Today's agenda

- Classes and objects
- Constructor
- Nested Class
- LinkedList Intro
- Point LinkedList

// mid Calculation

lo hi

int m = (lo + hi) / 2

XX

lo hi

int m = lo + (hi - lo) / 2

=

Prevent overflow

-2^{31}

$2^{31}-1$

lo hi

1 $2^{31}-1$

m = $\frac{1 + 2^{31}-1}{2} \rightarrow 2^{31}$

int m = lo + (hi - lo) / 2

lo hi

1 $2^{31}-1$

m = $1 + \frac{(2^{31}-1 - 1)}{2}$

//Classes and objects

int, Char, String, double

↳ int x = 20;

→ To combine different data types and data structures.

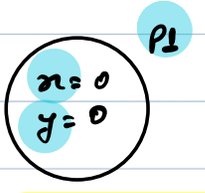
*

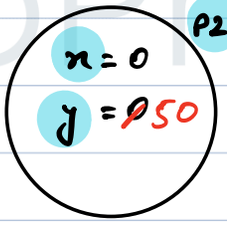
```
Public static class Pair {
    int x;
    int y;
}
```

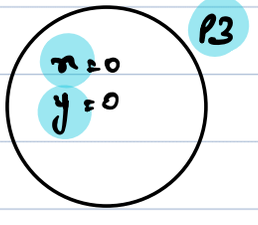
Pair P1 : new Pair();

Pair P2 : new Pair();

Pair P3 : new Pair();







System.out.println(P2.y); → 0

P2.y = 50;

System.out.println(P2.y); → 50

* class: It is a bluePrint.

* object: real instance of blue Print

```
Class home {
  int roomCount
  int floorCount
  int area
  String Color
  boolean Pool
}
```

Sha22y

```
home Sha22y = new home()
Sha22y.roomCount = 4
Sha22y.floorCount = 2
Sha22y.Color = "red"
boolean Pool = true;
```

roomCount = 04
floorCount = 02
area = 0
Color = "red"
Pool = ~~false~~ true

Sha22y

Chandra Sai

```
home Sai = new home();
```

Sai

Adarsh

```
home adarsh = new home();
```

// Constructor → To make your life easy.

```
Public static class Pair {
    int x;
    int y;
}
```

```
Pair p1 = new Pair();
p1.x = 10;
p1.y = 20;
```

10
20

x = ~~0~~ 10
y = ~~0~~ 20

p1

```
Public static class Pair {
    int x;
    int y;
    Pair(int v1, int v2) {
        x = v1;
        y = v2;
    }
    Pair() {
    }
    Pair(int v1) {
    }
}
```

```
Pair p1 = new Pair(10,20);
```

x: 10
y: 20

p1

```
Pair p2 = new Pair();
```

x=0
y=0

int x = "Hello"
Pair P1 = 10;
→ while assigning, type should be same.

Break till 9:28 PM

//Nested class

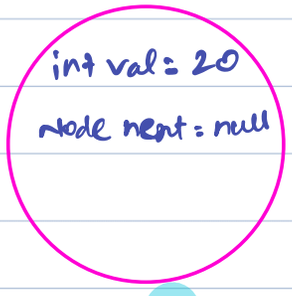
```
Class Node {
    int val;
    Node next;

    Node (int v1){
        val = v1;
    }
}
```

Node n1 = new Node (10);

n1

Node n2 = new Node (20);

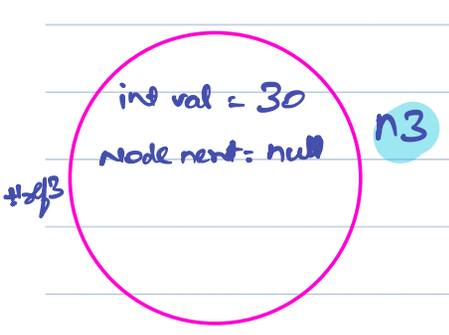


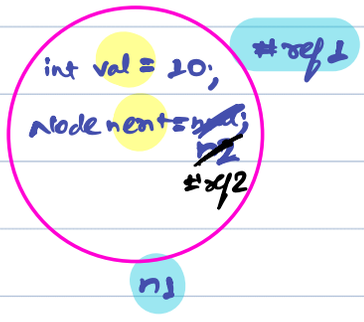
n2

n1.next = n2;

Print (n1.next.val); → 20

Node n3 = new Node(30); Node n1 = new Node(10);

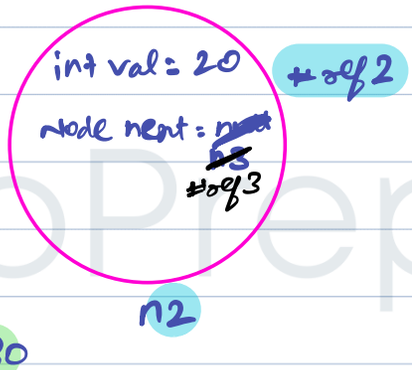




n1.next = n2;

n2.next = n3;

Node n2 = new Node(20);

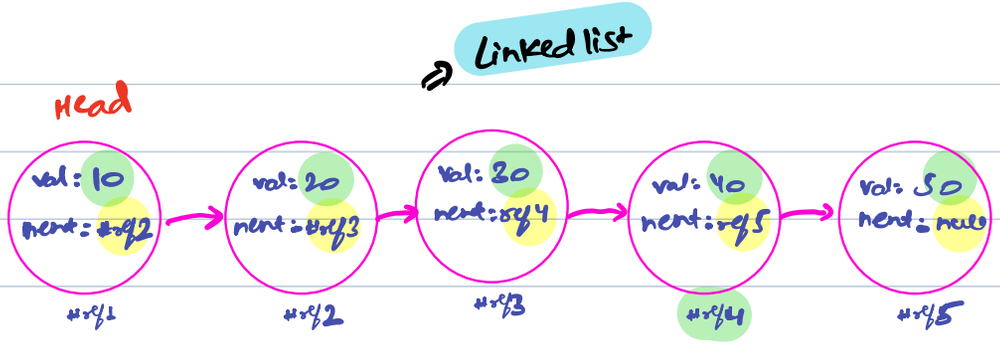


Print(n2.val); → 20

Print(n2.next.next); → null

Print(n1.next.next.val); → 30

Print(n1.next); → reference of n2



Print (Head.val); → 10

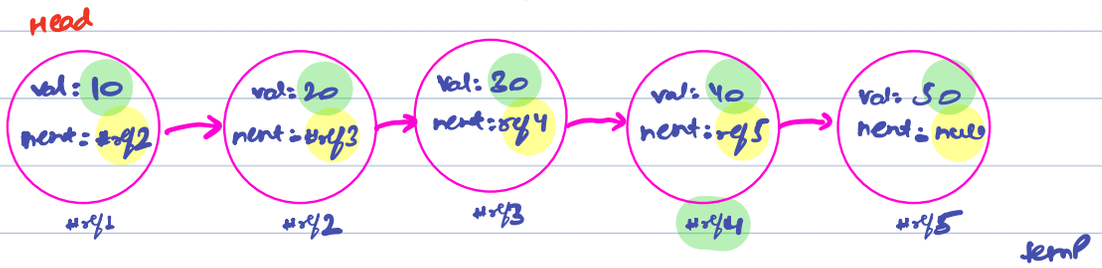
Print (Head.next.val); → 20

#ref2 #ref3 #ref4

Print (Head.next.next.next) → #ref4

Q) Print LinkedList

Given head of the LL Print the elements.




```
void PrintLinkedList (Node head){

    Node temp = head;

    while (temp!=null){

        Print (temp.val);
        temp= temp.next;

    }
}
```

10 20 30
40 50

T.C: O(N)
S.C: O(1)