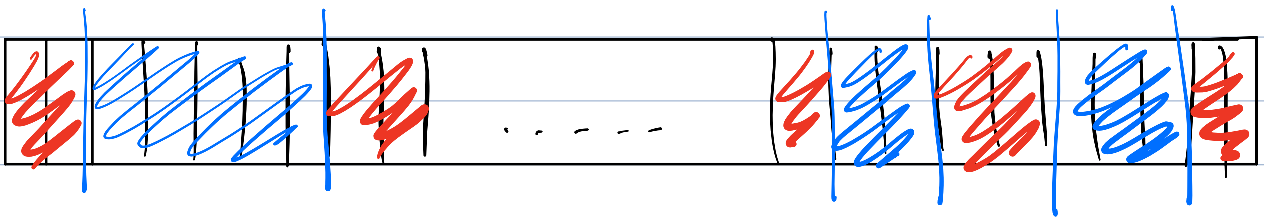
Neoversity ?? (Masters) ⇒ ECTS

7th May (11:00AM)

AMA (Ask me anything)

Abhimanyu & Mr. Joshua

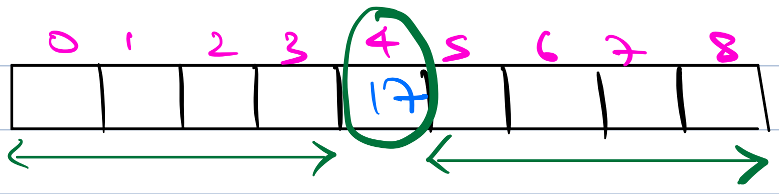
Array ⇒ Contiguous list of homogeneous elements.



RAM ⇒ 32 Bytes
8 Bytes
8 Bytes
48

N=12

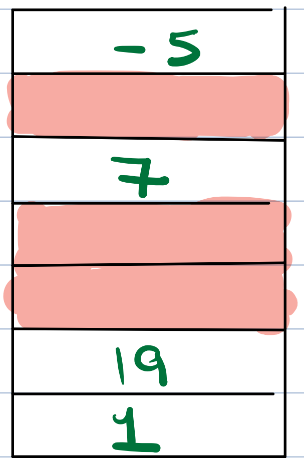
int A[12]; ⇒ 12×4 = 48 Bytes



Delete (4)

(N=9)

Linked List

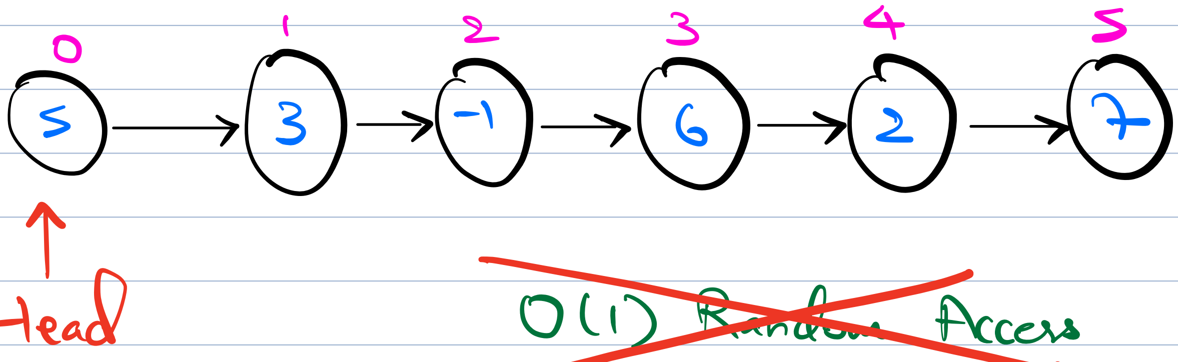


Head of Linked List.

Add 1
Add 7
Add -5
Add 19

Every element stores where the next element is present.

assume 1 Block = 4 Bytes



~~O(1) Random Access~~

Classes & Objects.

Class ⇒ Blueprint → Attributes / Properties.
→ Functionalities (Methods)

Object ⇒ Real Instance of Blueprint.

```
class Car {

    color
    seats
    power
    brand

    accelerate ()
    break ()
    music ()

}
```

```
Shashider
    color : grey
    seats : 7
    brand : bmw

    accelerate()
    break()
    music()
}
```

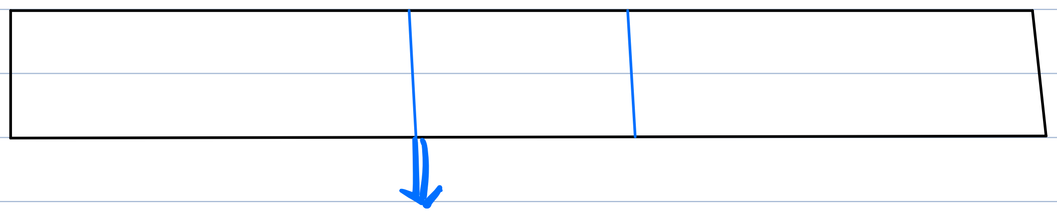
```
class Student {

    int roll_no;
    String name;
    int p, c, m;

    int totalMarks() {
        return p+c+m;
    }

    int percentage() {
        return ((p+c+m)/300) x 100;
    }
}
```

Reference variable > Address of object

$$\underbrace{\text{Student}}_{\text{Name of class}} \quad \underbrace{\text{S1}}_{\text{name of Object}} = \text{new Student();}$$

```
S1.roll_no = 17;
S1. name = "Sheela"
S1. p =
S1. c =
S1. m = 100
```

```
Student  s2 = s1;
s2.name = "Hemanth"

print (s1.name) ⇒  Hemanth
```



Mem Location = 51235

S1 = 51235
S2 = 51235

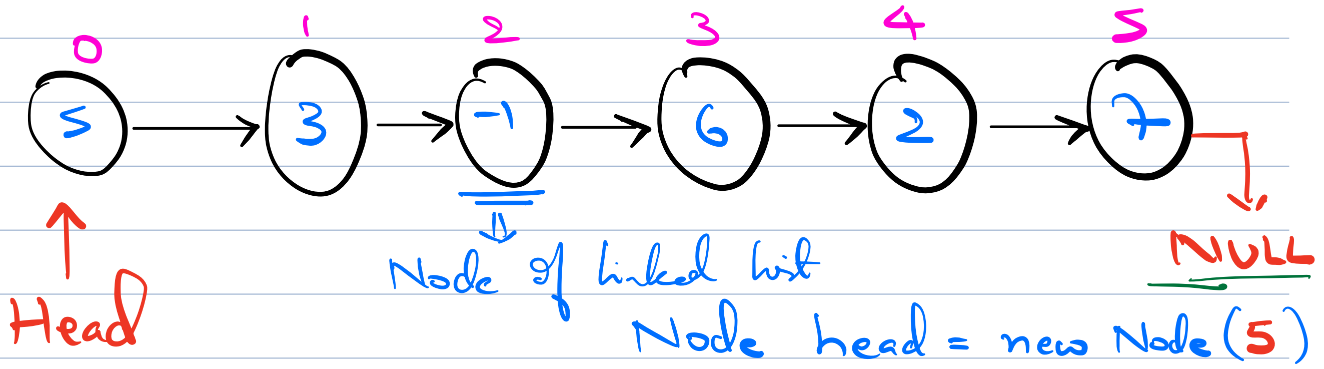
Constructor

```
class Student {

    int roll_no;
    String name;
    int p, c, m;

Student (roll_no, name, p, c, m) {
    this.roll_no = roll_no;
```

No ~~return~~ type

(attribute of class)

```
    int totalMarks() {
        return p+c+m;
    }

    int percentage() {
        return((p+c+m)/300)×100;
    }
}
```

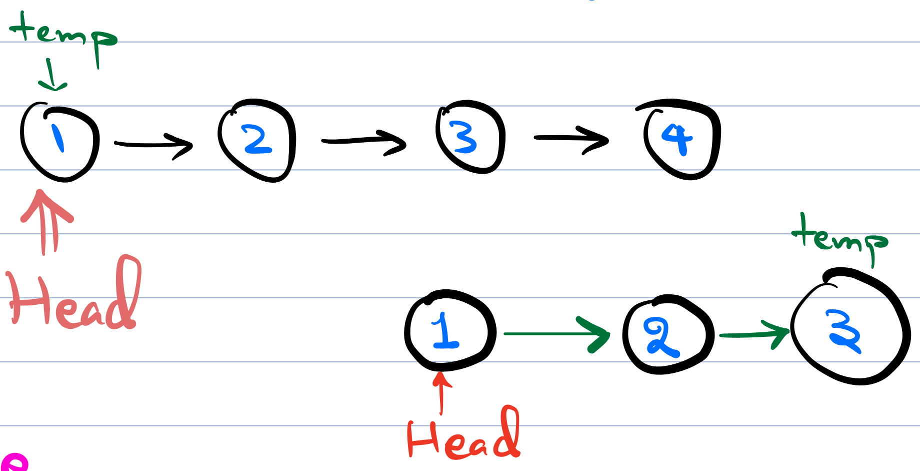


Node head = new Node(5)

```
class Node {

    int data;
    Node next;      (Reference of Next Element)

    Node ( int d) {
        data = d;
        next = NULL;
    }
}
```

Q Given an integer array of size N. Create a LL of the elements of the array.

A = [1, 2, 3, 4]



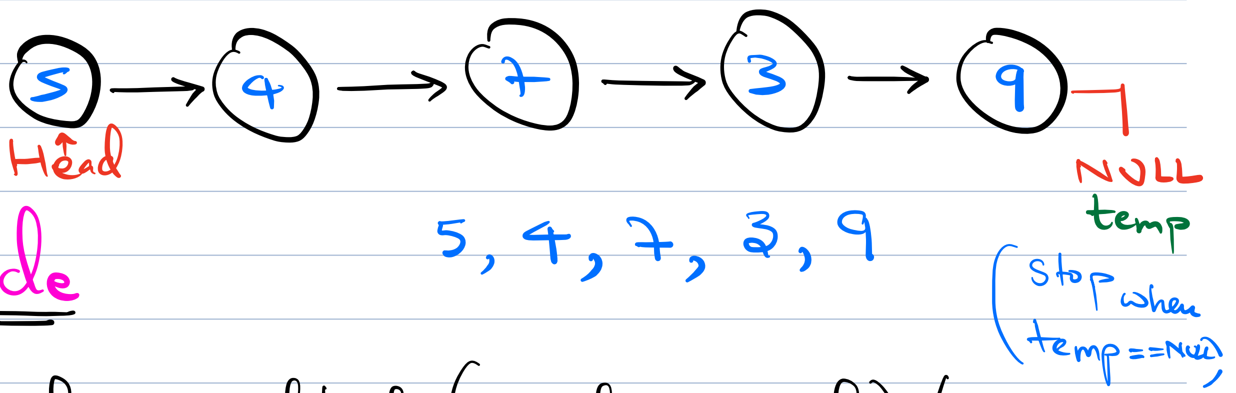
Code

```
Node createList (int[] A, int N){
        Node head = new Node (A[0]);
        Node temp = head;

        for(i=1; i<N; i++){
            temp.next = new Node (A[i]);
            temp = temp.next; (move temp)
        }
        return head;
}
```

Iterating a LL

Q Given the Head Node of a linked list.

Print all elements.



Code

5, 4, 7, 3, 9

```
void printList (Node Head) {
    Node temp = Head;
    while ( temp != NULL) {
                print (temp. data);
                temp = temp. next;
    }
}
```

		0	1	2	3	4	5	6	7
0	:	0							
1	:	0	1						
2	:	0	1	1	0				
3	:	0	1	1	0	1	0	0	1

	(Even	$0 \to 0, 1$
$0 \nearrow$	1	$1 \to 2, 3$
Eve		$2 \to 4, 5$
0	1	$3 \to 6, 7$
odd	odd	