Today's Content :-

a) Class & Object Concept → oops * Design

b) Object & Object reference → member.

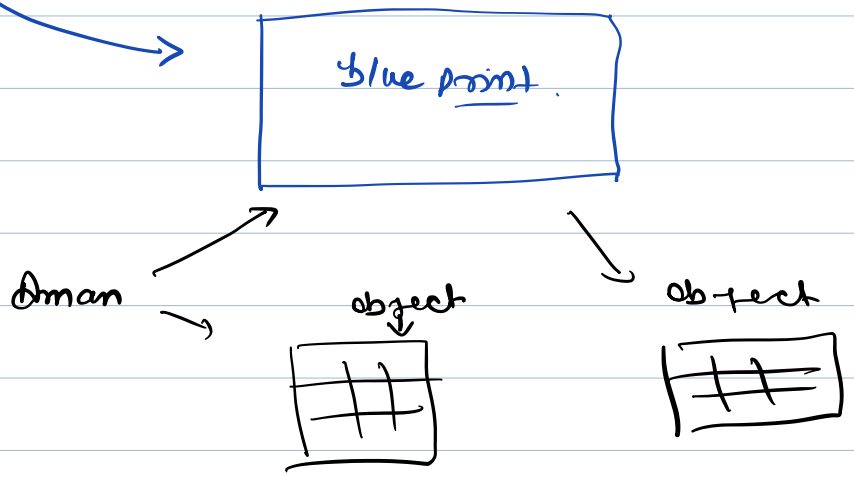
c) Object reference as mememberr

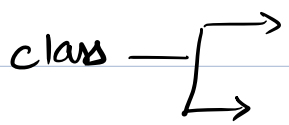
d) Linked list Basics

e) Problems in Linkedlist

Class :- It is a blue print.

Object :- It is an instance of class.





class Car {

Name
Colour
Milege
drive()
AC()
}

Attributes of class

Car : Ankush

Name : Creta
Colour : black
Milege : 12 kmpl
drive()
ac()

Car : Pavan
Name : XUV700
Colour : Silver
Milege : 10 KMPL
drive()
ac()

3

Class → Attribute
→ functionalities.

```
class Student {
    String name;
    int age
    bunk()
    exams()
    Proposing()
}
```

Student s1 = new Student()

s1 → name = mangesh, Age = 19

s1.name = 'Mangesh'
s1.age = 19.

Student s2 = new Student();

s2 → name = 'Dyanesh', Age = 25

s2.name = 'Dyaneshwar'
s2.age = 25

Object and Object reference

User defined data type.

```
class Student {
    String name;
    int age
    funic()
    exams()
    proposing()
}
```

```
Student s = new Student();
```

obj reference of student class

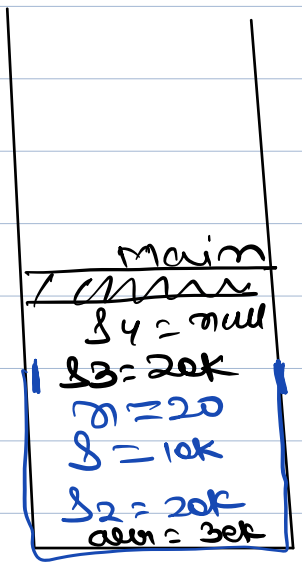
obj of student class

Obs:- Obj reference of a class can hold address of that particular class object

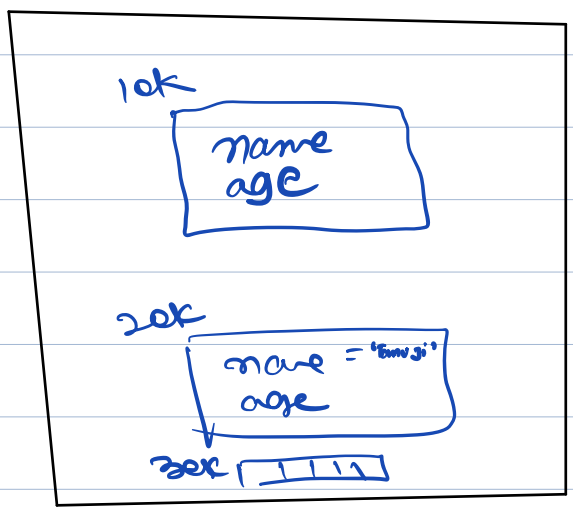
```
int a;
char ch;
Student s;
```

(int, String)

Stack



Heap

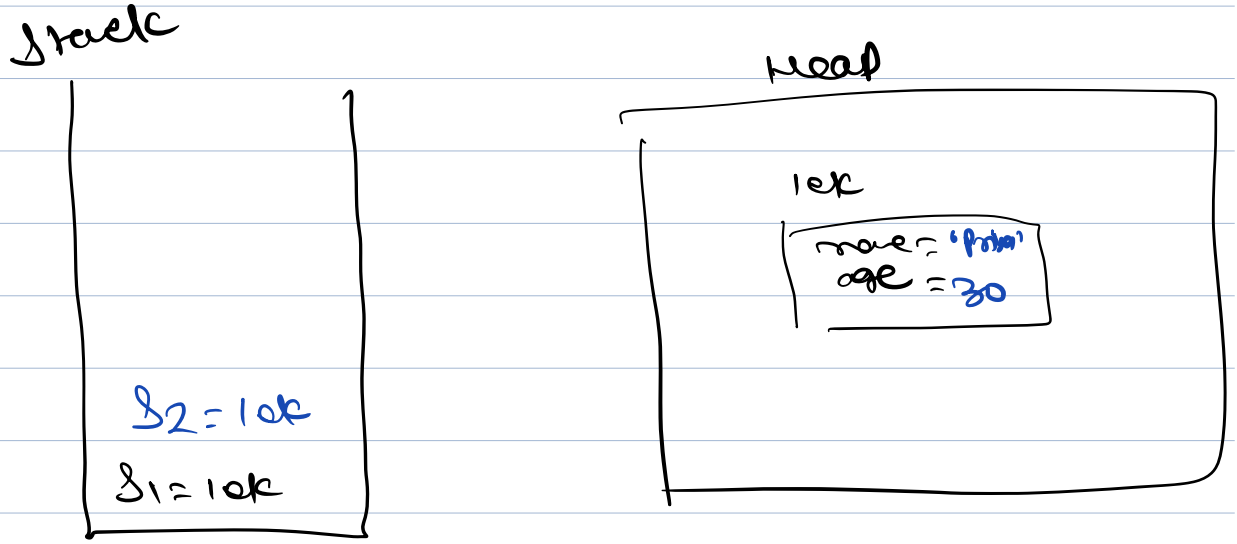


```
main() {
    Student s = new Student();
    Student s2 = new Student();
    int[] arr = new arr[5];
    Student s3;
    s3 = s2;
    s2.name = 'Tanuji'
    print(s3.name); // Tanuji.

    Student s4;
    print(s4.name);

int n = 20;
}
```

error → null pointer Exception.

*) Multiple object reference.

```
Student s1 = new Student();
s1.name = 'Abhishek'
s.Age = 25
Student s2 = s1;
s2.age = 30
print(s1.age));
s1.name = "Priya";
print(s2.name);
```



(*) class pair {
    int x;
    int y;
}

pair p1 = new pair();
p1.x = 40
p1.y = 50 // initialize data;

p1 → [x = 40, y = 50]

* Constructor; → Concept
  ↳ It is used to initialize attributes of class at the time of object creation itself.

```
class Pair {
    int x;
    int y;
    Pair (int a, int b) {
        x = a;
        y = b;
    }
}
```

a) Constructor name must be same as class name;

b) No return type;

Constructor.

Pair P1 = new Pair (3, 10);

P1. x = 122

P1 → | x = 122 |
     | y = 10 |

Pair P2 = new Pair (30, 40);

P2 → | x = 30 |
     | y = 40 |

10:05 - 10:15

kgf-1 kgf-2

```
class Node {
    int val;
    Node next;
    Node(int x) {
        val = x;
        next = null
    }
}
```

```
Node h = new Node(10);
```

h = 20k
l = 40k

20k

val = 10
next = 30k.

address of node object.

```
Node l = h;
l.next = new Node(5);
l = l.next;
```

```
l.next = new Node(15);
l = l.next
```

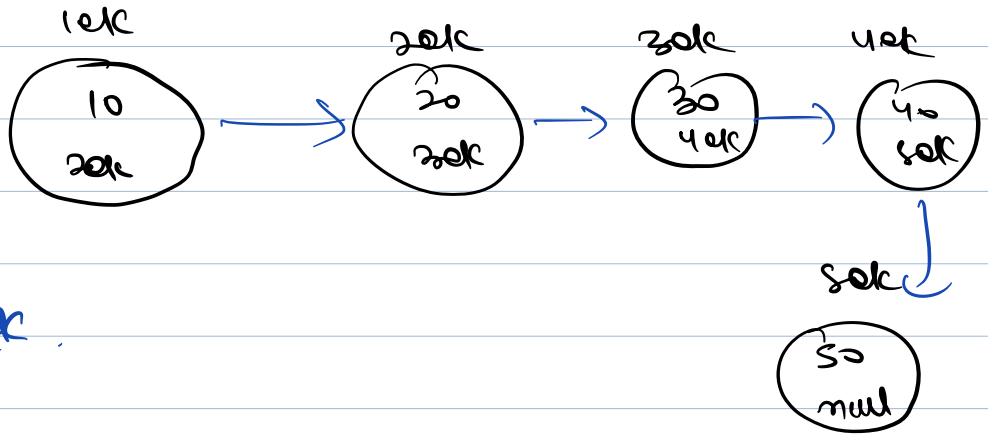
30k

val = 5
next = 40k

40k

val = 15
next = null.

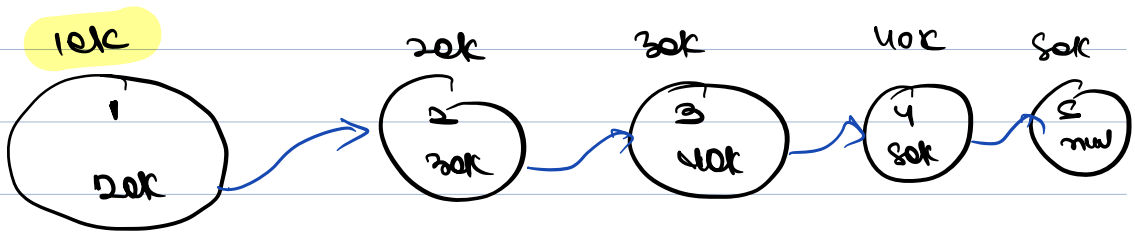
Linked List



head = 10k

Ques) Create a LL with n nodes, with data 1-n, return head node.

Ex, N = 5

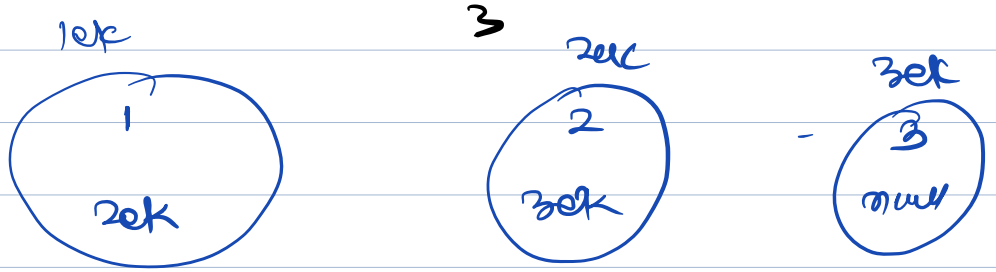



```
class Node {
    int val;
    Node next;

    Node (n) {
        val = n;
        next = null
    }
}
```

```
Node CreateList (int n) {
    Node h = new Node (1);
    Node t = h;
    for (i = 2; i <= n; i++) {
        t.next = new Node (i);
        t = t.next;
    }
}
```

i = 3.

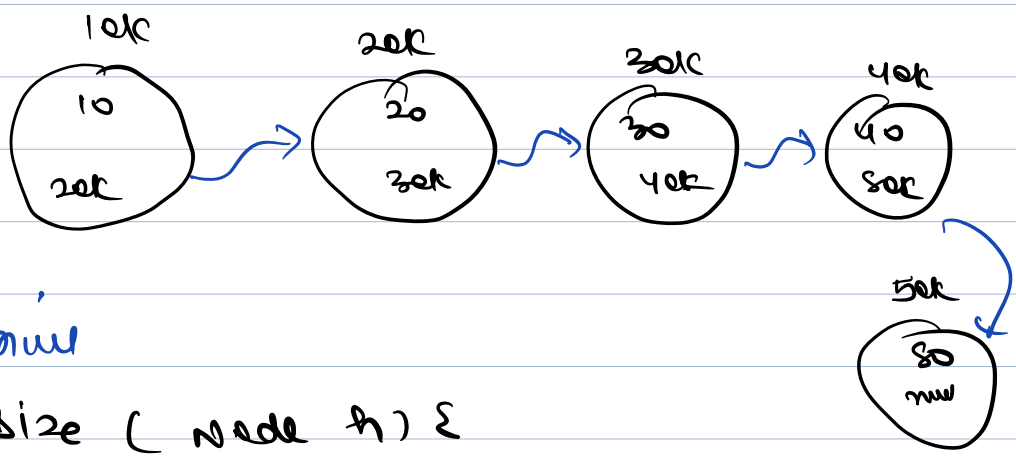
return h



h = 10k

d = 30k

Ques) Given head node of LL, return size,



h = 10k ,

t = null

```
int size ( Node h) {
    Node t = h
    int c = 0;
    while ( t != null) {
        t = t.next
        c++
    }
    return c;
}
```

t

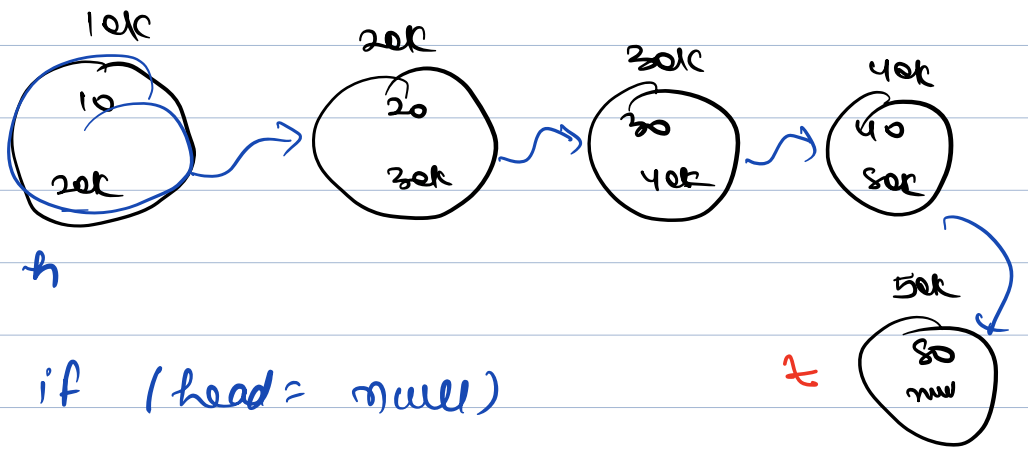
c = 1 2 3 4 5.

(when h is null it will fail)

```
int size ( Node h) {
    Node t = h
    int c = 1
    while ( t.next != null) {
        t = t.next
        c++
    }
    return c;
}
```

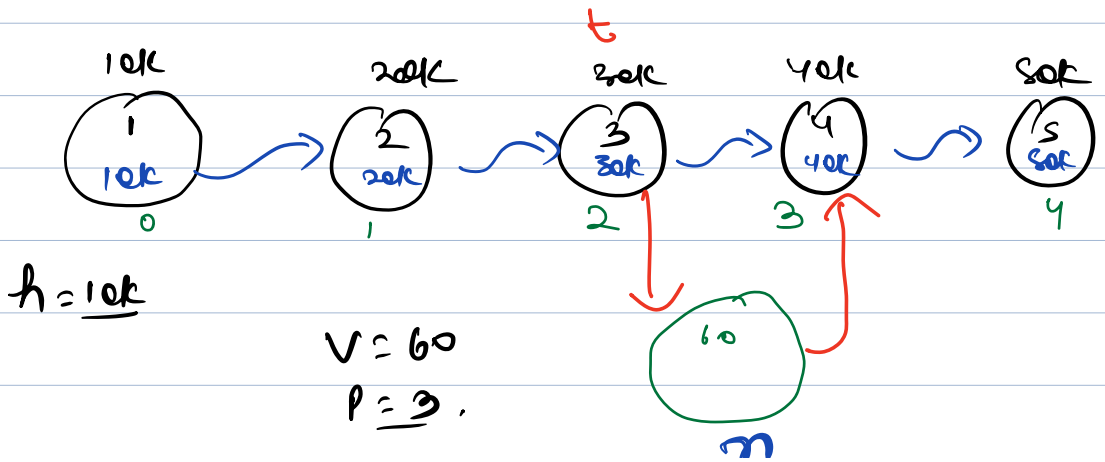
h = 10k
t = 10k

c = 4 2K BYS.



if (head = null)

(Ques) Given a LL, insert a new Node with data V at idx P.



h = 10k

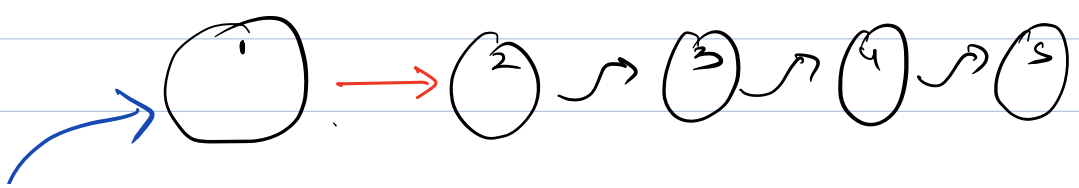
V = 60

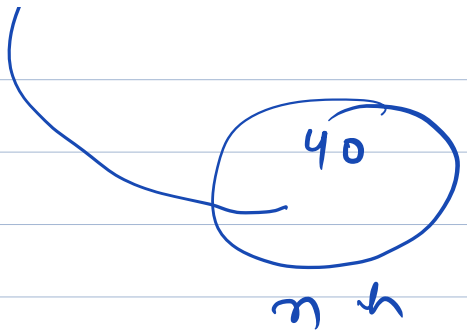
P = 3.

```
Node Insert (Node h, int v, int p){
Node n = new Node (v);

if (p==0) {
    n.next= h;
    h = n
    return h ;
}

Node t= h;
for (int i=1; i<p; i++){
    t= t.next
}

n.next= t.next;
t.next= n;
return h;
}
```

pos = 0, ele 40

pos = 0, t = 40






```
Node CreateList(int n){
  Node h = new Node(1);
  Node l = h;
  for(i=2; i<=n; i++){
    l.next = new Node(i);
     l = l.next;
  }
  return h
}
```

fun(1)

```
func (int n)
{
    func(n+1
}
```

3

(int) ((he%C) * (A%C))%C; and this: (int) (he%C * A%C)%C;

Arrays

Bit

recursion

for, arrays & I.O

Revision → every day.

↓

weekend

↓

Node c;

int arr[5];

monday.

Stack heap